

Microsoft®
SQL Server®

2019



کتاب آموزشی SQL Server 2019

مؤلف: فرشید بابجانی زاده

سرشناسه: بابجانی زاده، فرشید، ۱۳۶۵

عنوان و نام پدیدآور: کتاب آموزشی SQL Server 2019 / فرشید بابجانی زاده

مشخصات نشر: 3isco.ir

مشخصات ظاهری: ۴۶۳ ص

شابک:

وضعیت فهرست نویسی: فیا

موضوع:

موضوع:

رده بندی کنگره PIR:

رده بندی دیویی:

شماره کتاب شناسی ملی:

تقدیم بہ
روح بلند مادرم...

فهرست

۱۱	مقدمه
۱۳	فصل اول
۱۳	مباحث بنیادی پایگاه داده
۱۳	۱-۱ بررسی اولیه
۱۳	۱-۱-۱ داده چیست
۱۴	۱-۱-۲ اطلاعات چیست
۱۴	۱-۱-۳ موجودیت چیست
۱۵	۱-۱-۴ پایگاه داده چیست
۱۵	۱-۱-۴-۱ ویژگی‌های داده در پایگاه داده
۱۶	۱-۱-۴-۲ روش‌های ذخیره داده
۱۷	۱-۱-۴-۳ سیستم پایگاه داده (database system)
۱۷	۱-۱-۴-۴ عناصر اصلی سیستم پایگاه داده
۱۸	۱-۱-۴-۵ مزایا و معایب سیستم‌های پایگاه داده
۱۹	۱-۱-۵ سیستم مدیریت پایگاه داده
۱۹	۱-۱-۵-۱ وظایف سیستم مدیریت پایگاه داده
۲۰	۱-۱-۵-۲ تراکنش
۲۱	۱-۱-۵-۳ اجزای سیستم مدیریت پایگاه داده
۲۲	۱-۱-۶ انواع سیستم‌های مدیریت پایگاه داده
۲۲	۱-۱-۶-۱ سیستم مدیریت پایگاه داده توزیع شده
۲۳	۱-۱-۶-۲ سیستم مدیریت پایگاه داده بلادرنگ
۲۳	۱-۱-۶-۳ سیستم مدیریت پایگاه داده تحمل‌پذیر خطا
۲۳	۱-۱-۶-۴ سیستم مدیریت پایگاه داده امن
۲۴	۱-۱-۶-۵ سیستم مدیریت پایگاه داده ناهمگون

۲۴.....	۱-۱-۶-۶ سیستم مدیریت پایگاه داده چندرسانه‌ای
۲۴.....	۱-۱-۶-۷ سیستم مدیریت پایگاه داده متحد
۲۴.....	۱-۱-۷ کاربران پایگاه داده
۲۴.....	۱-۱-۷-۱ تحلیل گران سیستم
۲۴.....	۱-۱-۷-۲ طراحان پایگاه داده
۲۵.....	۱-۱-۷-۳ پیاده سازان برنامه‌های کاربردی
۲۵.....	۱-۱-۷-۴ مدیر پایگاه داده
۲۵.....	۱-۱-۷-۵ کاربران نهائی
۲۶.....	۱-۱-۸ دیکشنری داده
۲۶.....	۱-۱-۹ پایگاه داده XML
۲۷.....	فصل دوم
۲۷.....	پیاده‌سازی SQL Server 2019
۲۷.....	۲-۱ نیازمندی‌های SQL Server 2019
۳۰.....	۲-۲ نصب و راه‌اندازی SQL Server 2019 در ویندوز
۴۱.....	۲-۳ دسترسی از طریق شبکه به SQL Server
۵۰.....	۲-۴ نصب و راه‌اندازی SQL Server 2019 در لینوکس Ubuntu
۶۱.....	فصل سوم
۶۱.....	آشنایی با پایگاه داده
۶۱.....	۳-۱ ایجاد پایگاه داده و کار با آن
۶۱.....	۳-۱-۱ ایجاد پایگاه داده
۶۳.....	۳-۱-۲ ایجاد جدول در دیتابیس
۶۶.....	۳-۱-۳ انواع Data Type در جداول
۷۰.....	۳-۱-۴ حذف دیتابیس در SQL Server

۷۱	۳-۱-۵ تعریف پرس و جو یا Query در SQL
۷۱	۳-۱-۵-۱ بررسی دستور SELECT
۷۵	۳-۱-۵-۲ بررسی دستور Insert
۷۶	۳-۵-۱-۳ بررسی دستور Update
۷۷	۴-۵-۱-۳ بررسی دستور DELETE
۷۸	۳-۱-۵-۵ بررسی دستور WHERE
۸۰	۳-۱-۵-۶ بررسی دستور AND، OR و NOT
۸۳	۳-۱-۵-۷ بررسی دستور LIKE
۸۴	۳-۱-۵-۸ بررسی دستور BETWEEN
۸۶	۳-۱-۵-۹ بررسی دستور IN
۸۶	۳-۱-۵-۱۰ بررسی دستور TOP
۸۸	۳-۱-۵-۱۱ بررسی دستور MIN and MAX
۹۰	۳-۱-۵-۱۲ بررسی دستور AVG، COUNT و SUM
۹۲	۳-۱-۵-۱۳ بررسی Wildcards
۹۴	۳-۱-۵-۱۴ بررسی دستور Aliass
۹۵	۳-۱-۵-۱۵ بررسی دستور GROUP BY
۹۷	۳-۱-۵-۱۶ بررسی دستور HAVING
۹۷	۳-۱-۵-۱۷ بررسی PRIMARY KEY
۹۹	۳-۱-۵-۱۸ بررسی JOIN در SQL
۱۰۷	۳-۱-۵-۱۹ بررسی دستور synonym
۱۱۰	۳-۱-۶ کار با View در SQL
۱۱۷	۳-۱-۷ بررسی FileStream در SQL Server
۱۲۱	۳-۱-۷-۱ ایجاد دیتابیس برای استفاده از FILESTREAM
۱۲۴	۳-۱-۷-۲ ایجاد جدول در دیتابیس FILESTREAM

۱۲۶.....	۳-۱-۸ ارتباط با SQL از طریق Visual Studio
۱۲۸.....	۳-۱-۸-۱ ایجاد دیتابیس از طریق دستورات در Visual Studio
۱۳۰.....	۳-۱-۸-۲ ایجاد دیتابیس از طریق ابزار Visual Studio
۱۳۳.....	۳-۱-۸-۳ ایجاد فرم در Visual Studio و ثبت اطلاعات در جدول SQL
۱۳۶.....	۳-۱-۹ وارد کردن فایل اکسل در SQL
۱۴۰.....	۳-۱-۱۰ بررسی دستور Stored Procedures
۱۴۲.....	۳-۱-۱۰-۲ Stored procedure یک تغییر یک
۱۴۳.....	۳-۱-۱۰-۳ Stored procedure یک حذف
۱۴۴.....	۳-۱-۱۰-۱ استفاده از پارامتر در دستور Procedure
۱۵۱.....	۳-۱-۱۱ بررسی Trigger در SQL Server
۱۵۳.....	۳-۱-۱۲ توابع در SQL Server 2019
۱۵۷.....	۳-۱-۱۲-۱ توابع اسکالر در SQL Server
۱۶۰.....	۳-۱-۱۲-۲ توابع تاریخ یا Date
۱۹۸.....	۳-۱-۱۲-۳ توابع رشته‌ای
۲۱۹.....	۳-۱-۱۲-۴ توابع سیستمی
۲۳۵.....	فصل چهارم
۲۳۵.....	امنیت در SQL Server
۲۳۵.....	۴-۱ روش‌های احراز هویت
۲۳۶.....	۴-۲ ایجاد کاربر و اعطای دسترسی به آن
۲۴۶.....	۴-۳ ایجاد کاربر از طریق Query
۲۴۹.....	۴-۴ رمزگذاری بر روی دیتابیس
۲۵۰.....	۴-۴-۱ کلیدهای متقارن (Symmetric) و نامتقارن (Asymmetric)
۲۵۱.....	۴-۴-۲ هش کردن (Hashing)

۲۵۲.....	۴-۳ رمزگذاری بر روی ستون‌های جداول در SQL
۲۵۴.....	۴-۳-۱ ایجاد Master Key برای رمزگذاری رو ستون
۲۵۵.....	۴-۳-۲ ایجاد Certificate در SQL
۲۵۶.....	۴-۳-۳ ایجاد کلید متقارن
۲۵۷.....	۴-۳-۴ رمزگذاری داده
۲۶۳.....	فصل پنجم
۲۶۳.....	پشتیبان‌گیری و بازیابی
۲۶۳.....	۵-۱ پشتیبان‌گیری از طریق نرم‌افزار SQL
۲۸۵.....	۵-۱-۱ پشتیبان‌گیری به صورت دستی در SQL
۲۸۷.....	۵-۱-۲ نحوه‌ی بازگرداندن فایل‌های Backup
۲۸۹.....	۵-۱-۳ بازگرداندن دیتابیس حذف شده
۲۹۱.....	۵-۱-۴ Attach و Detach کردن دیتابیس
۲۹۳.....	۵-۲ پشتیبان‌گیری از طریق نرم‌افزار Veeam Backup
۲۹۳.....	۵-۲-۱ نصب نرم‌افزار Veeam Backup and Replication
۲۹۸.....	۵-۲-۲ اضافه کردن سرورها برای پشتیبان‌گیری
۳۰۵.....	۵-۲-۳ اضافه کردن سرور Backup
۳۱۴.....	۵-۲-۴ پشتیبان‌گیری از ماشین مجازی
۳۲۰.....	۵-۲-۵ استفاده از Veeam Agent
۳۳۱.....	فصل ششم
۳۳۱.....	SQL Replication
۳۳۱.....	۸-۱ بررسی سرویس Replication
۳۴۷.....	۸-۲ نصب و راه‌اندازی سرویس Replication
۳۷۱.....	فصل هفتم
۳۷۱.....	SQL Reporting Service

۳۸۶.....	نصب و راه‌اندازی Power BI
۴۰۳.....	فصل هشتم
۴۰۳.....	SQL Failover Clustering
۴۳۷.....	فصل نهم
۴۳۷.....	هوش تجاری – Business Intelligence
۴۳۷.....	۹-۱ هوش تجاری چیست
۴۳۷.....	۹-۲ چگونه کار را آغاز کنیم
۴۳۸.....	۹-۳ مقدمه‌ای بر SSIS
۴۳۸.....	۹-۴ نصب و راه‌اندازی سرویس SSIS
۴۴۰.....	۹-۵ ایجاد پروژه SSIS
۴۴۲.....	۹-۵-۱ بررسی Package در SSIS
۴۴۳.....	۹-۵-۲ بررسی Connection Manager
۴۴۴.....	۹-۵-۳ استفاده از FTP در SSIS
۴۴۸.....	۹-۵-۴ کار با Integrity, Shrink, Backup, Email در SSIS
۴۵۵.....	۹-۵-۵ گرفتن خروجی تصادفی از جداول با SSIS
۴۶۳.....	منابع

مقدمه

با تشکر از انتخاب این کتاب به‌عنوان مرجع کامل آموزش **SQL Server 2019** که به‌عنوان کامل‌ترین کتاب در این زمینه ارائه شده است، در این کتاب سعی کردیم همه‌ی جنبه‌های **SQL Server** را بررسی و آموزش دهیم و امیدواریم با موفقیت همراه بوده باشد، البته برای این که کتاب به صورت کامل ارائه شود از منابع مختلف در کتاب استفاده شده است که لیست آن در آخر کتاب ذکر شده است. مباحث کلی که در این کتاب بررسی شده است شامل:

- ۱- بررسی بنیادی نرم‌افزار **SQL Server**
- ۲- پیاده‌سازی **SQL Server 2019**
- ۳- کار با پایگاه‌داده و دستورات آن
- ۴- امنیت در **SQL Server 2019**
- ۵- پشتیبانگیری و بازیابی اطلاعات
- ۶- بررسی **SQL Replication**
- ۷- پیاده‌سازی سرویس **SQL Failover Clustering**
- ۸- هوش تجاری – **Business Intelligence**

فصل اول

مباحث بنیادی پایگاه داده

تقریباً بیشتر برنامه‌های کاربردی، امروزه به روش پایگاه داده طراحی می‌شوند. پایگاه داده امکان ذخیره‌سازی متمرکز داده با حداقل افزونگی و استفاده‌ی اشتراکی توسط کاربران مختلف را می‌دهد.

۱-۱ بررسی اولیه

۱-۱-۱ داده چیست



شکل ۱-۱

داده (data) دارای تعاریف مختلفی است، به طور کلی داده‌ها کلمات و ارزش‌های واقعی هستند که از طریق مشاهده و تحقیق بدست می‌آیند، به عبارت دیگر داده نمودی از وقایع، معلومات، رخدادها، پدیده‌ها و مفاهیم هستند. در محاسبات، داده به اطلاعاتی گفته می‌شود که به شکلی مناسب برای انتقال و پردازش ترجمه شود. در کامپیوتر و رسانه‌های ارتباطاتی امروزی داده به شکل باینری تبدیل می‌شود. بنابراین داده یک نمایش باینری از یک موجودیت منطقی ذخیره شده در حافظه کامپیوتر است.

ریشه کلمه‌ی داده از عبارت لاتین datum گرفته شده که به معنی اطلاع است و data فرم جمع آن است. اما datum به ندرت استفاده می‌شود و اکثراً ترجیح می‌دهند data را به صورت یک کلمه مفرد استفاده کنند.

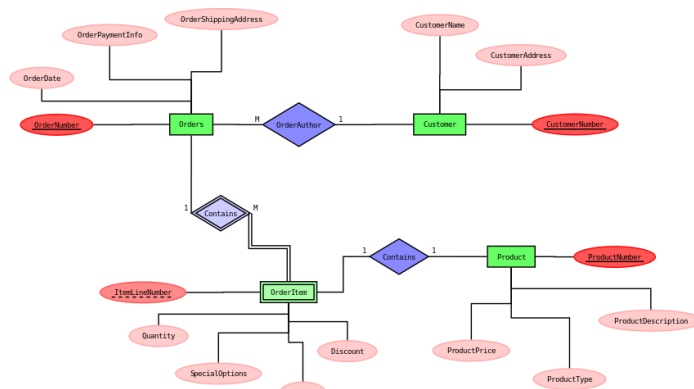
۱-۱-۲ اطلاعات چیست



شکل ۱-۲

اطلاعات (information) مفهومی است که برای گیرنده آن قابل درک بوده و با دانستن آن می‌تواند برای حل یک مسئله تصمیم‌گیری یا ارزیابی کند. وقتی اطلاعات وارد کامپیوتر شود، ذخیره می‌شود و به داده تبدیل می‌شود. بعد از پردازش، داده خروجی مجدداً به عنوان اطلاعات دریافت می‌شود. هنگامی که اطلاعات در یک بسته خاص قرار می‌گیرند و برای درک موضوعی یا انجام کاری استفاده می‌شود به دانش (knowledge) تبدیل می‌شود.

۱-۱-۳ موجودیت چیست



شکل ۱-۳

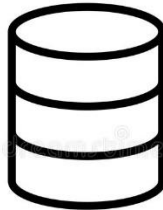
موجودیت (entity) مفهوم کلی پدیده، شیء یا فردی است که در مورد آن می‌خواهیم اطلاع داشته باشیم و در کامپیوتر ذخیره کنیم.

هر نوع موجودیت دارای مجموعه‌ای از صفات خاصه (attribute) است که ویژگی جداکننده یک نوع موجودیت از نوع دیگر محسوب می‌شود.

اگر در نظر داریم یک سیستم پایگاه داده برای یک دبیرستان پیاده‌سازی کنیم مواردی چون دانشجویان، استادان، دروس، کلاس‌ها و غیره جزء موجودیت‌های سیستم به شمار می‌روند.

موجودیت دانشجو در سیستم دانشگاه می تواند دارای صفات خاصه: نام، نام خانوادگی، سن، سال تولد، رشته تحصیلی، سال ورود و ... باشد و یا موجودیت درس صفات خاصه: کد درس، نام درس و تعداد واحد باشد.

۴-۱-۱ پایگاه داده چیست



DATABASE

شکل ۴-۱

یک بانک اطلاعاتی یا پایگاه داده (database) مجموعه‌ای سازمان یافته و بدون افزونگی از داده‌های مرتبط به هم است که می‌تواند توسط سیستم‌های کاربردی مختلف به اشتراک گذاشته شود و به راحتی دسترسی، مدیریت و بهنگام شود. وقتی داده به صورت پایگاه داده سازماندهی می‌شود، کاربر و برنامه‌نویس نیازی به دانستن جزئیات ذخیره‌سازی داده ندارند. علاوه بر این داده می‌تواند بدون تأثیر روی اجزای دیگر سیستم تغییر کند. برای مثال از اعداد حقیقی به صحیح، از یک ساختار فایل به دیگری یا از دستگاه ذخیره‌سازی نوری به مغناطیسی تغییر کند.

۴-۱-۱-۱ ویژگی‌های داده در پایگاه داده

ویژگی‌هایی که داده در پایگاه داده باید داشته باشند در لیست زیر آمده است:

۱. اشتراکی شدن (Shared)

داده در پایگاه داده بین چندین کاربر و برنامه کاربردی به اشتراک گذاشته می‌شود.

۲. ماندگاری (Persistence)

وقتی داده در پایگاه داده ذخیره شد پایدار است و از بین نمی‌رود، مگر آنکه توسط سیستم پایگاه داده تغییر کند.

۳. امنیت (Security)

داده در پایگاه داده از فاش شدن، تغییر و تخریب بدون مجوز محافظت می‌شود. مدیر سیستم توسط سطوح دسترسی و قیدهای امنیتی نحوه دستیابی به داده را مشخص می‌کند و اطمینان می‌دهد که دستیابی از طریق مناسب انجام می‌شود.

۴. اعتبار (Validity)

یا جامعیت (Integrity) و یا صحت (Correctness) به درستی داده در پایگاه داده نسبت به موجودیت دنیای واقعی معتبر اشاره دارد. مثلاً موجودی بانک نباید منفی باشد.

۵. سازگاری (Consistency)

داده در پایگاه داده با مقدار واقعی داده در دنیای خارج سازگار است. وقتی یک فقره اطلاع در بیش از یک نقطه ذخیره شود و لازم باشد به هنگام شود، اگر به هنگام سازی در همه نقاط انجام نشود ناسازگاری ایجاد می شود.

۶. کاهش افزونگی (Non redundancy)

داده در پایگاه داده دارای حداقل افزونگی است. افزونگی به این معناست که هیچ دو فقره داده در بانک معرف یک موجودیت در دنیای خارج نباشد.

۷. استقلال (independence)

تغییر در نمایش فیزیکی، تکنیک های دستیابی و سازماندهی داده تأثیری روی برنامه های کاربردی ندارد.

۲-۱-۴-۱ روش های ذخیره داده

دو روش کلی برای ذخیره و بازیابی خودکار داده ها وجود دارد: سیستم فایل ساده و سیستم پایگاه داده.

سیستم فایل (file system)

در این روش، داده ها در فایل های مجزا قرار گرفته و سیستم های جداگانه ای به نام سیستم پردازش فایل برای استفاده از فایل های داده ای طراحی می شوند. در این سیستم ها هر برنامه ای کاربردی تنها به فایل داده ای مربوط به خود می تواند مراجعه کند.

اشکالات چنین طراحی در ذخیره داده به طور خلاصه عبارت اند:

۱. افزونگی و ناسازگاری داده به دلیل چندین فرمت فایل و تکرار اطلاعات در فایل های مختلف.
۲. مشکل در دستیابی داده و نیاز به نوشتن برنامه جدیدی برای انجام هر کار.
۳. قیدهای جامعیت به جای اینکه صریحاً بیان شوند در کد برنامه از نظر پنهان می شوند. اضافه کردن قیدهای جدید یا تغییر قیدهای موجود به سختی صورت می گیرد.
۴. ایجاد ناسازگاری به دلیل وجود چندین کپی از فقره های داده.
۵. مشکلات امنیتی به دلیل دسترسی همروند و بدون کنترل توسط چند کاربر

۳-۴-۱-۱ سیستم پایگاه داده (database system)

در این روش کلیه داده‌ها به صورت مجتمع در پایگاه داده ذخیره می‌شود، ولی هر کاربر دید خاص خود را نسبت به داده‌ها دارد. کاربران مختلف می‌توانند به طور مشترک با پایگاه داده کار کنند. به دلیل تجمع داده افزونگی به حداقل ممکن کاهش می‌یابد.

نرم‌افزاری به نام سیستم مدیریت پایگاه داده (DBMS) به عنوان واسطه بین برنامه‌های کاربردی و پایگاه داده ایفای نقش می‌کند لذا امنیت داده‌ها در این روش بیشتر است.

چند نمونه از کاربردهای سیستم پایگاه داده موارد زیر هستند:

- انجام کلیه تراکنش‌های بانکداری.
- رزرواسیون و زمان‌بندی خطوط هوایی.
- ثبت نام دانشجویان، واحدگیری و ثبت نمرات در مراکز آموزشی.
- ثبت اطلاعات مشتریان، محصولات و فاکتورهای خرید و فروش.
- پیگیری سفارشات و پیشنهادهای در فروش online.
- ثبت رکوردهای کارمندان و محاسبات حقوق، کسورات مالیاتی در سازمان‌ها.

۴-۴-۱-۱ عناصر اصلی سیستم پایگاه داده

اجزاء اصلی سیستم بانک اطلاعاتی عبارت‌اند از:

۱. داده‌ها

شامل داده‌هایی درباره موجودیت‌های مختلف محیط و ارتباط بین موجودیت‌ها.

۲. سخت‌افزار

شامل عناصر پردازشی، رسانه‌های ذخیره‌سازی داده، دستگاه‌های جانبی، سخت‌افزارهای ارتباطی و غیره.

۳. نرم‌افزار

شامل سیستم‌عامل و نرم‌افزارهای ارتباطی شبکه، نرم‌افزار سیستم مدیریت پایگاه داده و برنامه‌های کاربردی.

۴. رویه‌های عملیاتی

شامل کلیه عملیاتی که روی پایگاه داده انجام می‌شود، نظیر تهیه پشتیبان، آمارگیری و ...

۵. کاربر

شامل کاربران یا کسانی که به نحوی با سیستم در ارتباط هستند نظیر مدیر پایگاه داده (DBA)، طراحان پایگاه داده

(DBD)، برنامه‌نویسان پایگاه داده (DBP) و کاربران نهایی (end users).

۵-۴-۱- مزایا و معایب سیستم‌های پایگاه داده

مزایای مهم سیستم پایگاه داده

- ۱- تجمع، وحدت ذخیره‌سازی و کنترل متمرکز داده‌ها.
کاهش افزونگی. تجمع داده و وحدت ذخیره‌سازی باعث کاهش افزونگی می‌شود. مثلاً آدرس‌های مختلف برای یک مشتری در قسمت‌های مختلف اداره ثبت نمی‌شود.
- ۲- به اشتراک گذاشتن داده‌ها
چند کاربر می‌توانند هم‌زمان به یک پایگاه داده دسترسی داشته باشند. برنامه‌های کاربردی موجود قادر به اشتراک گذاردن داده‌ها در پایگاه داده بوده و برنامه‌های کاربردی جدید نیز می‌توانند از این داده‌ها استفاده کنند.
- ۳- پرهیز از ناسازگاری
با کاهش افزونگی، کنترل متمرکز و جامعیت، سازگاری و یکپارچگی داده‌ها تضمین می‌شود.
- ۴- اعمال محدودیت‌های امنیتی
سیستم‌های امنیتی در پایگاه داده امکان اعمال کنترل‌های مختلف را برای هر نوع دسترسی (بازیابی، اصلاح، حذف و غیره) بر روی پایگاه داده فراهم می‌کند.
- ۵- صحت بیشتر داده و استقلال از برنامه‌های کاربردی.
- ۶- راحتی پیاده‌سازی برنامه‌های کاربردی جدید.

معایب سیستم پایگاه داده

- ۱- طراحی سیستم‌های پایگاه داده پیچیده‌تر، دشوارتر و زمان برتر است.
- ۲- هزینه قابل توجه، صرف سخت‌افزار و نصب نرم‌افزار می‌شود.
- ۳- آسیب دیدن پایگاه داده، روی کلیه برنامه‌های کاربردی تأثیر می‌گذارد.
- ۴- هزینه زیاد برای تبدیل از سیستم فایلی به سیستم پایگاه داده نیاز است.
- ۵- نیازمند تعلیم اولیه برنامه‌نویسان و کاربران و استخدام کارمندان خاص پایگاه داده است.
- ۶- نیاز به تهیه چندین کپی پشتیبان از پایگاه داده می‌باشد.
- ۷- خطاهای برنامه می‌توانند فاجعه برانگیز باشند.
- ۸- زمان اجرای هر برنامه طولانی‌تر می‌شود.
- ۹- بسیار وابسته به عملیات سیستم مدیریت پایگاه داده است.

۱-۱-۵ سیستم مدیریت پایگاه داده



شکل ۱-۵

سیستم مدیریت پایگاه داده یا به طور خلاصه DBMS (DataBase Management System) مهم‌ترین نرم‌افزاری در سیستم پایگاه داده است که به عنوان رابط بین پایگاه داده و کاربر و برنامه‌های کاربردی عمل می‌نماید. کلیه فایل‌های پایگاه داده فقط در اختیار این نرم‌افزار قرار گرفته و دستیابی به آنها تنها از طریق DBMS امکان‌پذیر است.

DBMS سرویس‌هایی جهت دسترسی داده در پایگاه داده فراهم می‌کند به نحوی که از کلیه خواص داده محافظت شود.

۱-۱-۵-۱ وظایف سیستم مدیریت پایگاه داده

وظایف DBMS در سیستم‌های مختلف تا حدودی متفاوت بوده و بستگی به نوع کاربران آن دارد. اما به‌طور کلی این وظایف عبارت‌اند از:

- ۱- امکان تعریف پایگاه داده.
- ۲- امکان ایجاد پایگاه داده.
- ۳- امکان دست‌کاری در داده‌ها.
- ۴- بازیابی پایگاه داده.
- ۵- به‌هنگام‌سازی پایگاه داده (عملیات درج، حذف و جایگزینی).
- ۶- تأمین تسهیلاتی برای کاربر به منظور توسعه سیستم.
- ۷- امکان سازماندهی مجدد.
- ۸- کنترل امنیت و جامعیت داده‌ها.
- ۹- ایجاد دیکشنری داده‌ها.
- ۱۰- امکان کنترل کارایی.

۲-۱-۵-۱ تراکنش



شکل ۶-۱

تراکنش (transaction) یک برنامه فعال است که دنباله‌ای از دستورات را شامل می‌شود و به طور خاص بعضی عملیات آن روی پایگاه داده است.

سه عمل تراکنشی خاص وجود دارد:

Start که نشان می‌دهد یک تراکنش در حال شروع شدن است.

Commit که دلالت بر اتمام عادی تراکنش دارد.

Abort که بیان‌کننده پایان یافتن تراکنش به دلیل لغو آن است و کلیه اثرات تراکنش لغو شده باید rollback یا بی‌اثر شود. وقتی تراکنش commit می‌شود تأثیرش روی پایگاه داده باید دائمی شود.

هر تراکنش باید پایگاه داده را از یک حالت سازگار به حالت سازگار بعدی ببرد. تراکنش باید دارای خواص ACID باشد تا پایگاه داده را در حالت سازگار باقی نگهدارند. خواص ACID حروف اول چهار خاصیت زیر است:

۱. اتمی بودن (Atomicity)

تراکنش‌ها اتمیک هستند یا اصلاً شروع نمی‌شوند یا وقتی آغاز شدند حتماً به پایان می‌رسند. یا تمام عملیات انجام می‌شود یا هیچ‌کدام.

نگه‌داشتن خاصیت اتمیک به عهده‌ی کنترل هم‌روندی و ترمیم است.

۲. سازگاری (Consistency)

یک تراکنش یا پایگاه داده را به حالت سازگار جدیدی می‌برد یا اگر شکستی رخ داد کلیه داده‌ها به حالت قبل از شروع تراکنش برمی‌گردند.

۳. ایزوله بودن (Isolation)

تراکنشی که در حال اجراست و هنوز به پایان نرسیده تأثیرش از بقیه مخفی است مگر اینکه commit شده باشند. اجرای هم‌روند تراکنش‌ها باید به صورتی باشد که انگار پشت سرهم اجرا شده‌اند. حفظ این خاصیت برعهده کنترل هم‌روندی است.

۴. ماندگاری (Durability)

از وقتی تراکنشی commit شد تأثیرش دائمی است؛ حتی اگر سیستم خراب شود، داده در حالت درست خود باقی می ماند.

۳-۵-۱-۱ اجزای سیستم مدیریت پایگاه داده

وظایف DBMS توسط تعدادی مؤلفه نرم افزاری انجام می شود. هرکدام از این مؤلفه ها ممکن است مرکب از چند واحد کوچک تر باشند. تعدادی از سرویس های که توسط مؤلفه های DBMS داده می شوند در زیر لیست شده است:

پردازش تراکنش (Transaction Processing)

پردازش تراکنش، عملیاتی که از منابع مختلف می رسد را روی پایگاه داده اجرا می کند به نحوی که خواص مطلوب تراکنش خدشه دار نشود. سرویس های کنترل همروندی و ترمیم به این مؤلفه برای برقراری خواص ACID کمک می کنند. به این ترتیب اجرای همروند تراکنش ها و سازگاری پایگاه داده حتی در صورت وقوع شکستی در سیستم تضمین می شود.

کنترل همروندی (Concurrency Control)

مدیریت اجرای همروند تراکنش ها روی پایگاه داده در حین برقراری سازگاری را به عهده دارد.

ترمیم (Recovery)

ترمیم تضمین می کند که اگر اجرای تراکنش با عدم موفقیت یا لغو روبرو شد، تأثیر نامطلوبی بر روی پایگاه داده یا تراکنش های دیگر نگذارد و حالت پایگاه داده را همیشه سازگار نگه دارد.

مدیریت ثبت (Log Management)

هر اتفاقی در سیستم در یک فایل ذخیره می شود و توسط مدیریت ترمیم برای حفظ صحت و اعتبار پایگاه داده هنگام خرابی یا لغو سیستم استفاده می شود.

واسطه زبانی (Language Interface)

دستوراتی را برای تعریف داده، کارکردن با آن اختیار کاربران و برنامه های کاربردی قرار می دهد.

تحمل پذیری خطا (Fault Tolerancy)

توانایی ارائه سرویس های قابل اطمینان توسط DBMS حتی در صورت بروز نقص را تحمل پذیری خطا می گویند. انواع خطاهایی که ممکن است پیش بیاید عبارت اند از:

- خطای منطقی: تراکنش موفق نمی شود مثلاً به دلیل ورودی بد، سرریزی.

- خطای سیستمی: تراکش موفق نمی‌شود مثلاً به دلیل بن‌بست.
- لغو سیستم: قطع برق، پاک شدن حافظه اصلی، پر شدن دیسک.
- ناتوانی دیسک: خرابی هد، خرابکاری عمدی، آتش‌سوزی.

کاتالوگ داده (Data Catalog)

یک دیکشنری داده یک پایگاه‌داده سیستمی شامل اطلاعاتی درباره داده، ارتباطات و قیدها در پایگاه اصلی است. گاهی به آن متا داده هم گفته می‌شود.

امنیت (Security)

امنیت به محافظت داده در مقابل افشا شدن، تغییر و خرابی اشاره دارد. هر کاربر و برنامه کاربردی امتیاز ویژه‌ای برای دسترسی به داده دارند. کاربران ممکن است دیدگاه‌های مختلفی نسبت به داده‌های پایگاه‌داده با توجه امتیازات ویژه خود داشته باشند. سیستم امنیتی همچنین، توسط رویه‌های شناسایی و مجوز، دسترسی به پایگاه‌داده را محدود می‌کند.

مدیریت ذخیره‌سازی (Storage Management)

DBMS مکانیسم‌های خاصی برای ذخیره دائمی داده و دسترسی به منبع فیزیکی و بازیابی داده دارد. مدیر ذخیره‌سازی بین داده ذخیره شده در پایگاه‌داده و برنامه کاربردی و پرس‌وجوهای ارسال شده به سیستم واسطه می‌شود.

مدیریت قفل (Lock Management)

هنگام استفاده اشتراکی از داده انواع مختلفی از قفل روی داده گذاشته می‌شود (مثل Read Lock و Write Lock).

مدیریت بن‌بست (Deadlock Management)

بن‌بست وقتی اتفاق می‌افتد که تراکش‌ها برای به‌دست‌آوردن منابع در یک دایره بسته قرار گیرند یعنی هر یک منبعی در اختیار دارد که مورد تقاضای دیگری است و درخواست منبعی را می‌کند که در اختیار تراکش منتظر منبع است. در پایگاه‌داده منابع رکوردها هستند. مدیریت منبع مسئول رفع این مشکل هستند.

۱-۱-۶ انواع سیستم‌های مدیریت پایگاه‌داده

انواع مختلفی از سیستم‌های پایگاه‌داده وجود دارند که هرکدام به‌منظور خاصی طراحی و پیاده‌سازی شده‌اند. دسته‌بندی سیستم‌های پایگاه‌داده به‌صورت زیر انجام گرفته است:

۱-۱-۶-۱ سیستم مدیریت پایگاه‌داده توزیع شده

سیستم‌های توزیع شده (Distributed DataBase Management System) بر توزیع داده و همچنین همبستگی فعالیت‌ها و کنترل روی اجزای توزیع شده سیستم دلالت دارند. اکثر سیستم‌های توزیع شده برای تقسیم کردن بار کاری یا برای انتقال عملکردهای پردازش داده به نزدیکی محل انجام این وظایف است. در هر دو حالت هدف نامحسوس بودن توزیع شدگی از دید کاربر است.

۱-۱-۶-۲ سیستم مدیریت پایگاه داده بلادرنگ

سیستم‌های بلادرنگ (Real-Time DataBase Management System) سیستم‌های سریع با سرعت پاسخگویی بالا هستند که زمان انجام کلیه عملیات نقش مهمی در آنها دارد. سیستم بلادرنگ در تعامل با دنیای واقعی پاسخ قابل پیش‌بینی را در قاب زمان می‌دهد. ورودی، پردازش و پاسخ‌ها همگی از قبل تعریف شده هستند و حد زمانی مشخصی دارند و به نحوی بهینه می‌شوند که هر حالت ورودی یک حالت خروجی قابل پیش‌بینی دارد که همیشه در یک‌زمان و به یک روش اتفاق می‌افتد.

۱-۱-۶-۳ سیستم مدیریت پایگاه داده تحمل‌پذیر خطا

سیستم تحمل‌پذیر خطا (Fault Tolerance DataBase Management System) سرویس‌هایی را دارد که با ناتوانی‌های اجزای سخت‌افزاری و نرم‌افزاری برخورد می‌کند. برای رسیدن به این منظور باید کلیه نقاطی که احتمال نقصی در آنها وجود دارد از قبل بررسی شده، ابزارهایی برای تشخیص، اصلاح و یا ترمیم آنها به نحوی طراحی شود که کمترین تأثیر را روی برنامه‌های کاربردی بگذارند. مکانیسم‌های RAID، Shadow Memory و کپی از جمله روش‌هایی هستند که استفاده می‌شوند.

۱-۱-۶-۴ سیستم مدیریت پایگاه داده امن

در یک سیستم پایگاه داده مطمئن (Secure DataBase Management System) کلیه اعمالی که کاربران و برنامه‌های کاربردی اجازه دارند انجام دهند همچنین زمان و مقدار انجام آنها کنترل می‌شوند. به‌عنوان مثال یک سیستم پرسنلی ممکن است در نظر داشته باشد به کلیه کاربران اجازه دستیابی به سابقه پرسنلی خودشان و استخراج اطلاعات شغلی‌شان را بدهد اما دسترسی به سابقه کارمندان دیگر یا حتی برخی اطلاعات مربوط به خودشان امکان‌پذیر نباشد. برای دادن چنین سرویسی سیستم پایگاه داده باید قابلیت تعریف حقوق دسترسی و رسیدگی به آنها را در قبال کاربرانی که به داده دسترسی دارند داشته باشد.

۱-۱-۶-۵ سیستم مدیریت پایگاه داده ناهمگون

سیستم ناهمگون (Hetrogenouse DataBase Management System) از DBMS های مختلف تشکیل شده است. برای مثال شعب یک شرکت هر کدام منحصراً نیازهای پردازشی خود را با سخت افزار و نرم افزار جداگانه برطرف می کنند. اگر نیاز باشد این سیستم ها با هم فعل و انفعال داشته باشند و از طریق شبکه به هم پیوند داده شوند یک HTDBMS ایجاد می شود تا پایگاه داده های مختلف با هم ارتباط برقرار کند.

۱-۱-۶-۶ سیستم مدیریت پایگاه داده چندرسانه ای

سیستم های محاسباتی چندرسانه ای (Multimedia DataBase Management System) انواع متنوعی از منابع داده ای گرافیکی، تصاویر ویدئویی، صوت و متن را استفاده و یا با هم ترکیب می کنند. این منابع داده ای پیچیده باید برای سیستم محاسباتی به سهولت قابل دسترس باشند. برای استفاده در برنامه های کاربردی interactive چنین سیستم هایی از ترکیب الزامات پایگاه داده های بلادرنگ با سیستم های گرافیکی تعاملی استفاده می کنند تا ارائه اطلاعات سنکرون شده و بلادرنگ حاصل شود.

۱-۱-۶-۷ سیستم مدیریت پایگاه داده متحد

Federated DataBase Management System، نسل جدید سیستم های مدیریت پایگاه داده سعی دارند اطلاعات جمع آوری شده از سنسورها را مستقیماً ذخیره کنند. این سیستم ها از پایگاه های دانش نیز حمایت می کنند.

۱-۱-۷ کاربران پایگاه داده

کاربران یک سیستم پایگاه داده توسط روش هایی که با سیستم تعامل می کنند از هم تفکیک می شوند.

۱-۱-۷-۱ تحلیل گران سیستم

تحلیل گران سیستم (system analysts) با گروه کاربران پایگاه داده به منظور درک نیازهای اطلاعاتی و پردازشی آنها ارتباط دارند. نیازهای اطلاعاتی و پردازشی هر گروه را مجتمع می کنند و مستندسازی می کنند.

۱-۱-۷-۲ طراحان پایگاه داده

طراحان پایگاه داده (database designers) ساختار مناسبی را برای نمایش اطلاعات مشخص شده توسط تحلیل گر سیستم به طریق نرمال سازی شده به منظور تضمین جامعیت و سازگاری داده انتخاب می کنند و با استفاده از DDL داده های پایگاه داده را تعریف می کنند.

۳-۷-۱-۱ پیاده سازان برنامه‌های کاربردی

برنامه‌نویسان برنامه‌های کاربردی (Application Developers) برای برآوردن نیازهای کاربران و کار با پایگاه داده برنامه‌هایی را آماده می‌کنند. تست، اشکال‌زدایی و مستندسازی برنامه و پایگاه داده از وظایف برنامه‌نویسان است. برنامه‌نویسان با سیستم توسط احکام DML ارتباط برقرار می‌کنند.

۴-۷-۱-۱ مدیر پایگاه داده

مدیر پایگاه داده (database administrator) یا به طور خلاصه DBA فردی است که مسئول کنترل عملیات کل سیستم پایگاه داده است. DBA کلیه فعالیت‌های سیستم پایگاه داده را هماهنگ می‌کند. این فرد باید درک خوبی از منابع و نیازهای اطلاعاتی کل سازمان داشته باشد و برای حصول اطمینان از اینکه داده مورد نیاز قابل دسترس کاربران قرار می‌گیرد با آنها در ارتباط باشد.

بعضی از وظایف DBA شامل:

- تعریف شماها توسط DDL
- تعریف ساختار ذخیره‌سازی و متدهای دسترسی توسط DDL
- اصلاح شما و سازماندهی فیزیکی
- اعطای مجوز دسترسی پایگاه داده به کاربران
- تعیین قیدهای جامعیت
- عامل ارتباطی کاربران
- نظارت اجرا و واکنش برای تغییر در صورت نیاز
- برقراری دیکشنری داده

۵-۷-۱-۱ کاربران نهایی

کاربران نهایی (End Users) شامل:

- کاربران پارامتری: که توسط برنامه‌های کاربردی نوشته شده با سیستم سروکار دارند. مانند تحویلدار بانک و کارکنان دفتری.
- کاربران ماهر: که نیازهای پیچیده‌تری دارند و با قابلیت‌های DBMS آشنائی کامل دارند. درخواست‌های خود از پایگاه داده را توسط یک زبان پرس‌وجو می‌سازند.
- کاربران نهایی اتفاقی: کسانی که دسترسی گاه‌وبیگاه به پایگاه داده دارند اما ممکن است هر بار نیازهای متفاوتی داشته باشند. از زبان‌های پرس‌وجوی و مرورگرهای حرفه‌ای تر استفاده می‌کنند.

۸-۱-۱ دیکشنری داده

دیکشنری داده‌ها (Data Catalog) یکی از امکاناتی است که در سیستم پایگاه‌داده در اختیار DBA قرار می‌گیرد. دیکشنری داده‌ها که به آن راهنمای سیستم نیز می‌گویند یک متا داده است یعنی اطلاعاتی درباره خود پایگاه‌داده و داده‌های ذخیره شده در آن را نگهداری می‌کند.

دیکشنری داده تعیین می‌کند چه داده‌ای موجود است و چه معنی دارد، داده چگونه ذخیره می‌شود و در کجا قرار دارد، مالک آن چه کسی است و چه کسانی اجازه دسترسی به داده را دارند، تاریخچه و آمار استفاده از داده را در بردارد.

۹-۱-۱ پایگاه‌داده XML

یک پایگاه‌داده XML سیستم نرم‌افزاری است که اجازه می‌دهد داده در فرمت XML وارد، پردازش و ارسال شود.

دودسته اصلی پایگاه‌داده XML وجود دارد:

XML-enabled - پایگاه داده‌ای که مستند XML را به‌عنوان ورودی گرفته به یک پایگاه‌داده دیگر نظیر رابطه‌ای تبدیل می‌کند و پس از انجام عملیات آنها را مجدداً به XML بر می‌گرداند.

Native XML (NXD) - مدل داخلی چنین پایگاه‌داده‌ای بر پایه XML است و مستندات XML را به‌عنوان منبع ذخیره‌سازی مستقیماً استفاده می‌کند.

دلیل استفاده XML در پایگاه‌داده شفافیت داده است. داده از پایگاه‌داده استخراج می‌شود و در مستندات XML قرار می‌گیرد و برعکس. به این صورت هزینه ذخیره داده در فرمت XML هم کمتر می‌شود.

فصل دوم

پیاده‌سازی SQL Server 2019

۲-۱ نیازمندی‌های SQL Server 2019

برای اینکه سرور SQL خود را راه‌اندازی کنیم، نیاز داریم بدانیم که این سرور چقدر از منابع سخت‌افزاری ما را قرار است مصرف کند، برای همین به جدول ۲-۱ توجه کنید.

جدول ۲-۱ نیازمندی سخت‌افزاری

نیازمندی	سخت‌افزار
نرم‌افزار SQL برای نصب اولیه نیازمند ۶ گیگابایت فضا در هارد دیسک است، اما برای یک محیط عملیاتی باید مشخص شود آن سازمان نیازمند چقدر از فضای هارد دیسک است؛ مثلاً برای محیطی که دارای ۱۰۰ کارمند است باید حداقل ۲۰۰ گیگ فضا برای آن در نظر گرفت. البته همه‌ی اینها مربوط به کاری است که انجام می‌دهید، توجه داشته باشید حتماً از RAID بندی در سرور خود برای هارد دیسک استفاده کنید تا در صورت خرابی هاردها اطلاعات شما از دست نرود.	هارد دیسک
سرور SQL برای اجرا نیازمند رزولیشن Super-VGA (800x600) است.	مانیتور
برای آپدیت نرم‌افزار SQL و استفاده از بخش‌های دیگر نیازمند اینترنت هستیم.	اینترنت
حداقل نیازمندی یک گیگابایت است، برای عملکرد بهتر باید حداقل ۴ گیگابایت رم برای این سرور در نظر بگیرید، البته در محیط عملیاتی واقعی امروزه، حداقل ۱۰ گیگابایت یک انتخاب ایده‌آل است.	رم
حداقل پردازنده X64 با سرعت 1.4 GHz است، اما برای عملکرد بهتر باید از پردازنده ۲ گیگاهرتز به بالا استفاده شود.	سرعت پردازنده
پردازنده‌های امروزی از نوع Intel و AMD به خوبی پاسخگوی نیاز این نرم‌افزار هستند.	نوع پردازنده

نکته:

نصب SQL Server فقط در پردازنده‌های x64 پشتیبانی می‌شود و در پردازنده‌های x86 پشتیبانی نمی‌شود که باید به این نکته توجه کنید.

جدول ۲-۲ نشان می‌دهد که کدام نسخه‌های SQL Server 2019 با کدام نسخه‌های Windows سازگار است:

جدول ۲-۲ سیستم‌عامل

پشتیبانی سیستم‌عامل					
Express	Web	Standard	Developer	Enterprise	نسخه‌های مختلف SQL
Yes	Yes	Yes	Yes	Yes	Windows Server 2019 Datacenter
Yes	Yes	Yes	Yes	Yes	Windows Server 2019 Standard
Yes	Yes	Yes	Yes	Yes	Windows Server 2019 Essentials
Yes	Yes	Yes	Yes	Yes	Windows Server 2016 Datacenter
Yes	Yes	Yes	Yes	Yes	Windows Server 2016 Standard
Yes	Yes	Yes	Yes	Yes	Windows Server 2016 Essentials
Yes	No	Yes	Yes	No	Windows 10 IoT Enterprise
Yes	No	Yes	Yes	No	Windows 10 Enterprise
Yes	No	Yes	Yes	No	Windows 10 Professional
					Windows 10 Home

از نیازمندی‌های دیگر SQL Server می‌توان به NET Framework اشاره کرد که باید نسخه‌ی مورد نظر آن را که در خود ویندوز ارائه می‌شود

نصب کنید، البته این نرم‌افزار به صورت پیش فرض بر روی سرور نصب خواهد شد.

توجه داشته باشید در هنگام نصب SQL Server اجزای زیر بر روی سرور نصب خواهد شد:

- SQL Server Native Client
- SQL Server Setup support files

اگر بخواهید SQL را به همراه اجزای کامل آن بر روی سرور نصب کنید، هرکدام از اجزای طبق جدول ۲-۳ نیازمند

فضای مورد نیاز هستند.

توجه داشته باشید که این مورد را در موقع نصب SQL انتخاب خواهیم کرد.

جدول ۲-۳ اجزای SQL

ویژگی مورد نظر	مقدار فضای مورد نیاز
Database Engine and data files, Replication, Full-Text Search, and Data Quality Services	1480 MB
Database Engine (as above) with R Services (In-Database)	2744 MB
Database Engine (as above) with PolyBase Query Service for External Data	4194 MB
Analysis Services and data files	698 MB
Reporting Services	967 MB
Microsoft R Server (Standalone)	280 MB
Reporting Services - SharePoint	1203 MB
Reporting Services Add-in for SharePoint Products	325 MB
Data Quality Client	121 MB
Client Tools Connectivity	328 MB
Integration Services	306 MB
Client Components (other than SQL Server Books Online components and Integration Services tools)	445 MB
Master Data Services	280 MB
SQL Server Books Online Components to view and manage help content*	27 MB
All Features	8030 MB

نکته:

SQL Server از دیسک با سکتورهای ۵۱۲ بایت تا ۴ کیلوبایت پشتیبانی می‌کند و اگر بخواهید از سکتورهای بالای ۴ کیلوبایت استفاده کنید، مطمئناً با خطا روبرو خواهید شد، برای دریافت اطلاعات بیشتر می‌توانید از لینک زیر استفاده کنید.

<https://support.microsoft.com/en-us/topic/hard-disk-drive-sector-size-support-boundaries-in-sql-server-4d5b73fa-7dc4-1d8a-2735-556e6b60d046>

SQL Server برای ذخیره‌ی اطلاعات از منابع ذخیره‌سازی زیر می‌تواند استفاده کند:

- ۱- حافظه داخلی سرور یا همان هارددیسک محلی.
- ۲- محل ذخیره‌سازی به اشتراک گذاشته شده در شبکه.
- ۳- SQL Server failover cluster.
- ۴- Storage Spaces Direct (S2D) (تکنولوژی شبیه به RAID که اطلاعات دیتابیس در چند سرور قابل دسترس است، این ویژگی همان Fault Tolerance است).
- ۵- فضای ذخیره‌سازی SMB (می‌توانید از یک Windows Server به‌عنوان FileServer برای انکار استفاده کنید).

به دلایل امنیتی که مایکروسافت اعلام کرده، بهتر است که SQL Server را بر روی Domain Controller نصب نکنید، چون موارد زیر را در پی خواهد داشت:

- ۱- نمی‌توانید سرویس‌های SQL Server را در یک Domain Controller تحت اکانت local service اجرا کنید که بسیار اذیت‌کننده خواهد بود.
- ۲- پس از نصب SQL Server بر روی سیستم مورد نظر، نمی‌توانید سیستم مورد نظر را از یک عضو دامنه به یک کنترل‌کننده‌ی دامنه تغییر دهید؛ قبل از تغییر سیستم میزبان به یک کنترل‌کننده‌ی دامنه، باید SQL Server را حذف نصب کنید.
- ۳- پس از نصب SQL Server بر روی رایانه، نمی‌توانید رایانه را از یک کنترل‌کننده‌ی دامنه به یک عضو دامنه تغییر دهید. قبل از تغییر رایانه میزبان به عضو دامنه، باید SQL Server را حذف و بعد نصب کنید.
- ۴- SQL Server failover cluster در یک دومین کنترل خواندنی پشتیبانی نمی‌شوند.
- ۵- SQL Server در یک کنترل‌کننده‌ی دامنه فقط خواندنی پشتیبانی نمی‌شود. SQL Server Setup نمی‌تواند گروه‌های امنیتی یا حساب‌های ارائه‌دهنده‌ی خدمات SQL Server را در یک کنترل‌کننده‌ی دامنه‌ی فقط خواندنی ایجاد کند. در این سناریو، نصب ناموفق است.
- ۶- یک نمونه خوشه‌ی شکست‌خورنده‌ی SQL Server در محیطی که فقط یک کنترل‌کننده‌ی دامنه‌ی فقط خواندنی قابل دسترسی است پشتیبانی نمی‌شود.

۲-۲ نصب و راه‌اندازی SQL Server 2019 در ویندوز

قبل از نصب SQL Server 2019 بهتر است یک موضوع بسیار مهم را بررسی کنیم، همان‌طور که می‌دانید دو نوع سیستم داریم:

۱- فیزیکی

۲- مجازی

به طور معمول یک سیستم فیزیکی با رم و هارد و فضای ذخیره‌سازی مناسب را برای نصب SQL Server در نظر می‌گیرید و فکر می‌کنید بهترین عملکرد را ارائه دادید، اما اگر در این بین، اطلاعات شما از دست برود و هارددیسک شما خراب شود، آن وقت چه کاری باید انجام دهید، آیا به نظر شما استفاده از سیستم فیزیکی برای استفاده‌ی مستقیم از SQL Server کار درستی خواهد بود؟

هرچند می‌توانید با روش‌هایی مانند Raid بندی و روش‌های دیگر جلوی از بین رفتن اطلاعات را بگیرید، اما بهترین کار این است که از مجازی‌سازی استفاده کنیم و از روی ماشین‌ی که ایجاد می‌کنیم، پشتیبان تهیه کنیم تا در موقع از دست رفتن سرور در سریع‌ترین زمان ممکن بتوانیم آن را برگردانیم، البته روش‌های پشتیبان‌گیری در SQL یک موضوع مفصل خواهد بود که در فصل مربوط به پشتیبان‌گیری به صورت کامل به آن خواهیم پرداخت.

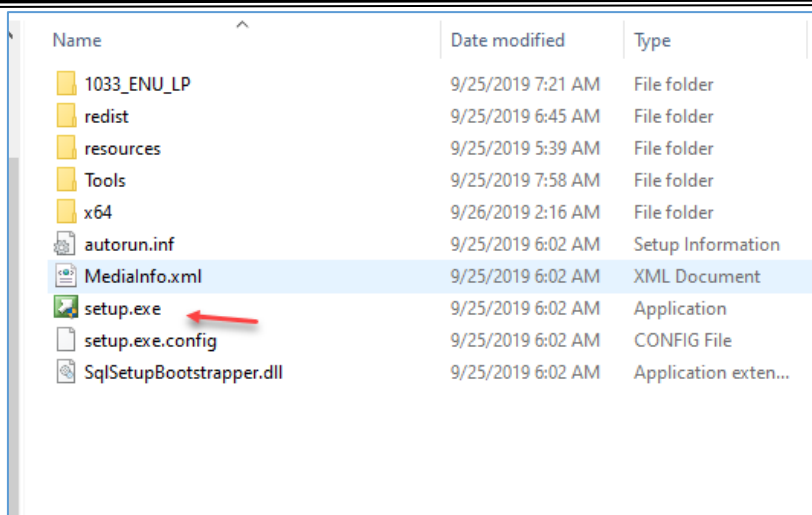
ما برای این کتاب سرور مجازی را انتخاب می‌کنیم، بهترین عملکرد در مجازی‌سازی را شرکت VMware ارائه می‌دهد و شما می‌توانید ماشین مورد نظر خود را توسط نرم‌افزار VMware Workstation و یا با سیستم‌عامل ESXi ایجاد کنید که کار با این نرم‌افزارها را در کتاب VMware Systems به طور کامل توضیح دادیم و می‌توانید نسخه‌ی الکترونیکی را از سایت بنده دریافت کنید.

برای این کتاب ما از یک سرور ESXi استفاده کردیم که روی آن یک ماشین مجازی ایجاد کردیم و سخت‌افزار مناسب را برای آن در نظر گرفتیم، بعد از این کار بر روی آن ویندوز سرور ۲۰۱۹ نصب کردیم تا همه چیز برای نصب نرم‌افزار SQL آماده باشد.

برای اینکه نرم‌افزار SQL Server 2019 را دانلود کنید می‌توانید به صورت مستقیم از سایت مایکروسافت دانلود کنید و یا اینکه آن را از سایت‌های ایرانی دانلود کنید که آدرس آن هم در زیر قرار دارد:

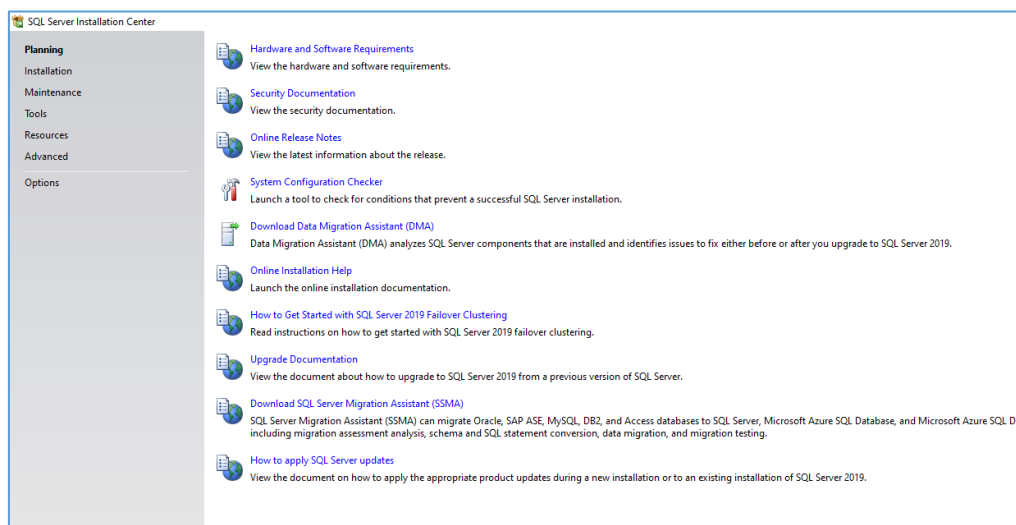
<https://soft98.ir/software/programming/3594-microsoft-sql-server-all-2017-full-1.html>

بعد از دانلود فایل مورد نظر، به‌مانند شکل ۱-۲ بر روی فایل Setup.exe دو بار کلیک کنید.



شکل ۲-۱ فایل Setup

اولین صفحه‌ای که به شما نمایش داده می‌شود، شکل ۲-۲ است که گزینه‌های مختلفی را به شما ارائه می‌دهد، مثلاً در گزینه‌ی اول اطلاعاتی را در مورد سخت‌افزار و نرم‌افزار موردنیاز برای نصب SQL Server توضیح می‌دهد و همچنین در گزینه‌های دیگر می‌توانید موارد دیگر را هم بررسی کنید. برای اینکه متوجه شویم که سیستمی که در حال نصب SQL بر روی آن هستیم آیا مناسب است یا نه باید در شکل ۲-۲ بر روی لینک System Configuration Checker کلیک کنید، تا به‌مانند شکل ۲-۳ همه‌ی گزینه‌ها اوکی باشد.

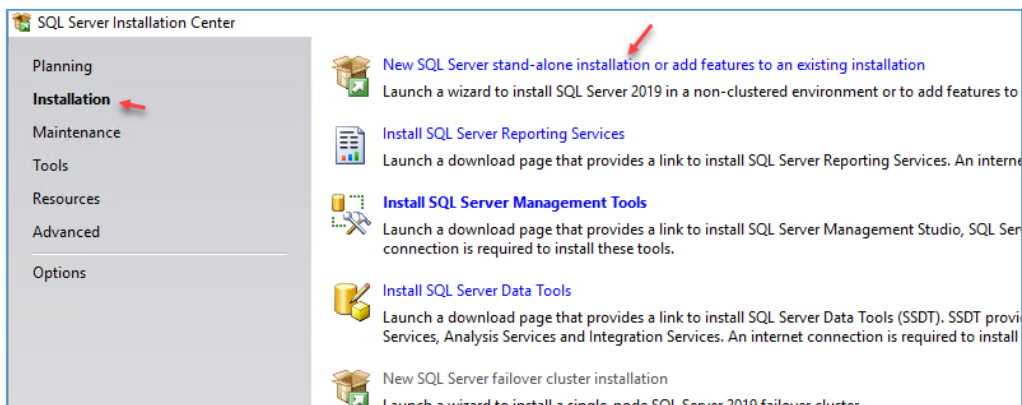


شکل ۲-۲ نصب SQL Server

Result	Rule	Status
✓	Setup administrator	Passed
✓	Restart computer	Passed
✓	Windows Management Instrumentation (WMI) service	Passed
✓	Consistency validation for SQL Server registry keys	Passed
✓	Long path names to files on SQL Server installation media	Passed
✓	SQL Server Setup Product Incompatibility	Passed
✓	Computer domain controller	Passed
✓	Edition WOW64 platform	Passed

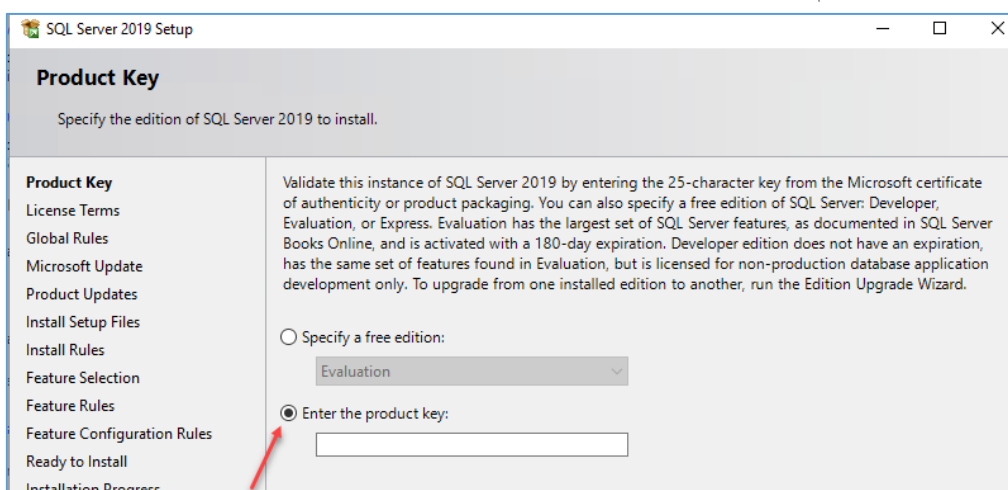
شکل ۲-۳ بررسی نصب SQL

برای اینکه نصب SQL Server را آغاز کنید، باید به‌مانند شکل ۲-۴ وارد Installation شوید و بر روی New SQL Server stand-alone installation or add feature to an existing installation کلیک کنید.



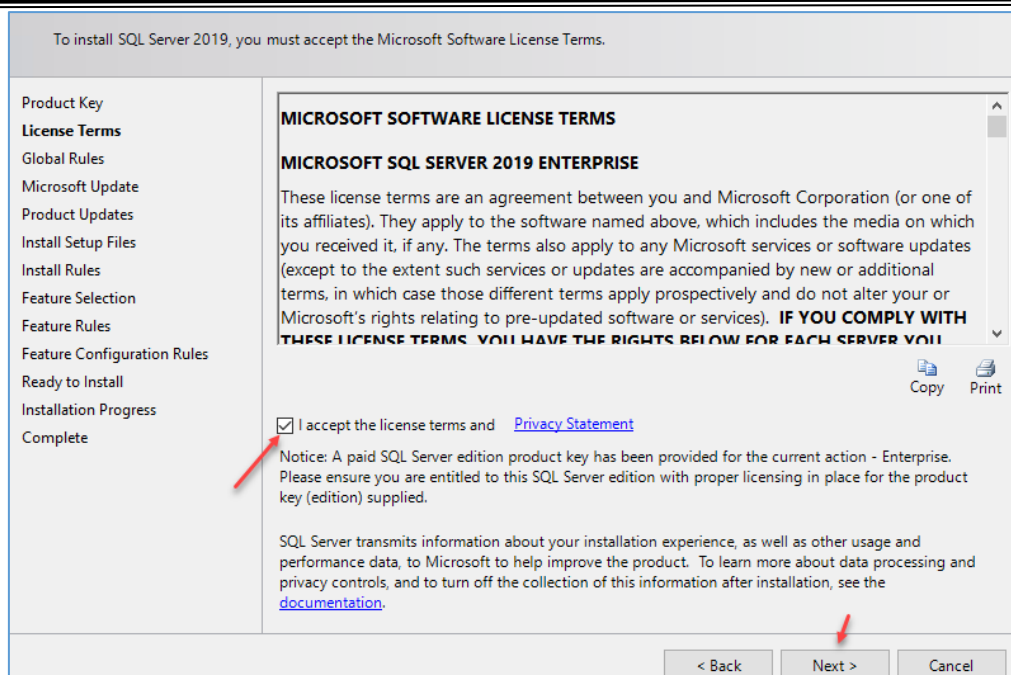
شکل ۲-۴ نصب SQL

در شکل ۲-۵ باید سریال نرم‌افزار را وارد و بر روی Next کلیک کنید.



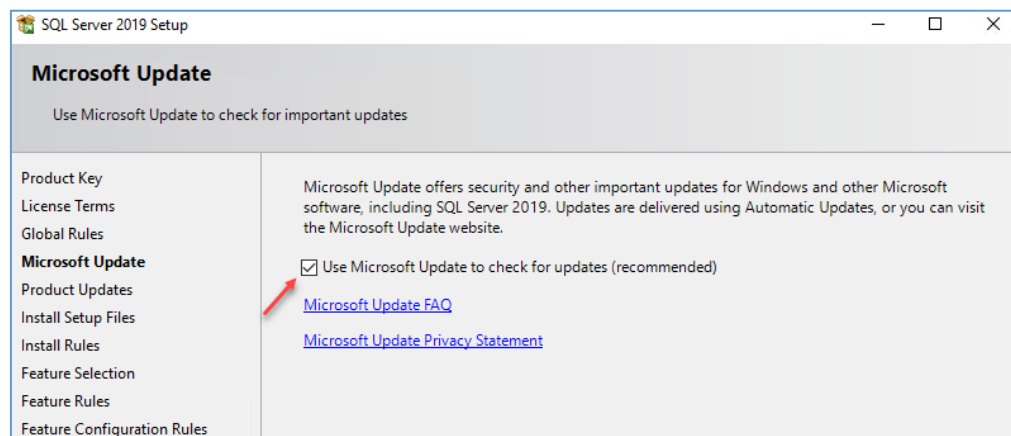
شکل ۲-۵ سریال برنامه

در شکل ۲-۶ باید توافقمای استفاده از این نرم‌افزار را مطالعه و در صورتی که آن را قبول دارید، تیک گزینه‌ی I accept the license terms را انتخاب و بر روی Next کلیک کنید.



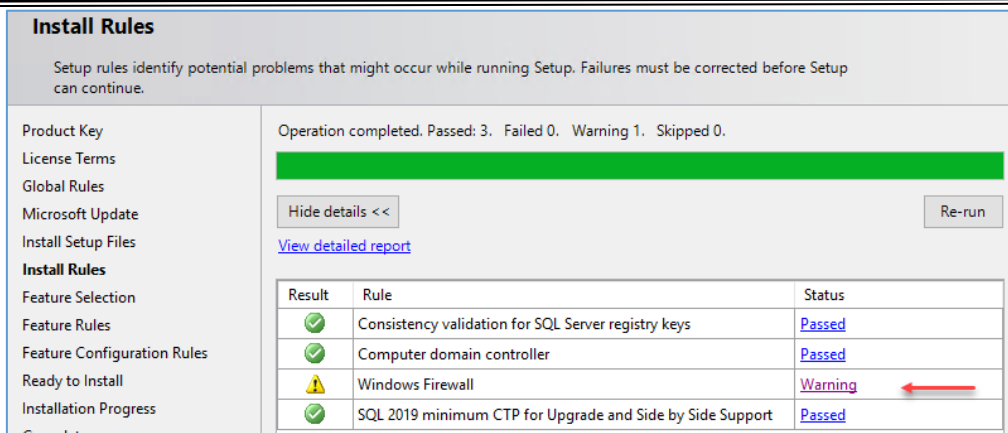
شکل ۲-۶ تأیید توافقتنامه

به‌مانند شکل ۲-۷ می‌توانید با انتخاب تیک گزینه‌ی Use Microsoft Updates To ... آخرین آپدیت‌های نرم‌افزار SQL را از سایت مایکروسافت دریافت کنید.



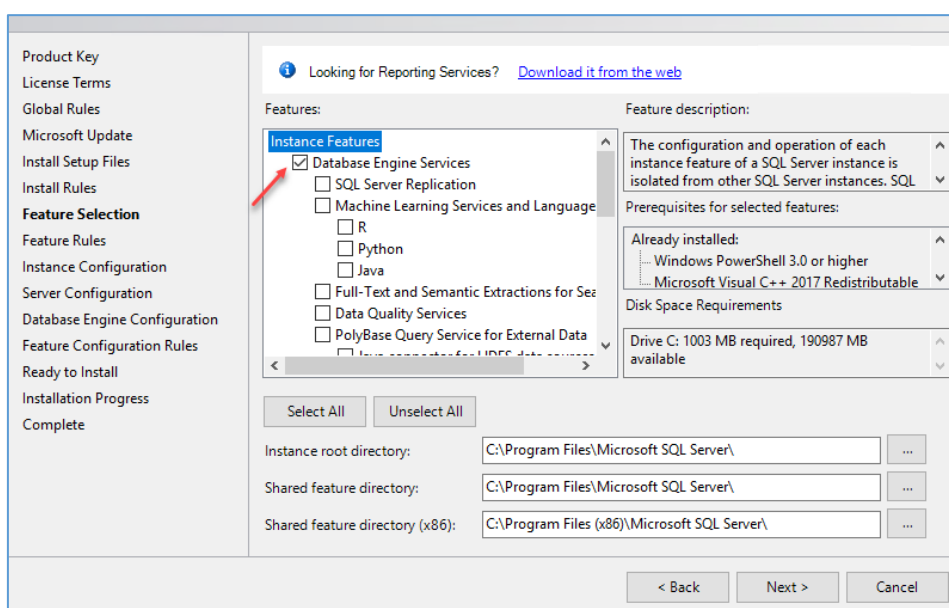
شکل ۲-۷ دریافت آپدیت

در شکل ۲-۸ بررسی اولیه انجام می‌شود و در این قسمت همه چیز باید Passed باشد، اگر به شکل ۲-۸ توجه کنید، متوجه خواهید شد که قسمت Firewall با یک اخطار روبرو شده است و به این موضوع اشاره دارد که برای استفاده از SQL در شبکه باید پورت‌های مورد نظر آن در فایروال باز باشد که با یاری خدا در ادامه این کار را انجام خواهیم داد.



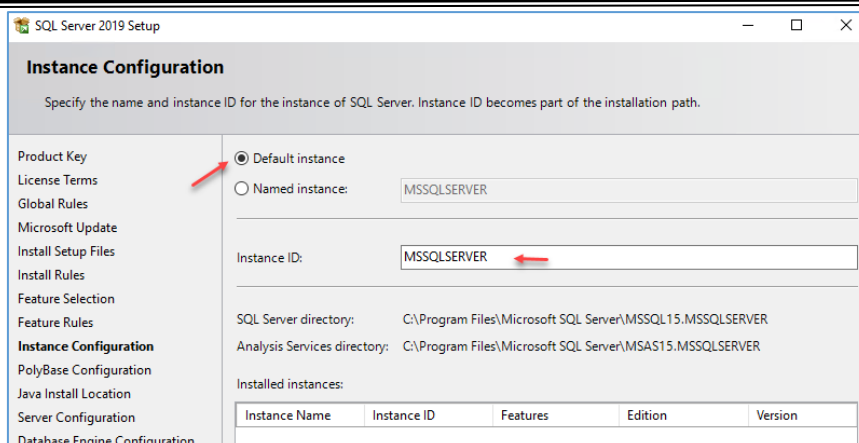
شکل ۲-۸ بررسی نیازمندی‌های اولیه

در شکل ۲-۹ باید تیک گزینه‌ی Database Engine را انتخاب کنید و همچنین می‌توانید مسیر نصب را تغییر دهید.



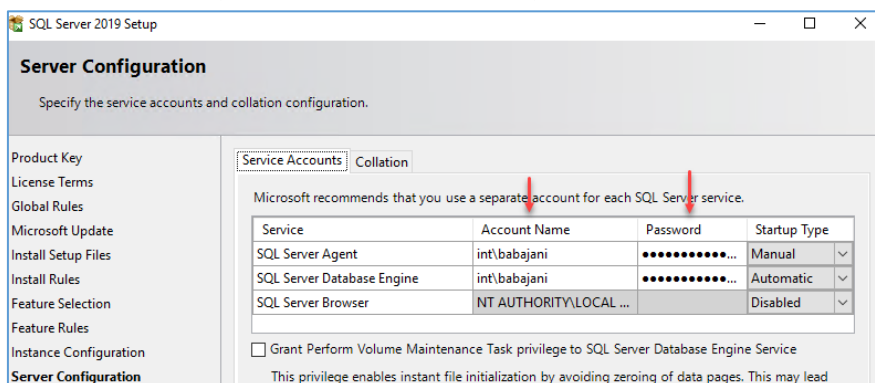
شکل ۲-۹ انتخاب Feature

در شکل ۲-۱۰ باید Instance را مشخص کنید، Instance را به‌عنوان یک ظرف در نظر بگیرید که داخل آن می‌توانید دیتابیس خود را قرار دهید و نرم‌افزارهای خاص خود را داشته باشید، مثلاً اطلاعاتی که در Instance با نام DB1 قرار دارد با اطلاعاتی که در Instance با نام DB2 قرار دارد متفاوت است، اصولاً Instance را به‌عنوان یک مزرعه‌ی جدا می‌شناسند که داخل آن می‌توانید دیتابیس‌ها و سرویس‌های خود را داشته باشید، در شکل ۲-۱۰ Instance با نام پیش‌فرض MSSQLSERVER قرار دارد که سرویس را بر روی آن فعال می‌کنیم.



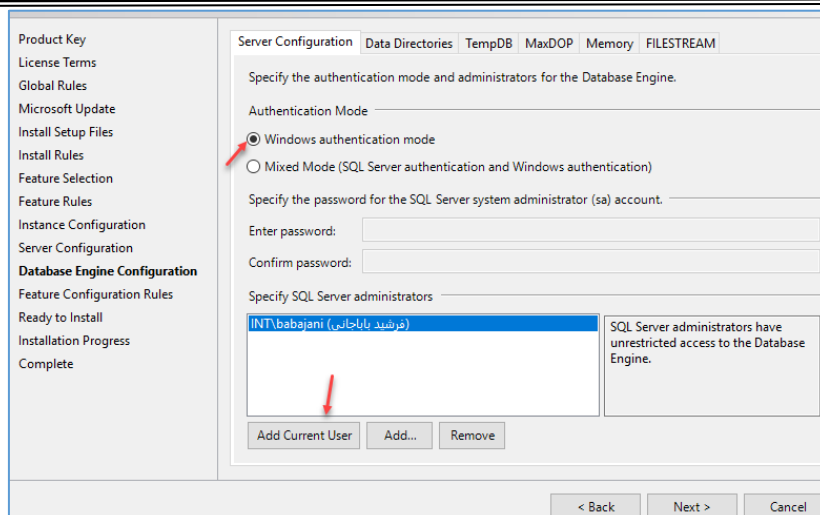
شکل ۲-۱۰ بررسی Instance

در شکل ۲-۱۱ باید یک کاربر را برای اجرای سرویس‌های SQL در نظر بگیرید، توجه داشته باشید که آن کاربر دسترسی لازم در شبکه را داشته باشد، برای همین منظور کاربر babajani که یک کاربر تحت دومین است با دسترسی کامل به شبکه وارد شده است و با همین کاربر هم باید در ادامه وارد SQL شویم.



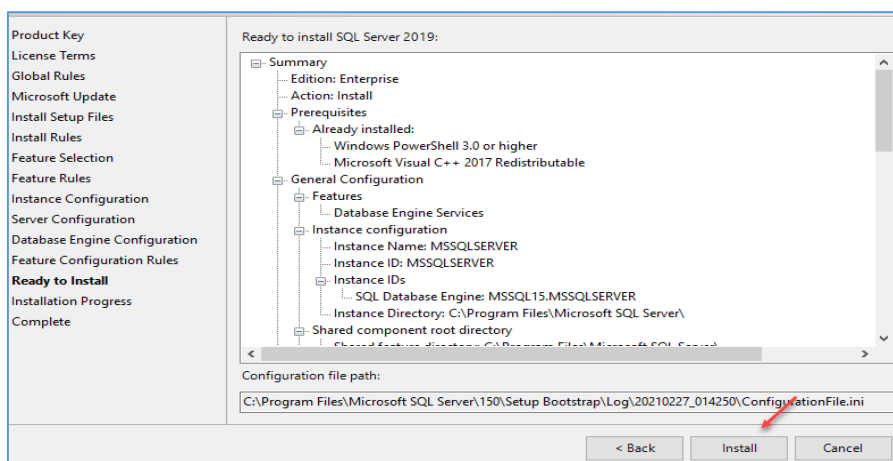
شکل ۲-۱۱ تنظیم سرور

در شکل ۲-۱۲ باید مشخص کنید که چه کاربرانی دسترسی به Database داشته باشند که با کلیک بر روی Add Current User می‌توانید با کاربری که در حال نصب SQL هستید، آن را به لیست اضافه کنید و یا اینکه هر کاربر دیگری که مورد نظر شماست با کلیک بر روی Add آن را به لیست اضافه کنید، توجه داشته باشید دو حالت احراز هویت وجود دارد که به صورت پیش فرض Windows authentication mode انتخاب شده است و اگر بخواهید کاربر sa که کاربر پیش فرض در SQL است باید در این قسمت گزینه Mixed Mode را انتخاب و یک رمز عبور برای آن در نظر بگیرید.



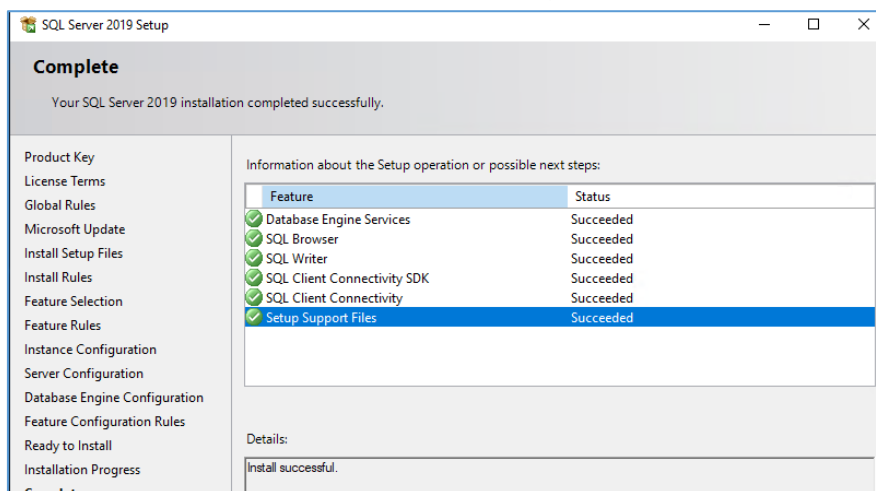
شکل ۲-۱۲ تنظیم دسترسی به دیتابیس

در شکل ۲-۱۳ بر روی Install کلیک کنید تا کار نصب آغاز شود.



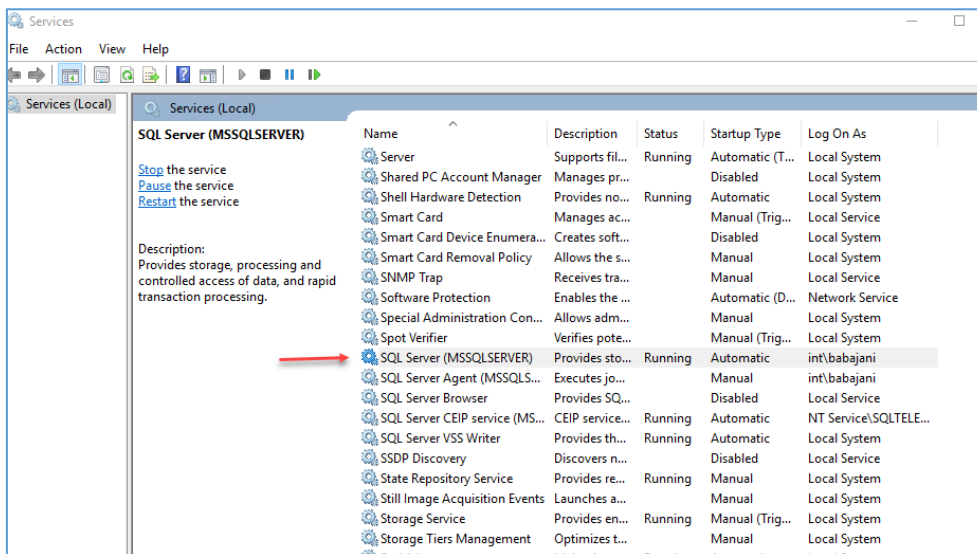
شکل ۲-۱۳ Install SQL

همانطور که در شکل ۲-۱۴ مشاهده می کنید نرم افزار SQL Server به درستی بر روی سرور نصب شده است.



شکل ۲-۱۴ Finish Install

اگر بعد از نصب به‌مانند شکل ۱۵-۲ وارد Services شوید، مشاهده خواهید کرد که سرویس SQL Server به‌درستی در حال اجرا است.

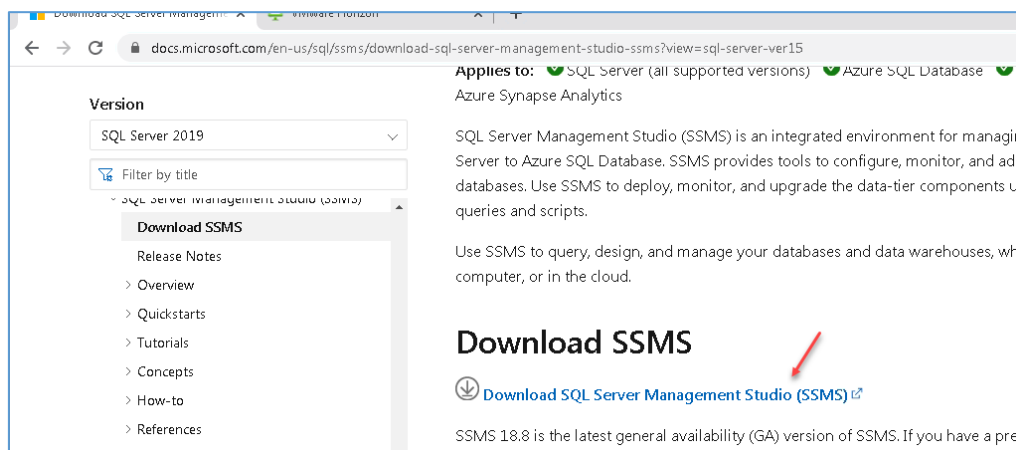


شکل ۱۵-۲ سرویس SQL Server

بعد از نصب نرم‌افزار SQL Server نیاز به یک نرم‌افزاری داریم تا بتوانیم SQL را مدیریت کنیم، یعنی کاربر جدید تعریف کنیم، دسترسی آن را مشخص کنیم، دیتابیس جدید و موارد دیگر؛ همه‌ی این کارها توسط نرم‌افزار SQL Server Management Tools یا به‌اختصار SSMS انجام خواهد گرفت، در نسخه‌های جدید این نرم‌افزار به‌صورت جداگانه ارائه می‌شود و می‌توانید از طریق لینک زیر آن را دانلود کنید:

<https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>

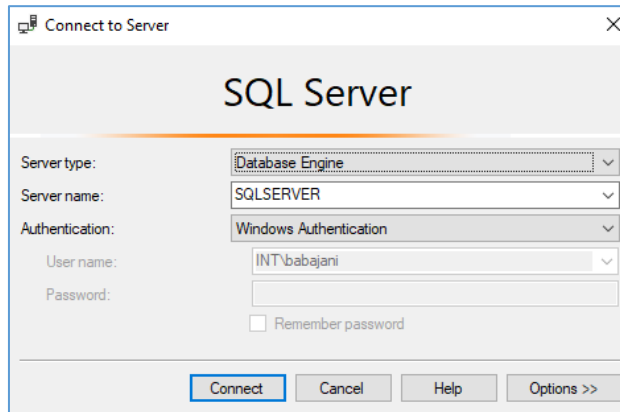
بعد از ورود به صفحه شکل ۱۶-۲ بر روی لینک دانلود کلیک کنید تا نرم‌افزار مورد نظر دانلود شود و بعد از دانلود آن را بر روی سرور و یا هر سیستمی که می‌خواهید نصب کنید.



شکل ۱۶-۲ دانلود SSMS

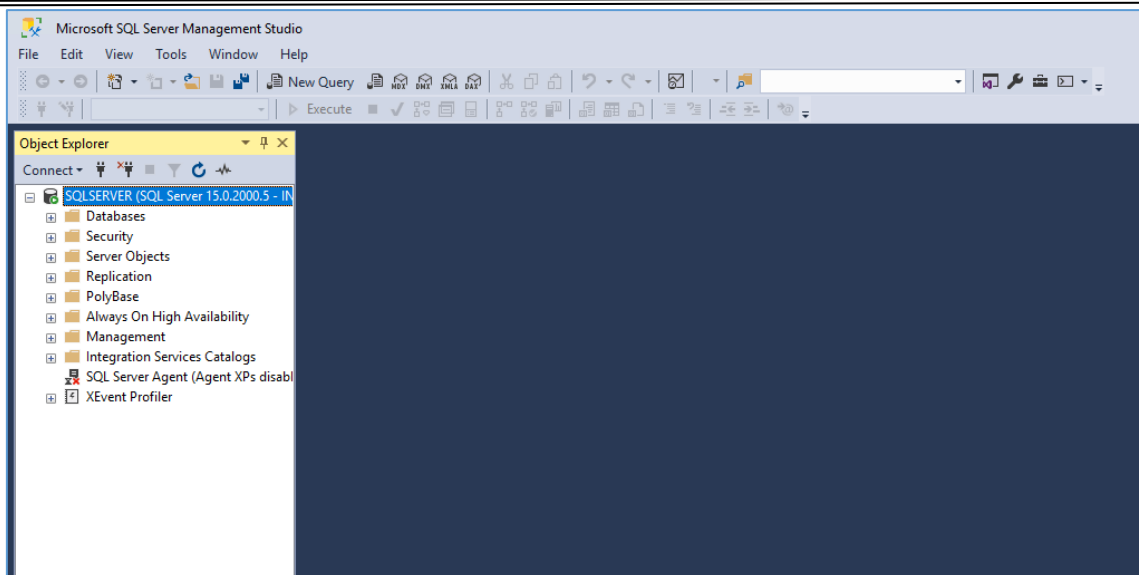
بعد از نصب و از طریق منوی Start نرم‌افزار SSMS را اجرا کنید که به‌مانند شکل ۱۷-۲ صفحه مورد نظر برای شما ظاهر خواهد شد. در این صفحه شما باید در قسمت Server Type، گزینه‌ی Database Engine را انتخاب کنید که شامل

دیتابیس شما خواهد بود؛ در قسمت Server Name باید نام سرور SQL خود را وارد کنید، توجه داشته باشید اگر داخل سرور هستید می‌توانید از نام localhost نیز استفاده کنید، اما برای دسترسی از بیرون باید از نام سرور استفاده کنید. در قسمت Authentication نیز اگر بر روی Windows Authentication قرار دهید با همان نام کاربری که Login کردید وارد خواهد شد که مسلماً باید دسترسی لازم را داشته باشد و یا اینکه می‌توانید از SQL Server Authentication استفاده کنید.



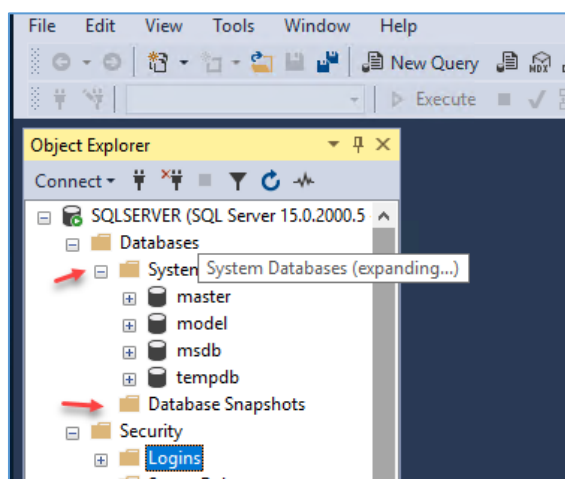
شکل ۱۷-۲ ورود به SQL Server

در شکل ۱۸-۲ یک نمای کلی از نرم‌افزار SSMS را که به Database Engine متصل شده است را مشاهده می‌کنید، در قسمت Databases می‌توانید دیتابیس‌های خود و دیتابیس‌هایی که توسط نرم‌افزارهای دیگر به صورت اتوماتیک ایجاد می‌شود را مشاهده کنید. در قسمت Security باید کاربران خود را معرفی و دسترسی‌های لازم برای آنها را مشخص کنید؛ در قسمت Server Objects یک سری اشیا یا همان Object وجود دارد که برای مانیتور کردن عملکرد سرور خواهد بود. در قسمت Replication نیز یک سری ابزار وجود دارد تا بتوانید دیتابیس‌ها را از یک پایگاه داده به یک پایگاه داده به صورت درست و امن انتقال دهید، PolyBase نیز یک ابزار برای انتقال اطلاعات از یک پایگاه داده متفاوت، مانند Oracle به SQL است؛ گزینه‌ی Always On High Availability برای ایجاد یک گروه از سرورها برای پایدار نگه داشتن دیتابیس‌ها و سرورها است که یک روش جدید در میان روش‌های دیگر است. در قسمت Managemnt یک سری ابزار وجود دارد، مانند پشتیبان‌گیری از دیتابیس‌ها، LOG گیری و موارد دیگر که در مدیریت SQL بسیار کمک‌کننده خواهد بود؛ قسمت Inetgaration Service Catalog نیز برای یکپارچه‌سازی داده‌ها در سازمان شما است که با یاری خدا همه‌ی این گزینه‌ها را در ادامه توضیح خواهیم داد و در آخر نیز سرویس SQL Agent برای انجام پشتیبان‌گیری و کارهایی دیگر موردنیاز است.



شکل ۲-۱۸ SQL Server Management Studio

اولین قسمتی که بررسی می‌کنیم Databases است، به‌مانند شکل ۲-۱۹، بعد از بازکردن قسمت Databases، دو گزینه را مشاهده می‌کنید، یکی Systems و دیگری Database Snapshots؛ در قسمت Systems، چهار دیتابیس را مشاهده می‌کنید که به‌صورت پیش‌فرض ایجاد می‌شوند.



شکل ۲-۱۹ بررسی Database

جدول ۲-۴ دیتابیس‌های سیستم

توضیحات	دیتابیس سیستم
<p>پایگاه داده Master به‌عنوان قلب تپنده SQL است و اگر از دست برود با مشکل مواجه خواهید شد، دیتابیس Master شامل اطلاعاتی حیاتی زیر است:</p> <ul style="list-style-type: none"> • زمانی که یک کاربر ایجاد می‌کنید، ID آن در این دیتابیس قرار می‌گیرد. • تمام رویدادها یا همان Logها در این دیتابیس قرار می‌گیرد. • نام و اطلاعات مربوط به پایگاه داده‌ها. 	master

<ul style="list-style-type: none"> • خطاهای سیستم و تمام پیام‌های موجود در سرور. • مقداردهی اولیه‌ی SQL Server . • پایگاه داده‌های محلی. • جدول‌های خاص پایگاه داده‌ها. 	
<p>همان‌طور که از اسم آن مشخص است، یک پایگاه‌داده موقت است، زمانی که شما ۲۰۱۹ Server SQL را اجرا می‌کنید، اطلاعات موقت در این پایگاه‌داده قرار می‌گیرد، مثالی که در این رابطه می‌توان زد، مانند RAM سیستم شما که زمانی نرم‌افزاری را اجرا می‌کنید، اطلاعات به‌صورت موقت در این حافظه قرار می‌گیرد و بعد از بستن نرم‌افزار، اطلاعات نیز از حافظه پاک می‌شوند. زمانی که یک دستور را در SQL اجرا می‌کنید، اطلاعات این دستور به‌صورت موقت در این پایگاه‌داده قرار می‌گیرد و پردازش می‌شود و طول عمر آن به کار کاربر مورد نظر برمی‌گردد.</p>	tempdb
<p>این پایگاه‌داده به‌عنوان یک الگو در نظر گرفته می‌شود، یعنی اینکه یک سری استانداردهایی در آن تعریف شده است که همه‌ی پایگاه داده‌ها از آن استاندارد پیروی می‌کنند. مجموعه‌های از پیش تعیین شده در این پایگاه وجود دارد که برای ساخت پایگاه داده‌های دیگر به کار می‌رود، مانند حجم پایگاه داده‌ها، اندازه‌ی جدول‌ها و موارد دیگر.</p>	model
<p>یک پایگاه‌داده‌ی فقط خواندنی که اطلاعات System Object در آن نگهداری می‌شود، البته در لیست ظاهری وجود ندارد.</p>	Resource
<p>در این پایگاه‌داده یک سری کارهای از پیش تعیین شده قرار دارد این کارها می‌تواند پشتیبان‌گیری و یا بازگردانی اطلاعات باشد که این کار بدون دخالت کسی و به‌صورت خودکار انجام می‌شود.</p>	msdb

در بالای نرم‌افزار چندین منو قرار دارد که با هم آنها را بررسی می‌کنیم:

در منوی فایل می‌توانید با کلیک بر روی Connect Object Explorer به دیتابیس جدید خود متصل شوید و یا با انتخاب گزینه‌ی Disconnect، ارتباط را قطع کنید؛ گزینه‌های دیگر برای ایجاد پروژه‌ی جدید و یا بازکردن پروژه‌های قبلی است؛ در قسمت Recent Projects and Solutuin می‌توانید آخرین پروژه‌هایی را که باز کردید را مشاهده کنید.

در منوی Edit می‌توانید اطلاعات را جستجو، کپی، حذف و... کنید.

در منوی View، گزینه‌ی Explorer Object ابزاری برای نمایش کلی دیتابیس‌ها، سرویس‌ها و... است که اگر به نرم‌افزار توجه کنید در سمت چپ، این ابزار را می‌توانید ببینید؛ گزینه‌ی Details Explorer Object ابزاری است زیرمجموعه‌ی ابزار Explorer Object که اطلاعات داخلی آن را نمایش می‌دهد.

گزینه‌ی Explorer Solution، ابزاری که در سمت راست برنامه ظاهر می‌شود و برای نمایش اطلاعات پروژه‌ی شما ایجاد شده است؛ این پروژه‌ها می‌تواند پروژه‌های در Visual Studio شما باشد.

گزینه‌ی Window Bookmark، این ابزار مفید برای ایجاد Bookmarkهایی در کد است که شما را سریع به کد مورد نظر در پروژه می‌رساند.

گزینه‌ی Explorer Utility، ابزاری برای مدیریت پایگاه‌داده‌ها و نظارت کلی بر روی آنها در داخل سازمان و یا سازمان‌هایی در فواصل دورتر از آن است.

منوی Debug، این منو برای کنترل پروژه است و برای بررسی و اشکال‌زدایی پروژه کاربرد دارد که در خلال کار بیشتر با آن آشنا خواهیم شد.

منوی Tools، با استفاده از Profiler Server SQL می‌توانیم تمام Logهای مربوط به پروژه را در مسیر مشخصی ذخیره کنیم تا بتوانیم در صورت مواجه شدن با مشکل آنها را بررسی کنیم.

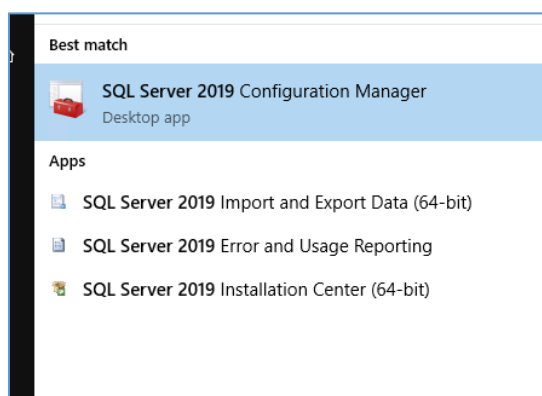
اگر در منوی فایل بر روی Options کلیک کنید، می‌توانید تنظیمات کلی نرم‌افزار SQL را مشاهده و آنها را تغییر دهید.

۲-۳ دسترسی از طریق شبکه به SQL Server

شاید شما در سازمان خود احتیاج داشته باشید که هم‌زمان چندین کاربر به SQL سرور شما متصل شوند و این کار نیاز به فعال کردن دسترسی از طریق شبکه به SQL است.

شما می‌توانید سرور SQL خود را عضو شبکه دومین خود کنید و به کاربرانی که نیاز به دسترسی به پایگاه‌داده دارند، دسترسی لازم را اعمال کنید.

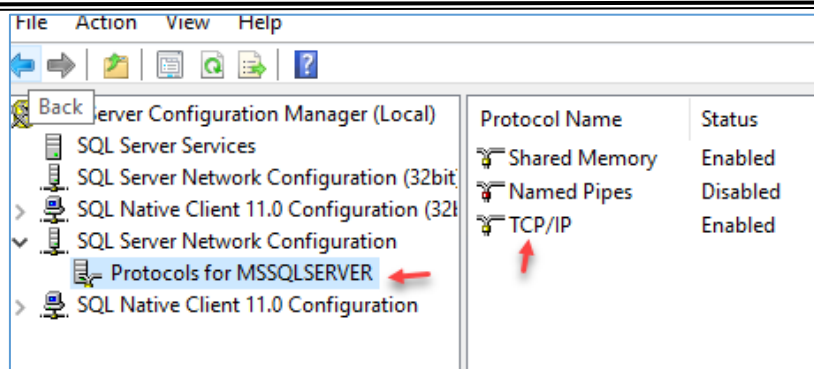
برای شروع باید وارد سرور SQL خود شوید و گزینه‌ی Configuration را در جستجو وارد کنید و ابزار SQL Server Configuration Manager 2019 را به‌مانند شکل ۲-۲۰ اجرا کنید.



شکل ۲-۲۰ سرویس Configuration

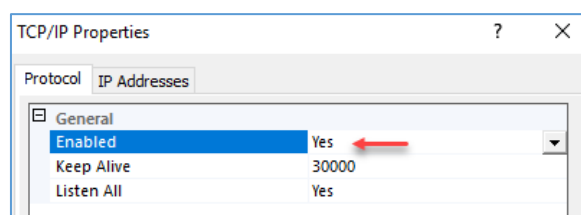
در شکل ۲-۲۱، از سمت چپ وارد SQL Server network Configuration شوید و بر روی گزینه‌ی مورد نظر کلیک کنید تا لیست آن باز شود.

در لیست مورد نظر بر روی TCP/IP دو بار کلیک کنید.



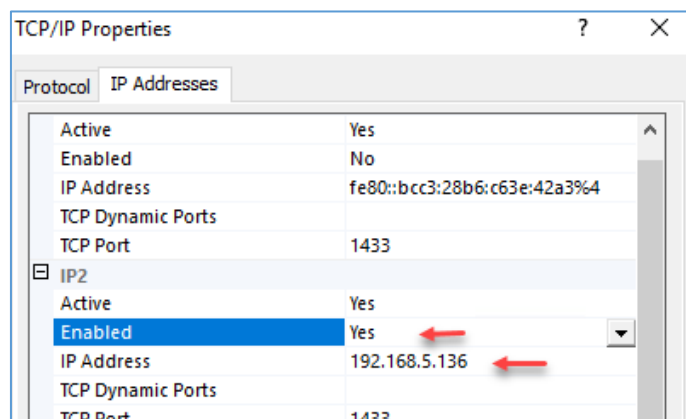
شکل ۲-۲۱ تنظیم پروتکل دسترسی

در شکل ۲-۲۲ و در تب Protocol، گزینه‌ی Enabled را در حالت Yes قرار دهید و وارد تب IP Addresses شوید.



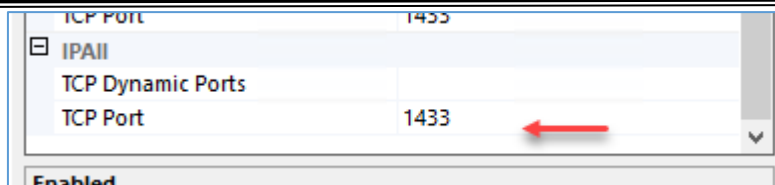
شکل ۲-۲۲ بررسی TCP/IP

به‌مانند شکل ۲-۲۳، در تب IP addresses و در قسمتی که IP سرور شما مشخص شده است باید دو گزینه‌ی اول را در حالت Yes قرار دهید و در قسمت TCP Port، شماره‌ی پورت ۱۴۳۳ را وارد کنید، زمانی که این پورت را وارد می‌کنید باید آن را در Firewall سیستم خود باز کنید تا کاربران بتوانند از طریق شبکه به SQL دسترسی داشته باشند. (در صورت خاموش بودن Firewall نیاز به این کار نیست).



شکل ۲-۲۳ تنظیم TCP/IP

برای اینکه تنظیمات پورت بر روی همه‌ی گزینه‌ها انجام شود، بهتر است در پایین شکل ۲-۲۳ گزینه‌ی TCP Port را به‌مانند شکل ۲-۲۴ بر روی 1433 قرار دهید.

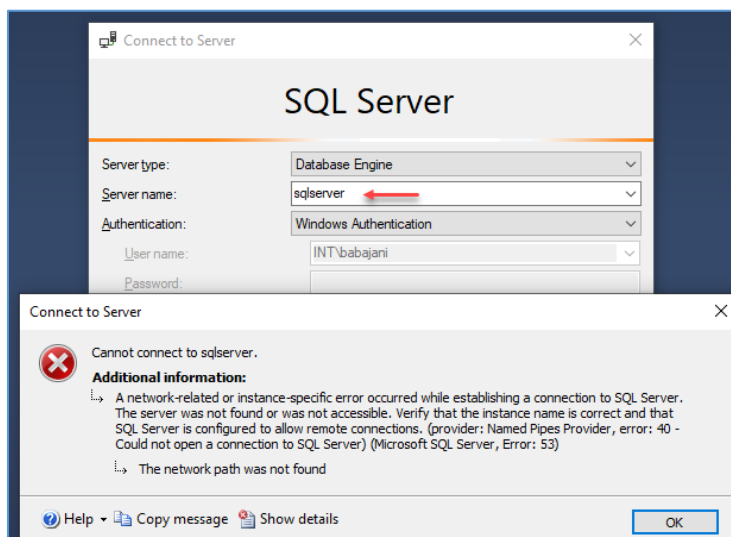


شکل ۲-۲۴

برای تست عملکرد ارتباطی با SQL از طریق شبکه، در یکی از کلاینت‌هایی که عضو شبکه است نرم‌افزار SSMS را نصب و اجرا می‌کنیم.

همان‌طور که در شکل ۲-۲۵ مشاهده می‌کنید در قسمت Server Name، نام سرور SQL خود را وارد کردیم و بعد از کلیک بر روی Connect با خطای مورد نظر روبرو شدیم؛ این خطا چند دلیل می‌تواند داشته باشد:

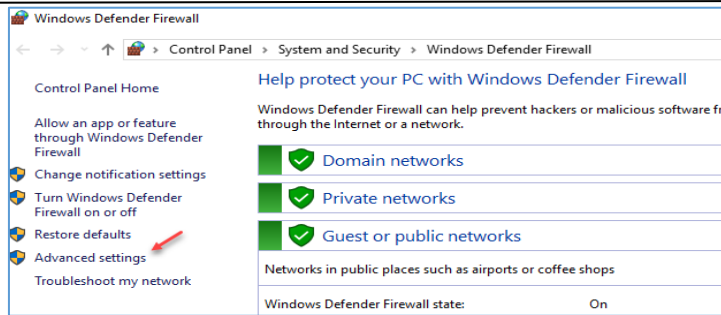
- ۱- شبکه در دسترس نباشد.
- ۲- Firewall مربوط به سرور SQL روشن باشد و جلوی ورود را بگیرد.
- ۳- تنظیمات سرور SQL مشکلی داشته باشد.



شکل ۲-۲۵ خطای ورود به SQL Server

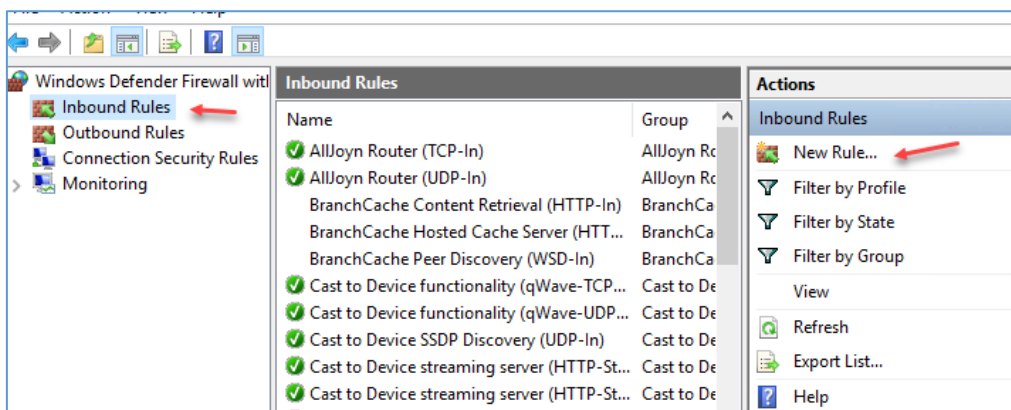
در گزینه‌ی اول که باید برای تست یکی از سرورهای دیگر در شبکه را تست بگیرید، توجه کنید که آن را می‌بینید یا نه، اگر اوکی بود که هیچ، اگر نه که باید شبکه را به صورت فیزیکی بررسی کنید، شاید کابل یا کارت شبکه ایراد دارد و یا مشکل نرم‌افزاری است.

در گزینه‌ی دوم باید وارد سرور SQL شوید و سرویس Firewall را اجرا کنید، برای این کار باید به‌مانند شکل ۲-۲۶ بر روی Advanced settings کلیک کنید.



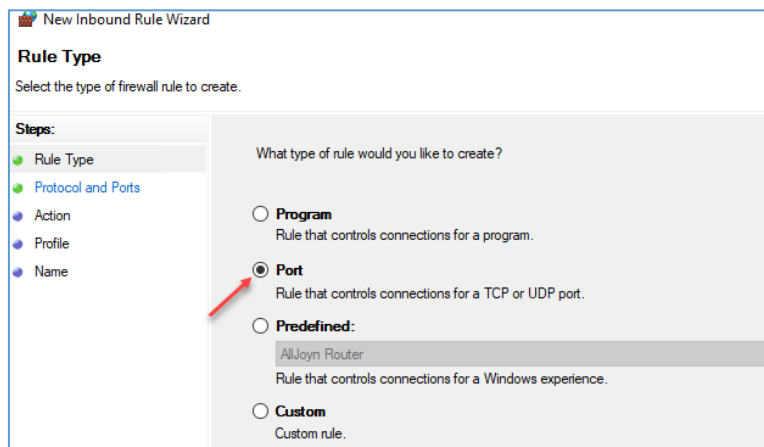
شکل ۲-۲۶ تنظیم فایروال

در شکل ۲-۲۷ برای اینکه به ترافیک ورودی به سرور SQL دسترسی لازم دهید باید بر روی Inbound Rules کلیک کنید و در صفحه‌ی باز شده بر روی New Rule کلیک کنید.



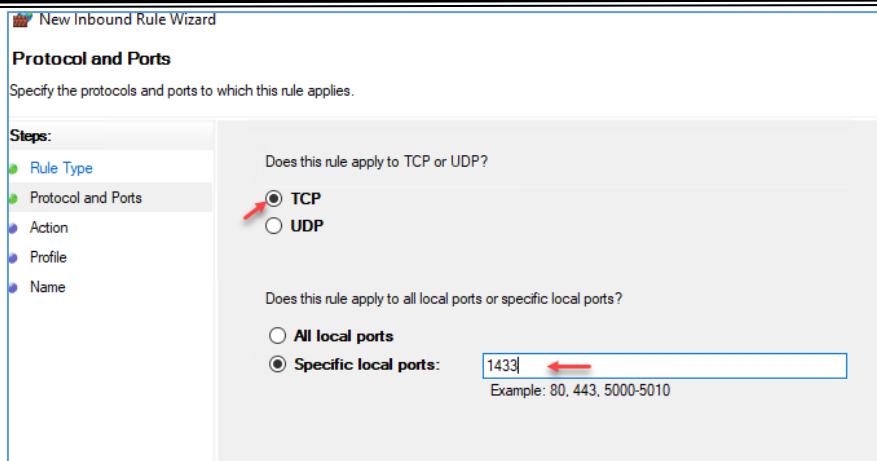
شکل ۲-۲۷ تنظیم فایروال

در شکل ۲-۲۸ باید گزینه‌ی Port را انتخاب و بر روی Next کلیک کنید.



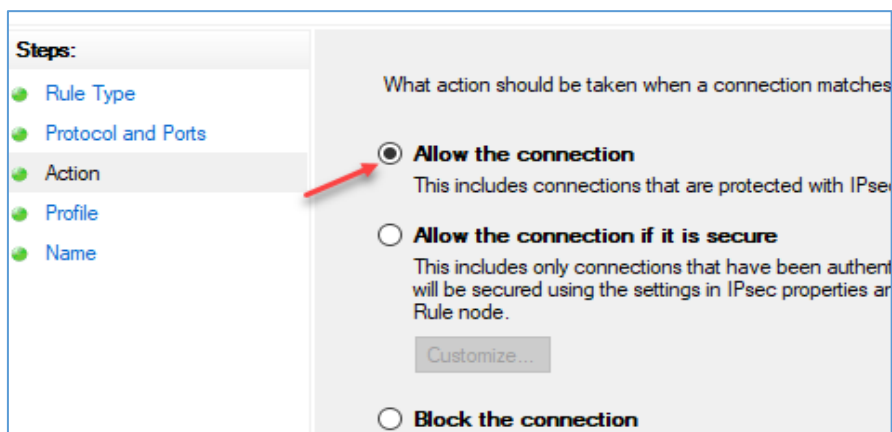
شکل ۲-۲۸ تنظیم Port SQL

در شکل ۲-۲۹ باید گزینه‌ی TCP را انتخاب کنید و در قسمت Specific Local ports، پورت ۱۴۳۳ مربوط به SQL را وارد کنید تا درخواست‌هایی که از بیرون به داخل سرور با این پورت وارد می‌شود، مجوز دسترسی داشته باشد.



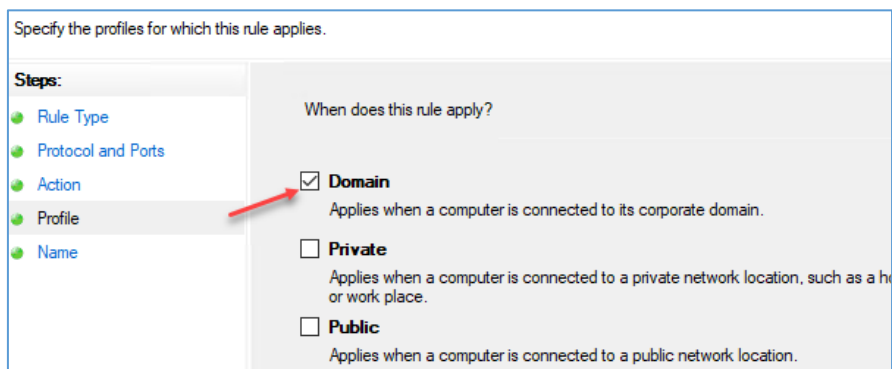
شکل ۲-۲۹ تنظیم Port SQL

در شکل ۲-۳۰ باید گزینه‌ی Allow the connection را انتخاب کنید.



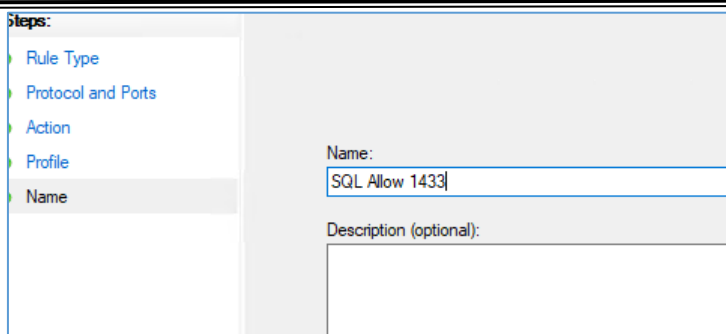
شکل ۲-۳۰ دسترسی به Port

در شکل ۲-۳۱ باید مشخص کنید که این دسترسی در چه قسمتی اعمال شود، اگر چنانچه کلاینت شما در منطقه‌ی دومین قرار دارد فقط گزینه‌ی دومین را انتخاب کنید، اما اگر کلاینت در منطقه‌ی دیگر، مثلاً در Workgroup قرار داشته باشد، نمی‌تواند به SQL متصل شود.



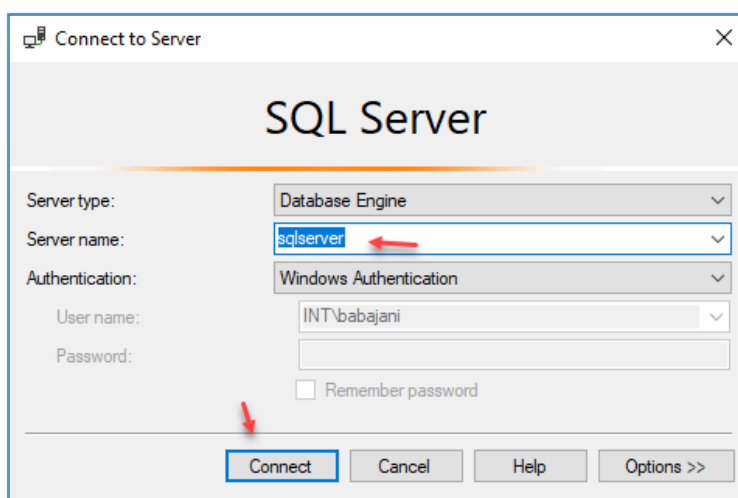
شکل ۲-۳۱ دسترسی به Port

یک نام به‌مانند شکل ۲-۳۲ وارد کنید و بر روی Finish کلیک کنید تا Rule مورد نظر ایجاد شود.



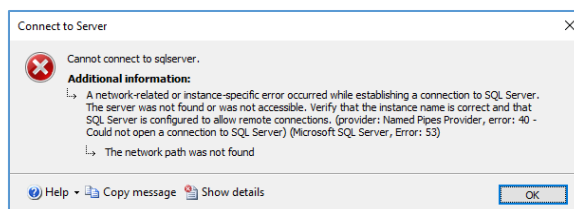
شکل ۲-۳۲ دسترسی به Port

بعد از ایجاد Rule در فایروال سرور SQL باید از طریق کلاینت تست بگیریم تا ببینیم می‌توانیم به سرور SQL از طریق شبکه متصل بشویم یا نه؛ برای این کار SQL Server Management Studio را طبق شکل ۲-۳۳ اجرا کنید و در قسمت Server Name باید نام سرور SQL خود را وارد و بر روی Connect کلیک کنید.



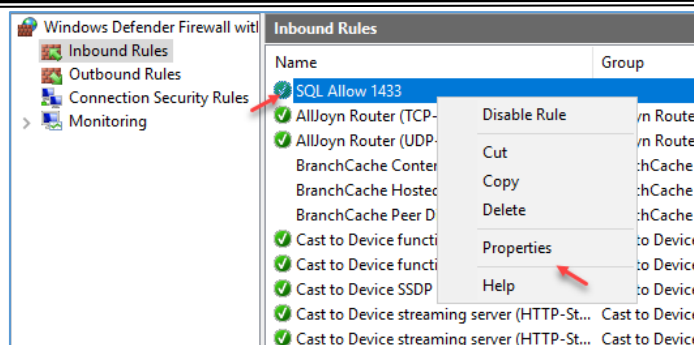
شکل ۲-۳۳ ورود به SQL Server

اگر مشکلی در ارتباط شبکه و دومین شما وجود نداشته باشد به راحتی به سرور SQL متصل خواهید شد، اما اگر طبق شکل ۲-۳۴ با خطا روبرو شدید، باید دوباره وارد فایروال سرور SQL شوید و منطقی کاری را تغییر بدهید.



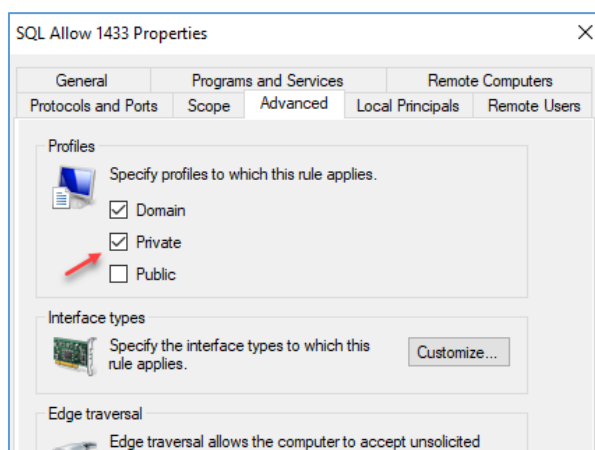
شکل ۲-۳۴ خطای ورود به SQL

طبق شکل ۲-۳۵ وارد فایروال SQL شوید و بر روی Rule مورد نظر خود کلیک راست و گزینه‌ی Properties را انتخاب کنید.



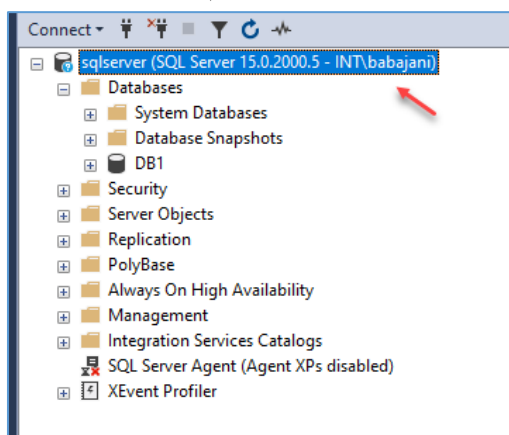
شکل ۲-۲۵ بررسی Firewall

در شکل ۲-۳۶ وارد تب Advanced شوید و تیک گزینه‌ی Private را انتخاب و بر روی OK کلیک کنید.



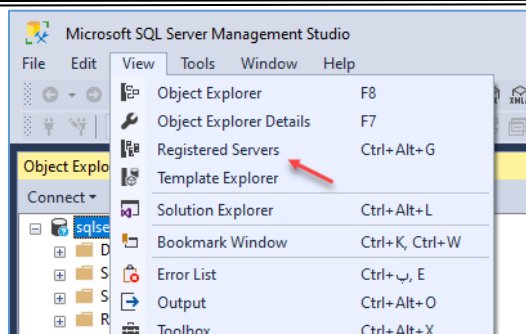
شکل ۲-۳۶ دسترسی Private

همان‌طور که در شکل ۲-۳۷ مشاهده می‌کنید به‌درستی توانستیم به سرور SQL از طریق شبکه متصل شویم.



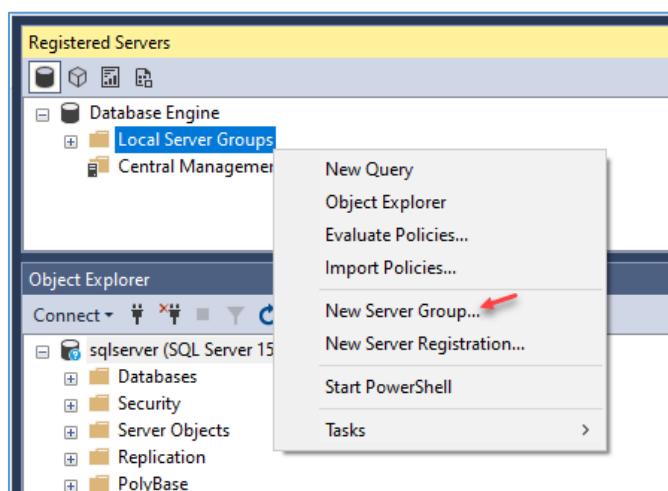
شکل ۲-۳۷ متصل شدن به SQL

شاید در سازمان خود چندین سرور SQL داشته باشید و به‌همه‌ی آنها از طریق شبکه متصل می‌شوید، برای راحتی کار خود بهتر است یک گروه ایجاد کنید و همه‌ی آنها را در گروه مورد نظر خود قرار دهید. برای این کار در نرم‌افزار SQL Management Studio به‌مانند شکل ۲-۳۸ وارد منوی View شوید و بر روی گزینه‌ی Registered Servers کلیک کنید.



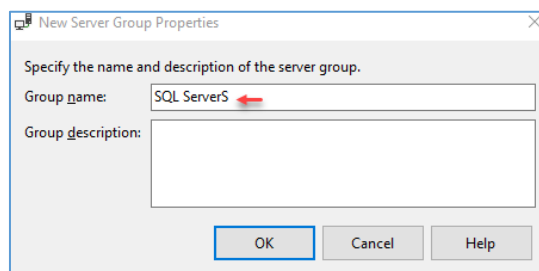
شکل ۲-۳۸ Register Server

در شکل ۲-۳۹ بر روی Local Server Groups کلیک راست کنید و گزینه‌ی New Server Group را انتخاب کنید.



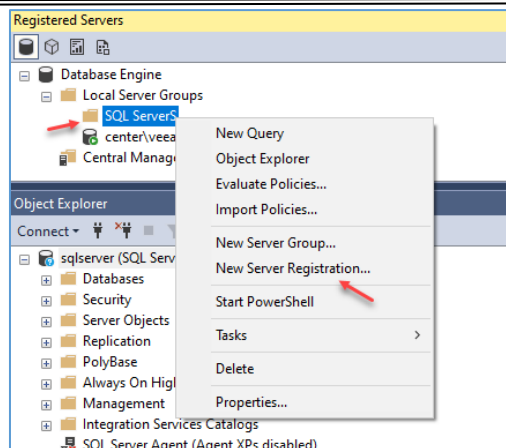
شکل ۲-۳۹ Register Server

در شکل ۲-۴۰ باید یک اسم برای گروه خود وارد کنید و بر روی OK کلیک کنید.



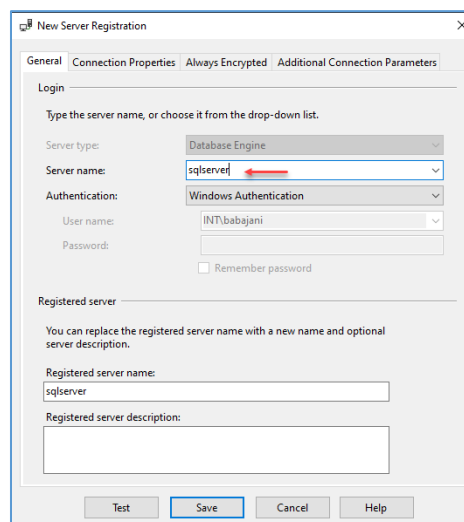
شکل ۲-۴۰ Register Server

بر روی نام گروهی که ایجاد کردید، طبق شکل ۲-۴۱ کلیک راست کنید و گزینه‌ی New Server Registration را انتخاب کنید.



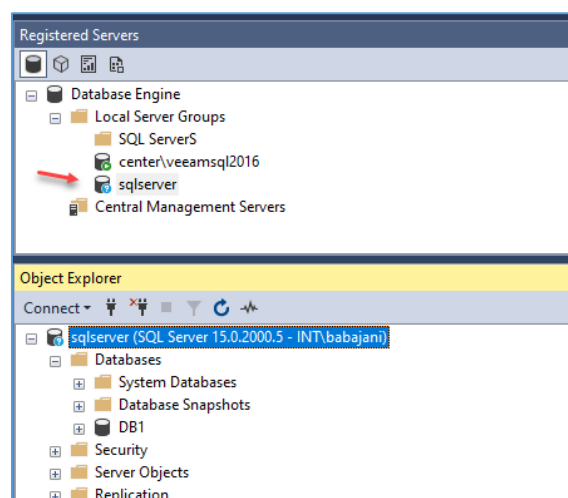
شکل ۲-۴۱ Register Server

در شکل ۲-۴۲ باید نام سرور را در قسمت Server name وارد کنید و بر روی Save کلیک کنید.



شکل ۲-۴۲ Register Server

همان‌طور که در شکل ۲-۴۳ مشاهده می‌کنید، سرور به لیست اضافه شده است و با کلیک بر روی آن به راحتی به سرور SQL متصل خواهید شد.



شکل ۲-۴۳ Register Server

۲-۴ نصب و راه‌اندازی SQL Server 2019 در لینوکس Ubuntu

تا به اینجا توانستیم نرم‌افزار SQL را بر روی ویندوز نصب کنیم و آن را اجرا و از راه دور به آن متصل شویم، اما شرکت مایکروسافت در نسخه‌های جدید خود، توانایی نصب SQL را بر روی سیستم‌عامل لینوکس فعال کرده است که با هم در این قسمت نحوه‌ی راه‌اندازی آن را فرا خواهیم گرفت.

برای شروع کار باید یک نسخه از سیستم عامل لینوکس که واقعاً هم زیاد است را انتخاب کنید؛ برای این قسمت، Linux Ubuntu را انتخاب می‌کنیم. برای دانلود Linux Ubuntu می‌توانید از لینک زیر استفاده کنید.

<https://ubuntu.com/download/desktop>

بعد از دانلود لینوکس Ubuntu باید آن را پیاده‌سازی کنید، بهترین کار این است که از یک نرم‌افزار مجازی‌سازی، مانند VMware Workstation و یا برنامه‌ی دیگر برای این کار استفاده کنید تا سهولت دسترسی و عملکرد بهتری را شاهد باشید، اگر برنامه VMware Workstation را در دسترس ندارید می‌توانید از لینک زیر آن را دانلود کنید:

<https://soft98.ir/os/virtual-machine/1232-vmware-workstation.html>

بعد از دانلود، آن را نصب کنید و برای آغاز کار باید به‌مانند شکل ۲-۴۴ وارد منوی File شوید و گزینه‌ی New Virtual Machine را انتخاب کنید.



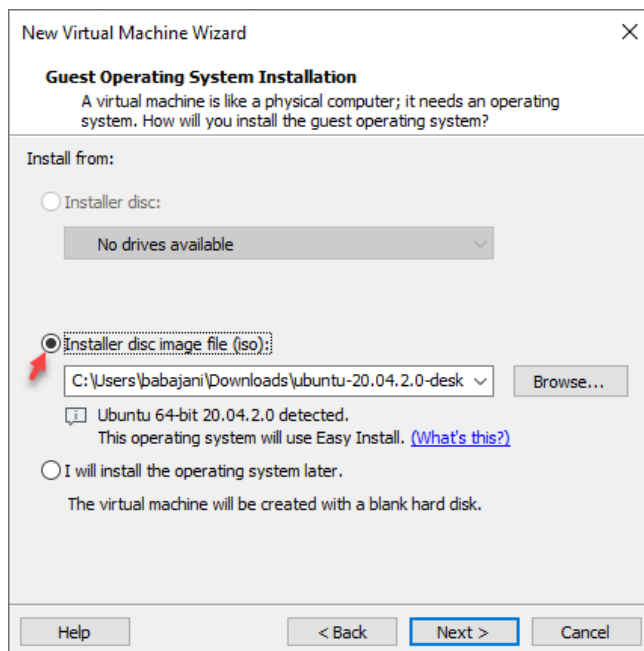
شکل ۲-۴۴ ایجاد ماشین مجازی

در شکل ۲-۴۵ گزینه‌ی Typical را انتخاب و بر روی Next کلیک کنید.



شکل ۲-۴۵ ایجاد ماشین مجازی

در شکل ۲-۴۶ باید فایل ISO مربوط به لینوکس Ubuntu را که لینک آن را در قسمت پیش قرار دادیم معرفی کنید، بعد از معرفی به صورت اتوماتیک نوع سیستم عامل در زیر آن مشخص خواهد شد.



شکل ۲-۴۶ ایجاد ماشین مجازی

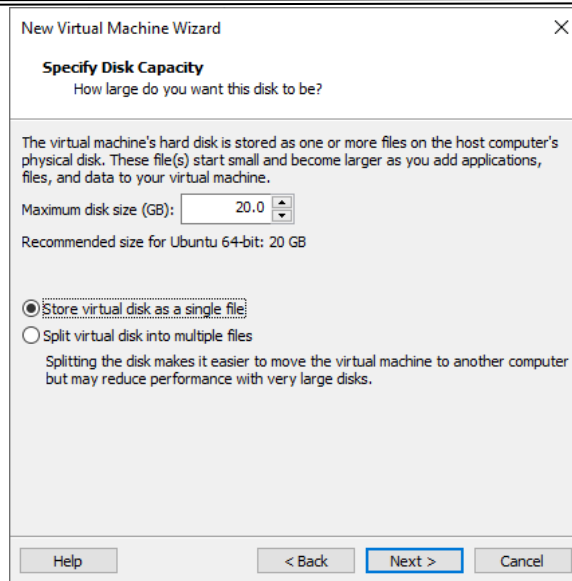
در شکل ۲-۴۷ باید اطلاعات تکمیلی را وارد کنید، مانند نام کاربری و رمز عبور برای لینوکس Ubuntu.

شکل ۲-۴۷ ایجاد ماشین مجازی

در شکل ۲-۴۸ باید نام ماشین مجازی خود به همراه آدرس ذخیره‌سازی آن در هارددیسک را مشخص کنید.

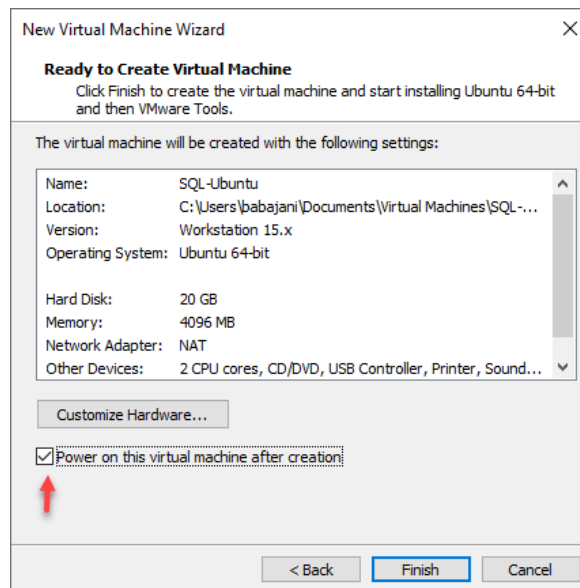
شکل ۲-۴۸ ایجاد ماشین مجازی

در شکل ۲-۴۹ باید مقدار فضای هارددیسک را مشخص کنید که برای این سیستم‌عامل، ۲۰ گیگابایت کفایت می‌کند و برای اینکه هارددیسک مجازی فقط یک فایل تکی باشد باید گزینه‌ی Store Virtual disk as single Files را انتخاب کنید.



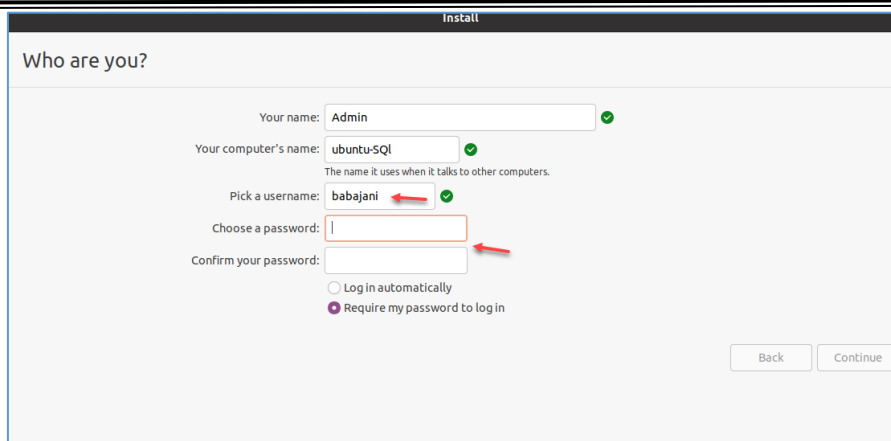
شکل ۲-۴۹ ایجاد ماشین مجازی

در شکل ۲-۵۰ اطلاعات کلی را مشاهده می‌کنید که اگر نیاز باشد تغییراتی در سخت‌افزار اعمال کنید و یا اینکه بخواهید سخت‌افزار جدید به این ماشین اضافه کنید باید بر روی **Customize Hradware** کلیک کنید، در پایان برای اینکه ماشین بعد از ایجاد شدن، روشن شود تیک گزینه‌ی **Power on** را انتخاب کنید.



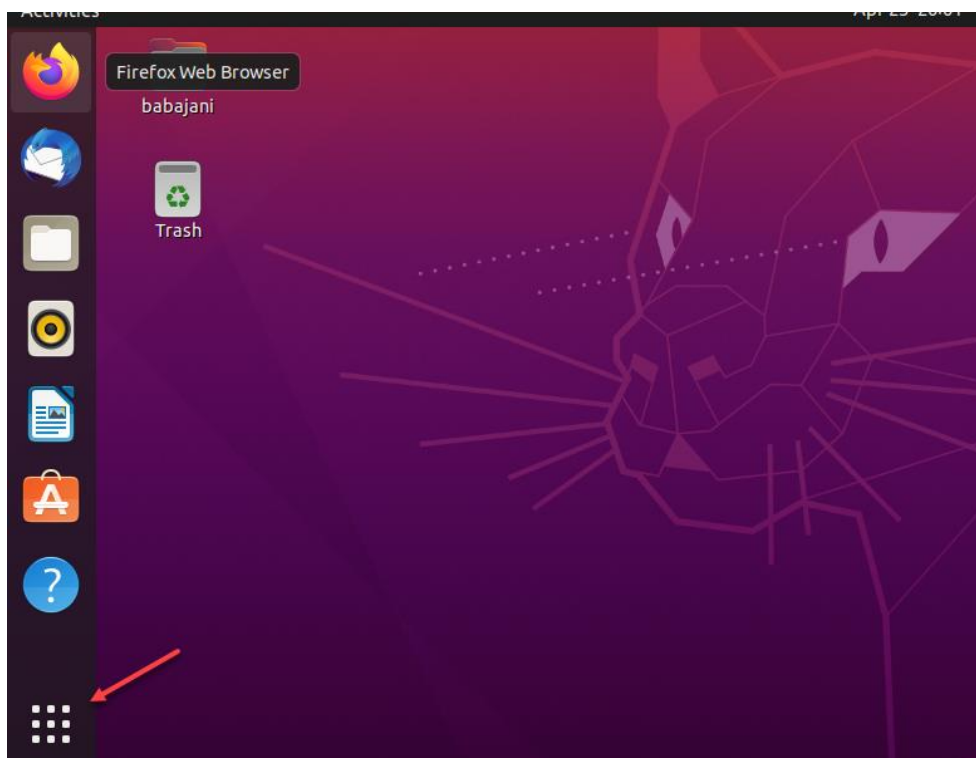
شکل ۲-۵۰ ایجاد ماشین مجازی

بعد از اجرای ماشین مورد نظر مراحل نصب به صورت اتوماتیک انجام خواهد شد، اما به مانند شکل ۲-۵۱ چون نام کاربری **admin** یک نام رزرو شده است باید یک نام جدید، وارد و رمز آن را مشخص کنید و بر روی **Continue** کلیک کنید تا کار نصب به پایان برسد.



شکل ۲-۵۱ نصب سیستم عامل لینوکس

در ادامه‌ی کار و بعد از نصب کامل لینوکس Ubuntu باید به‌مانند شکل ۲-۵۲ بر روی Show Applications کلیک کنید و در کادر جستجو، Terminal را اجرا کنید.



شکل ۲-۵۲ لینوکس Ubuntu

بعد از اجرا شدن سرویس باید به‌مانند شکل ۲-۵۳ کاربر Root را فعال کنید، برای این کار از دو دستور زیر استفاده کنید:

```
sudo passwd root
sudo passwd -u root
```

توجه داشته باشید بعد از اجرای دستور اول باید یک رمز عبور جدید برای کاربر Root وارد کنید تا بتوانید از آن استفاده کنید.

```

babajani@ubuntu-SQL: ~
babajani@ubuntu-SQL:~$ sudo passwd root
[sudo] password for babajani:
New password:
Retype new password:
passwd: password updated successfully
babajani@ubuntu-SQL:~$ sudo passwd -u root
passwd: password expiry information changed.
babajani@ubuntu-SQL:~$

```

شکل ۲-۵۳ فعال‌سازی کاربر root

در ادامه‌ی کار با دستور `sudo -i`، کاربر را به root تغییر دهید تا بتوانید دستورات مورد نظر خود را با بالاترین دسترسی اجرا کنید.

```

root@ubuntu-SQL: ~
babajani@ubuntu-SQL:~$ sudo -i
[sudo] password for babajani:
root@ubuntu-SQL:~#

```

شکل ۲-۵۴ اجرای کاربر root

برای فعال‌سازی SQL Server بر روی لینوکس Ubuntu باید دستورات زیر را وارد کنید.
 دو دستور زیر آخرین آپدیت‌ها را بر روی لینوکس Ubuntu نصب می‌کند تا در ادامه بتوانید، دستورات را به راحتی اجرا کنید که نتیجه‌ی آن را در شکل ۲-۵۵ مشاهده می‌کنید.

```

sudo apt-get update
sudo apt-get -y upgrade

```

```

root@ubuntu-SQL: ~
babajani@ubuntu-SQL:~$ sudo -i
[sudo] password for babajani:
root@ubuntu-SQL:~# sudo apt-get update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [109 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [24.3 kB]
Hit:4 https://packages.microsoft.com/ubuntu/18.04/mssql-server-2019 bionic InRelease
Get:5 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [184 kB]
Hit:6 https://packages.microsoft.com/ubuntu/18.04/prod bionic InRelease
Get:7 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Get:9 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metadata [264 kB]
Get:10 http://us.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [207 kB]
Get:11 http://us.archive.ubuntu.com/ubuntu focal-updates/restricted i386 Packages [16.3 kB]
Get:12 http://us.archive.ubuntu.com/ubuntu focal-updates/universe amd64 DEP-11 Metadata [303 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [24.0 kB]

```

شکل ۲-۵۵ آپدیت لینوکس Ubuntu

با دستور زیر، بعد از اینکه آپدیت‌ها فعال شدند لینوکس را یک بار Restart کنید.

```

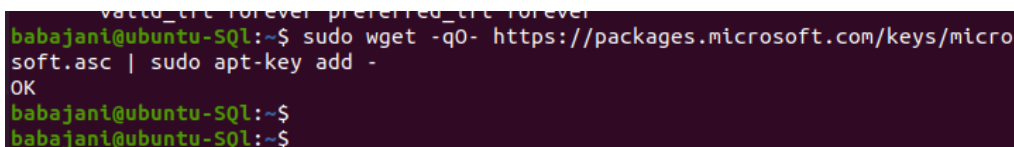
sudo reboot

```

بعد از اجرا شدن سرور مجازی لینوکس، وارد ترمینال شوید و دستور زیر را برای فعال‌سازی repository اجرا کنید.

```
sudo wget -qO- https://packages.microsoft.com/keys/microsoft.asc | sudo
apt-key add -
```

همان‌طور که در شکل ۲-۵۶ مشاهده می‌کنید، دستور مورد نظر به‌درستی اجرا شده است.

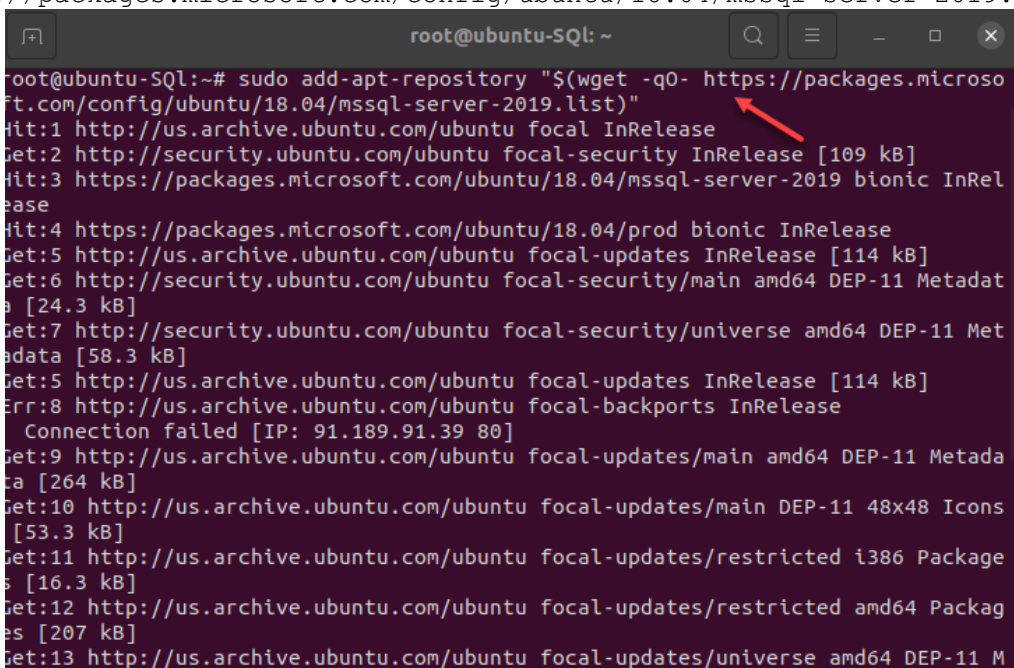


```
babajani@ubuntu-SQL:~$ sudo wget -qO- https://packages.microsoft.com/keys/micro
soft.asc | sudo apt-key add -
OK
babajani@ubuntu-SQL:~$
babajani@ubuntu-SQL:~$
```

شکل ۲-۵۶ فعال‌سازی Repository

در ادامه‌ی کار با دستور زیر، Repository یا همان مخزن مربوط به SQL Server را دانلود و فعال می‌کنیم، اگر به شکل ۲-۵۷ دقت کنید این دستور به‌درستی اجرا شده است.

```
sudo add-apt-repository "$ (wget -qO-
https://packages.microsoft.com/config/ubuntu/18.04/mssql_server-2019.list)"
```



```
root@ubuntu-SQL:~# sudo add-apt-repository "$ (wget -qO- https://packages.microso
ft.com/config/ubuntu/18.04/mssql_server-2019.list)"
Hit:1 http://us.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://security.ubuntu.com/ubuntu focal-security InRelease [109 kB]
Hit:3 https://packages.microsoft.com/ubuntu/18.04/mssql-server-2019 bionic InRel
ease
Hit:4 https://packages.microsoft.com/ubuntu/18.04/prod bionic InRelease
Get:5 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:6 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadat
a [24.3 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/universe amd64 DEP-11 Met
adata [58.3 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Err:8 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease
Connection failed [IP: 91.189.91.39 80]
Get:9 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metada
ta [264 kB]
Get:10 http://us.archive.ubuntu.com/ubuntu focal-updates/main DEP-11 48x48 Icons
[53.3 kB]
Get:11 http://us.archive.ubuntu.com/ubuntu focal-updates/restricted i386 Package
s [16.3 kB]
Get:12 http://us.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packag
es [207 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu focal-updates/universe amd64 DEP-11 M
```

شکل ۲-۵۷ نصب Repository

در ادامه‌ی کار با دستور زیر، MSSQL Server را بر روی لینوکس Ubuntu نصب کنید:

```
sudo apt update
sudo apt install mssql-server
```

همان‌طور که در شکل ۲-۵۸ مشاهده می‌کنید، دستورات بالا اجرا و MSSQL نصب شده است که در ادامه باید آن را تنظیم کنید.


```

Activities Terminal Apr 23 06:20
babajani@ubuntu-SQL: ~
..
Unpacking libssl2-modules-gssapi-mit:amd64 (2.1.27+dfsg-2) ...
Selecting previously unselected package libc++1:amd64.
Preparing to unpack .../4-libc++1_1%3a10.0-50-exp1_amd64.deb ...
Unpacking libc++1:amd64 (1:10.0-50-exp1) ...
Selecting previously unselected package libsss-nss-idmap0.
Preparing to unpack .../5-libsss-nss-idmap0_2.2.3-3ubuntu0.4_amd64.deb ...
Unpacking libsss-nss-idmap0 (2.2.3-3ubuntu0.4) ...
Selecting previously unselected package mssql-server.
Preparing to unpack .../6-mssql-server_15.0.4123.1-5_amd64.deb ...
Unpacking mssql-server (15.0.4123.1-5) ...
Setting up gawk (1:5.0.1+dfsg-1) ...
Setting up libc++abi1-10:amd64 (1:10.0.0-4ubuntu1) ...
Setting up libsss-nss-idmap0 (2.2.3-3ubuntu0.4) ...
Setting up libssl2-modules-gssapi-mit:amd64 (2.1.27+dfsg-2) ...
Setting up libc++1-10:amd64 (1:10.0.0-4ubuntu1) ...
Setting up libc++1:amd64 (1:10.0-50-exp1) ...
Setting up mssql-server (15.0.4123.1-5) ...
+-----+
Please run 'sudo /opt/mssql/bin/mssql-conf setup'
to complete the setup of Microsoft SQL Server
+-----+
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
babajani@ubuntu-SQL:~$

```

شکل ۲-۵۸ نصب MSSQL

برای تنظیم و استارت سرویس MSSQL باید دستور زیر را وارد کنید:

```
/opt/mssql/bin/mssql-conf setup
```

با اجرای دستور بالا، شکل ۲-۵۹ ظاهر خواهد شد که باید یک نسخه از SQL سرور را انتخاب کنید که در اینجا گزینه‌ی یک را انتخاب می‌کنیم.

```

babajani@ubuntu-SQL:~$ sudo /opt/mssql/bin/mssql-conf setup
[sudo] password for babajani:
usermod: no changes
Choose an edition of SQL Server:
 1) Evaluation (free, no production use rights, 180-day limit)
 2) Developer (free, no production use rights)
 3) Express (free)
 4) Web (PAID)
 5) Standard (PAID)
 6) Enterprise (PAID) - CPU Core utilization restricted to 20 physical/40 hyperthreaded
 7) Enterprise Core (PAID) - CPU Core utilization up to Operating System Maximum
 8) I bought a license through a retail sales channel and have a product key to enter.

Details about editions can be found at
https://go.microsoft.com/fwlink/?LinkId=2109348&clcid=0x409

Use of PAID editions of this software requires separate licensing through a
Microsoft Volume Licensing program.
By choosing a PAID edition, you are verifying that you have the appropriate
number of licenses in place to install and run this software.

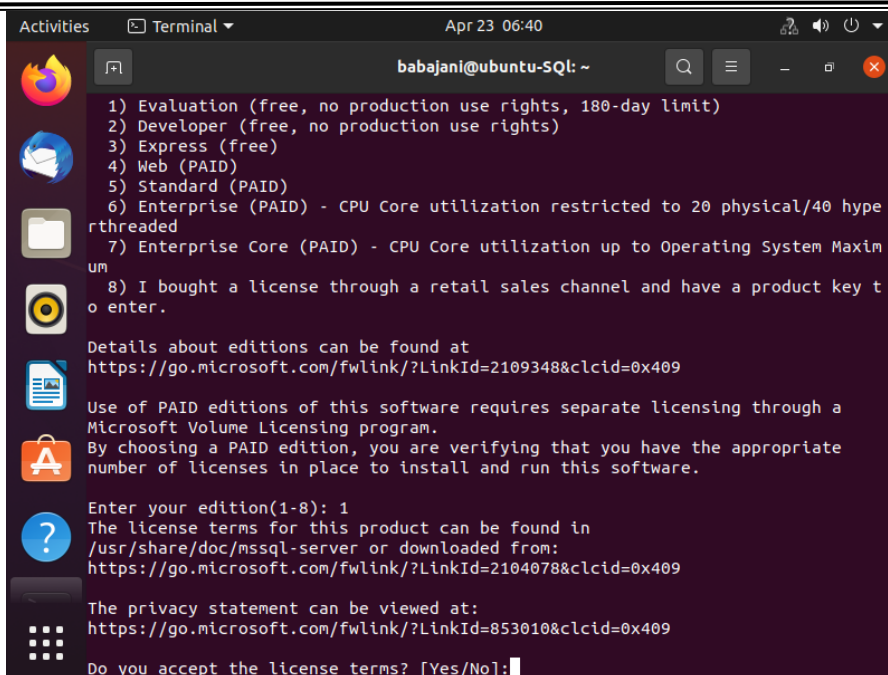
Enter your edition(1-8): 1

```

شکل ۲-۵۹ اجرای سرویس MSSQL

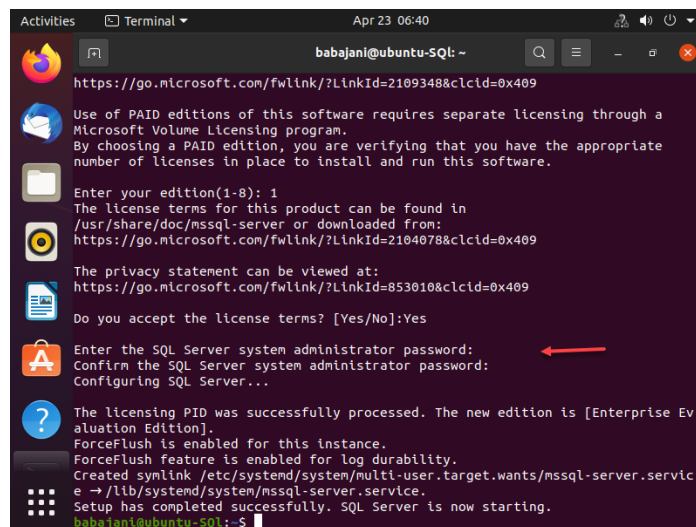
بعد از وارد کردن عدد یک باید کلمه‌ی Yes را به‌مانند شکل ۲-۶۰ برای تأیید لایسنس برنامه وارد کنید تا کار ادامه

پیدا کند.



شکل ۲-۶۰ اجرای سرویس MSSQL

در ادامه به مانند شکل ۲-۶۱ باید یک رمز عبور برای مدیریت SQL وارد کنید، بعد از آن سرویس SQL اجرا خواهد شد.



شکل ۲-۶۱ اجرای سرویس SQL

برای اینکه متوجه شویم، سرویس فعال شده است یا نه باید دستور زیر را وارد کنید، تا شکل ۲-۶۲ ظاهر شود.
systemctl status mssql-server.service

```

Activities Terminal Apr 23 06:43
babajani@ubuntu-SQL: ~
evaluation Edition].
ForceFlush is enabled for this instance.
ForceFlush feature is enabled for log durability.
Created symlink /etc/systemd/system/multi-user.target.wants/mssql-server.service
→ /lib/systemd/system/mssql-server.service.
Setup has completed successfully. SQL Server is now starting.
babajani@ubuntu-SQL:~$ systemctl status mssql-server.service
● mssql-server.service - Microsoft SQL Server Database Engine
   Loaded: loaded (/lib/systemd/system/mssql-server.service; enabled; vendor
   Active: active (running) since Fri 2021-04-23 06:40:50 PDT; 1min 59s ago
   Docs: https://docs.microsoft.com/en-us/sql/linux
   Main PID: 5864 (sqlservr)
     Tasks: 131
    Memory: 595.9M
   CGroup: /system.slice/mssql-server.service
           └─5864 /opt/mssql/bin/sqlservr
             └─5890 /opt/mssql/bin/sqlservr

Apr 23 06:40:56 ubuntu-SQL sqlservr[5890]: [158B blob data]
Apr 23 06:40:56 ubuntu-SQL sqlservr[5890]: [155B blob data]
Apr 23 06:40:56 ubuntu-SQL sqlservr[5890]: [61B blob data]
Apr 23 06:40:57 ubuntu-SQL sqlservr[5890]: [96B blob data]
Apr 23 06:40:57 ubuntu-SQL sqlservr[5890]: [66B blob data]
Apr 23 06:40:57 ubuntu-SQL sqlservr[5890]: [75B blob data]
Apr 23 06:40:57 ubuntu-SQL sqlservr[5890]: [96B blob data]
Apr 23 06:40:57 ubuntu-SQL sqlservr[5890]: [100B blob data]
Apr 23 06:40:57 ubuntu-SQL sqlservr[5890]: [71B blob data]
Apr 23 06:40:57 ubuntu-SQL sqlservr[5890]: [124B blob data]
lines 1-21/21 (END)

```

شکل ۶۲-۲ سرویس SQL

در ادامه‌ی کار باید ابزار مورد نیاز را بر روی سرور لینوکس برای کار با SQL نصب کنید؛ برای این کار دستورات زیر را به صورت متوالی اجرا کنید:

```

curl https://packages.microsoft.com/keys/microsoft.asc | sudo apt-key add -
curl https://packages.microsoft.com/config/ubuntu/19.10/prod.list >
/etc/apt/sources.list.d/mssql-release.list

sudo apt update

sudo ACCEPT_EULA=Y apt-get install mssql-tools unixodbc-dev

```

با اجرای دستورات بالا به مانند شکل ۶۳-۲ ابزار مربوط به SQL بر روی لینوکس نصب می‌شود.

```

root@ubuntu-SQL:~# curl https://packages.microsoft.com/keys/microsoft.asc | sudo apt-key add -
100 983 100 983 0 0 194 0 0:00:05 0:00:05 --:--:-- 200
OK
root@ubuntu-SQL:~# curl https://packages.microsoft.com/config/ubuntu/16.04/prod.list | sudo tee /etc/a
100 79 100 79 0 0 15 0 0:00:05 0:00:05 --:--:-- 16
deb [arch=amd64] https://packages.microsoft.com/ubuntu/16.04/prod xerial main
root@ubuntu-SQL:~# sudo apt-get update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [109 kB]
Hit:2 http://us.archive.ubuntu.com/ubuntu focal InRelease
Get:3 https://packages.microsoft.com/ubuntu/18.04/mssql-server-2019 bionic InRelease
Get:4 https://packages.microsoft.com/ubuntu/16.04/prod xerial InRelease [4,003 B]
Hit:5 https://packages.microsoft.com/ubuntu/19.10/prod eoan InRelease
Get:6 https://packages.microsoft.com/ubuntu/16.04/prod xerial/main amd64 Packages [204 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
0% [7 InRelease 57.0 kB/114 kB 50%]
Get:7 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Fetched 441 kB in 1min 16s (5,797 B/s)
Reading package lists... Done
root@ubuntu-SQL:~#

```

شکل ۶۳-۲ نصب ابزار SQL

در ادامه‌ی دستورات در قسمتی که در شکل ۶۴-۲ مشخص شده است باید Yes را وارد کنید تا ابزار MSSQL که حجمی حدود ۱۰۲ مگابایت دارد بر روی سرور لینوکس نصب شود.

```
The following additional packages will be installed:
autoconf automake autotools-dev binutils binutils-common
binutils-x86-64-linux-gnu gcc gcc-9 libasan5 libatomic1 libbinutils
libc-dev-bin libc6-dev libcrypt-dev libctf-nobfd0 libctf0 libgcc-9-dev
libitm1 liblsan0 libltdl-dev libodbc1 libquadmath0 libtool libtsan0
libubsan1 linux-libc-dev m4 manpages-dev msodbcsql17 odbcinst
odbcinstdebian2 unixodbc
Suggested packages:
autoconf-archive gnu-standards autoconf-doc gettext binutils-doc
gcc-multilib make flex bison gcc-doc gcc-9-multilib gcc-9-doc gcc-9-locales
glibc-doc libtool-doc unixodbc-bin gfortran | fortran95-compiler gcj-jdk
m4-doc
The following NEW packages will be installed:
autoconf automake autotools-dev binutils binutils-common
binutils-x86-64-linux-gnu gcc gcc-9 libasan5 libatomic1 libbinutils
libc-dev-bin libc6-dev libcrypt-dev libctf-nobfd0 libctf0 libgcc-9-dev
libitm1 liblsan0 libltdl-dev libodbc1 libquadmath0 libtool libtsan0
libubsan1 linux-libc-dev m4 manpages-dev msodbcsql17 mssql-tools odbcinst
odbcinstdebian2 unixodbc unixodbc-dev
0 upgraded, 34 newly installed, 0 to remove and 88 not upgraded.
Need to get 23.3 MB of archives.
After this operation, 102 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
0% [Connecting to kazoote.canonical.com]
```

شکل ۲-۶۴ نصب ابزار SQL

در ادامه‌ی کار، دو دستور زیر را وارد کنید تا کار نصب و تنظیم SQL به پایان برسد:

```
echo 'export PATH="$PATH:/opt/mssql-tools/bin"' >> ~/.bash_profile
```

```
echo 'export PATH="$PATH:/opt/mssql-tools/bin"' >> ~/.bashrc
source ~/.bashrc
```

بعد از نصب SQL و ابزارهای آن بر روی سرور لینوکس، حال می‌توانید بر روی SQL کار کنید، برای تست کارایی سرویس SQL دستور زیر را وارد کنید تا به آن متصل شوید:

```
sqlcmd -S 127.0.0.1 -U SA
```

همان‌طور که در شکل ۲-۶۵ مشاهده می‌کنید، بعد از اجرای دستور بالا از ما رمز عبوری را می‌خواهد که در هنگام نصب SQL وارد کردیم، یعنی شکل شماره‌ی ۲-۶۱؛ بعد از وارد کردن رمز عبور حالا وارد خط فرمان SQL شدیم و می‌توانیم دستورات مورد نظر را اجرا کنیم. برای تست دستور زیر را اجرا می‌کنیم:

```
select name from sys.databases;
go
```

با اجرای دستور بالا، لیست دیتابیس‌های اصلی و مربوط به تنظیمات خود SQL را مشاهده می‌کنید که در مورد این دیتابیس‌ها در فصل سوم به صورت کامل صحبت خواهیم کرد.

```
babajani@ubuntu-SQL: ~
root@ubuntu-SQL:~#
root@ubuntu-SQL:~#
root@ubuntu-SQL:~#
root@ubuntu-SQL:~# sqlcmd -S 127.0.0.1 -U SA
Password:
1> Select name from sys.databases
2> go
name
-----
master
tempdb
model
msdb
(4 rows affected)
```

شکل ۲-۶۵ لیست دیتابیس SQL

فصل سوم

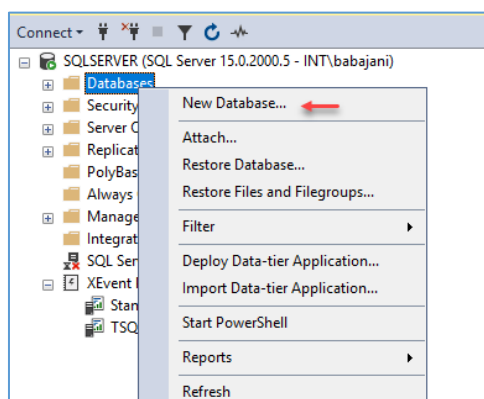
آشنایی با پایگاه داده

۳-۱ ایجاد پایگاه داده و کار با آن

۳-۱-۱ ایجاد پایگاه داده

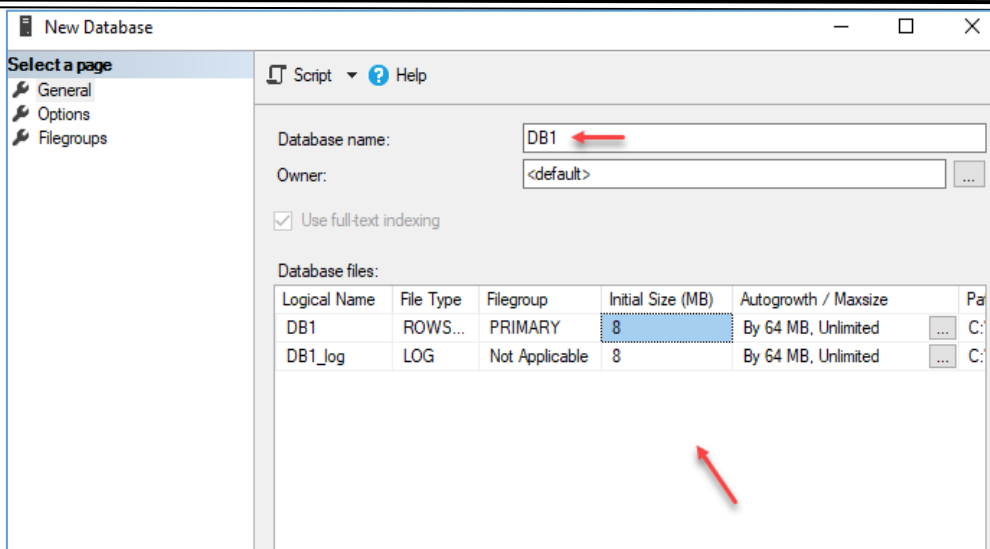
در ادامه‌ی کار می‌خواهیم برای شروع، یک دیتابیس ایجاد و بر روی آن کارهای مختلفی را انجام می‌دهیم، ایجاد دیتابیس را می‌توانیم به دو صورت انجام دهیم، یکی به صورت گرافیکی و از طریق نرم‌افزار SSMS و یکی از طریق کد T-SQL که در ادامه هر دو روش را بررسی خواهیم کرد.

برای شروع به‌مانند شکل ۳-۱ وارد SSMS شوید و بر روی Databases کلیک راست کنید و بر روی New Database کلیک کنید.



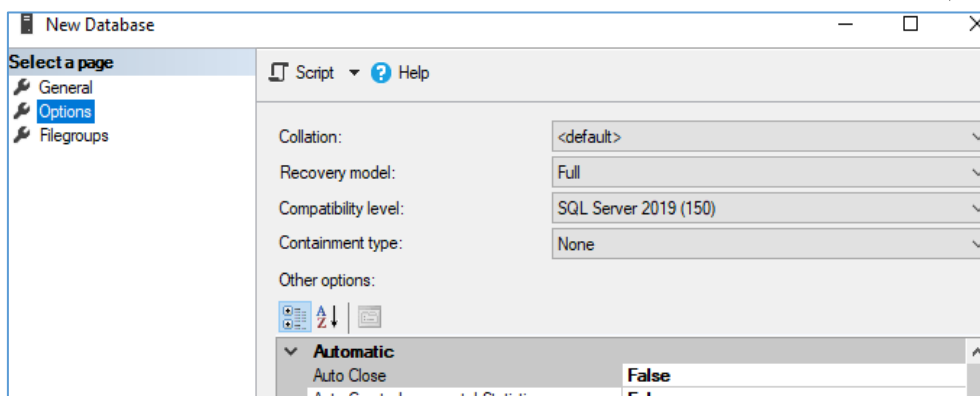
شکل ۳-۱ / ایجاد دیتابیس

در شکل ۳-۲ باید نام دیتابیس مورد نظر خود را در قسمت Database Name وارد کنید و در قسمت Owner باید یک کاربر را به‌عنوان صاحب این دیتابیس انتخاب کنید که اگر انتخاب نکنید، همین کاربری که با آن در حال ایجاد دیتابیس هستید به‌عنوان صاحب آن در نظر گرفته خواهد شد؛ بعد از وارد کردن نام در قسمت Database name، دو گزینه با نام دیتابیس شما ایجاد می‌شود که اولی (DB1) نام دیتابیس و دومی (DB1_Log) دیتابیس‌ی برای ثبت Log است، این دو مکمل هم هستند و حتماً برای اجرا به همدیگر نیاز دارند.



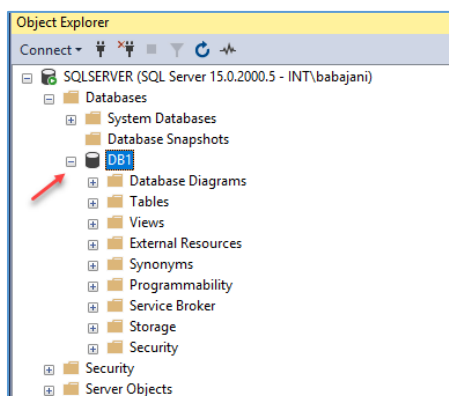
شکل ۳-۲ / ایجاد دیتابیس

در قسمت Option که در شکل ۳-۳ مشخص شده است، گزینه‌هایی وجود دارد، گزینه‌ی Collation به روشی برای مقایسه‌ی اطلاعات با هم اشاره دارد؛ قسمت Recovery model که مربوط به عملیات پشتیبان‌گیری و بازیابی اطلاعات است که در موقع مناسب توضیح خواهیم داد. در قسمت Compatibility level هم می‌توانید مشخص کنید که این دیتابیس با کدام نسخه از SQL سازگاری داشته باشد.



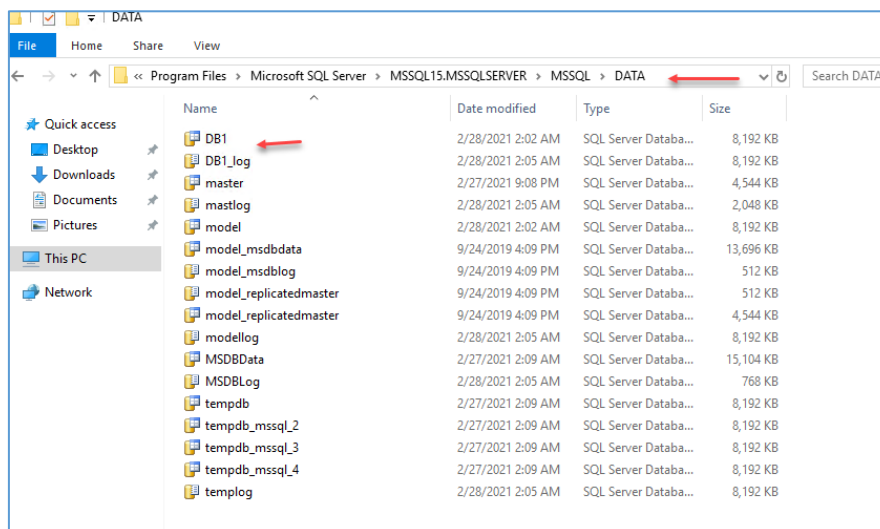
شکل ۳-۳ / ایجاد دیتابیس

بعد از ایجاد دیتابیس می‌توانید به‌مانند شکل ۳-۴ در قسمت Databases، این دیتابیس جدید را مشاهده کنید.



شکل ۳-۴

اگر وارد مسیر مورد نظر در شکل ۳-۵ شوید، فایل دیتابیس تولید شده را مشاهده می‌کنید که همان دو فایل است که اشاره کردیم.



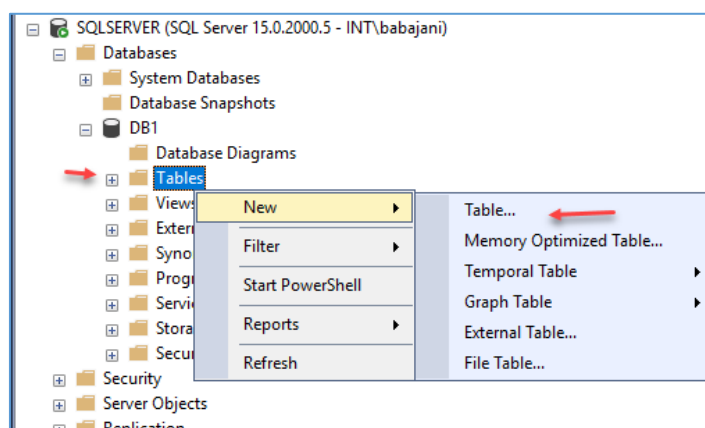
شکل ۳-۵ لیست دیتابیس

نکته:

سعی کنید در اول کار زمانی که می‌خواهید دیتابیس خود را ایجاد کنید، آدرس آن را در یک مکان مطمئن و با حجم بالا قرار دهید تا در آینده با اضافه شدن داده، حجم و اطلاعات شما حفظ شود.

۲-۱-۳ ایجاد جدول در دیتابیس

برای ایجاد جدول در دیتابیس مورد نظر باید به‌مانند شکل ۲-۲۵ بر روی پوشه‌ی Tables کلیک راست کنید و گزینه‌ی Table را از قسمت New انتخاب کنید.



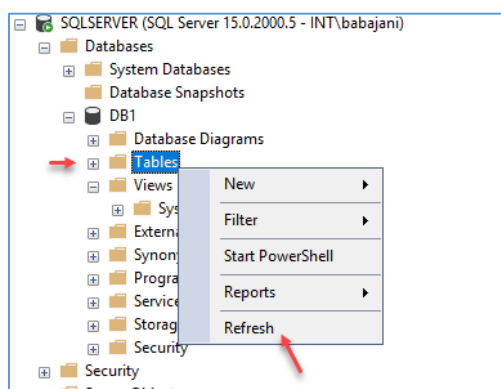
شکل ۳-۶ ایجاد جدول

در شکل ۷-۳ باید در قسمت Column Name، نام ستون خود را وارد کنید که چهار مورد وارد شده است و در قسمت Data Type باید نوع ستون مورد نظر را از نظر عددی، حرفی و ... انتخاب کنید، برای ذخیره کردن این جدول باید بر روی عنوان جدول کلیک راست کنید و گزینهی Save را انتخاب کنید.

Column Name	Data Type	Allow Nulls
[First Name]	nchar(10)	<input checked="" type="checkbox"/>
[Last Name]	nchar(10)	<input checked="" type="checkbox"/>
City	nchar(10)	<input checked="" type="checkbox"/>
Age	int	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

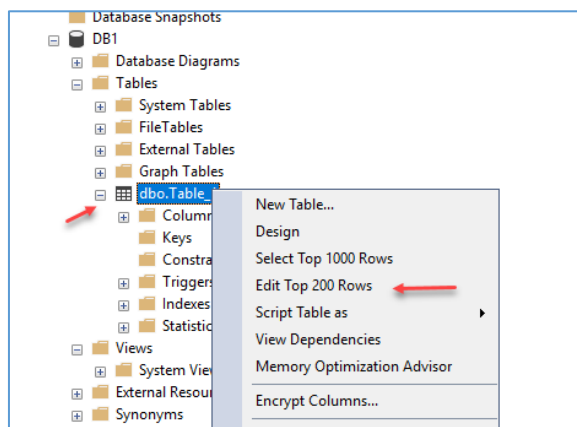
شکل ۷-۳/ ایجاد جدول

بعد از ایجاد جدول باید جدول مورد نظر در قسمت Table نمایش داده شود، اگر این چنین نشد باید به مانند شکل ۸-۳ بر روی Tables کلیک راست کنید و گزینهی Refresh را انتخاب کنید تا جدول مورد نظر مشخص شود.



شکل ۸-۳/ ایجاد جدول

بعد از ایجاد جدول می‌خواهیم اطلاعاتی در آن وارد کنیم؛ برای این کار در شکل ۹-۳ بر روی dbo.Table_1 کلیک راست کنید و گزینهی Edit Top 200 Rows را انتخاب کنید.



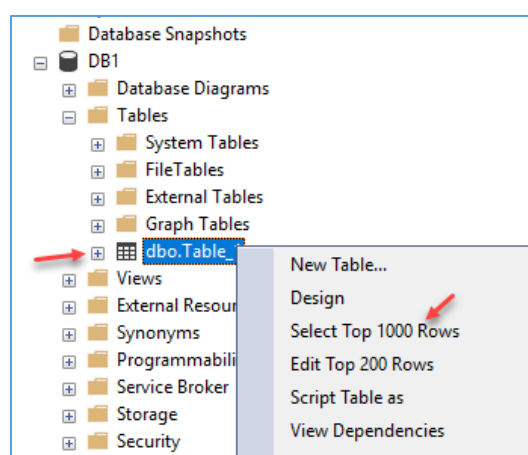
شکل ۳-۹ ورود اطلاعات در جدول

در شکل ۳-۱۰ اطلاعاتی را وارد کنید و صفحه‌ی مورد نظر را ببندید.

First Name	Last Name	City	Aqe
ali	modares	babol	25
reza	hazir	dezfol	32
azadeh	kordi	amol	33
meysam	moham	shiraz	35

شکل ۳-۱۰ ورود اطلاعات در جدول

برای اینکه اطلاعات موجود در جدول را مشاهده کنید، می‌توانید به‌مانند شکل ۳-۱۱ بر روی جدول مورد نظر کلیک راست کنید و گزینه‌ی Select Top 1000 Rows را انتخاب کنید.



شکل ۳-۱۱ نمایش اطلاعات جدول

همان‌طور که در شکل ۳-۱۲ مشاهده می‌کنید، اطلاعات جدول توسط یک اسکریپت در خروجی به نمایش گذاشته شده است که در ادامه، در مورد کد مورد نظر توضیحاتی خواهیم داد.

```

SQLQuery3.sql - SQ...(INT\babajani (59))
/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [First Name]
, [Last Name]
, [City]
, [Age]
FROM [DB1].[dbo].[Table_1]

```

First Name	Last Name	City	Age
1 ali	modares	babol	25
2 reza	hazir	dezfol	32
3 azadeh	kordi	amol	33
4 meysam	moham	shiraz	35

شکل ۳-۱۲ نمایش اطلاعات جدول

بعد از ایجاد اولیه‌ی جدول و وارد کردن مقادیری در آن، می‌خواهیم در مورد نوع داده‌ی ورودی صحبت کنیم که بسیار مهم است.

۳-۱-۳ انواع Data Type در جداول

زمانی که می‌خواهیم یک Filed جدید در جدول، مانند: نام، نام خانوادگی، شماره‌ی دانشجویی و... ایجاد کنیم باید از انواع مختلف داده‌ای استفاده کنیم، مثلاً برای نوشته‌ای مانند نام باید از نوع داده‌ی Char/nchar استفاده کنید و به همین ترتیب از انواع مختلف دیگر می‌توان استفاده کرد. در زیر این نوع داده‌ها را بررسی می‌کنیم:

انواع داده‌های رشته‌ای

جدول ۳-۱ انواع داده‌های رشته‌ای

نوع داده	اندازه داده	توضیحات
CHAR(size)	حداکثر اندازه ۸۰۰۰ کاراکتر.	طول داده ثابت است.
VARCHAR(size) or VARCHAR(max)	حداکثر اندازه ۸۰۰۰ کاراکتر با افزایش حجم.	طول داده متغیر است و توانایی افزایش حجم تا ۲ گیگابایت را داراست.
TEXT	حداکثر اندازه ۲ گیگابایت.	داده‌های غیر یونیک با طول متغیر.
NCHAR(size)	حداکثر اندازه ۴۰۰۰ کاراکتر.	داده‌های غیر یونیک با طول ثابت.
NVARCHAR(size) or NVARCHAR(max)	حداکثر اندازه ۴۰۰۰ کاراکتر با افزایش حجم.	طول داده متغیر است و توانایی افزایش حجم تا ۲ گیگابایت را داراست.
NTEXT	حداکثر اندازه ۱,۰۷۳,۷۴۱,۸۲۳ bytes است.	داده‌های غیر یونیک با طول متغیر.
BINARY(size)	حداکثر اندازه ۸۰۰۰ کاراکتر.	طول داده ثابت است.
VARBINARY(size) or VARBINARY(max)	حداکثر اندازه ۸۰۰۰ کاراکتر با افزایش حجم.	طول داده متغیر است و توانایی افزایش حجم تا ۲ گیگابایت را داراست.

داده‌های غیر یونیک با طول متغیر.	حداکثر اندازه ۲ گیگابایت.	IMAGE
----------------------------------	---------------------------	--------------

انواع داده‌های عددی

جدول ۳-۲ داده‌های عددی

توضیحات	حداکثر اندازه	نوع داده عددی
	عدد صحیح است که می‌تواند ۰ یا ۱ یا NULL باشد.	BIT
	از ۰ تا ۲۵۵	TINYINT
	از -۳۲۷۶۸ تا ۳۲۷۶۷	SMALLINT
	از -۲,۱۴۷,۴۸۳,۶۴۸ تا ۲,۱۴۷,۴۸۳,۶۴۷	INT
	از -۹,۲۲۳,۳۷۲,۰۳۶,۸۵۴,۷۷۵,۸۰۸ تا ۹,۲۲۳,۳۷۲,۰۳۶,۸۵۴,۷۷۵,۸۰۷	BIGINT
M تعداد ارقام و d تعداد ارقام اعشار بعد از ممیز است.	M اگر مشخص نشده باشد به صورت پیش فرض ۱۸ در نظر گرفته می‌شود.	DECIMAL(m,d)
	d اگر مشخص نشده باشد، به صورت پیش فرض ۰ در نظر گرفته می‌شود.	
M تعداد ارقام و d تعداد ارقام اعشار بعد از ممیز.	M اگر مشخص نشده باشد به صورت پیش فرض ۱۸ در نظر گرفته می‌شود.	DEC(m,d)

	<p>d اگر مشخص نشده باشد، به صورت پیش فرض</p> <p>در نظر گرفته می شود.</p>	
<p>M تعداد ارقام و d تعداد ارقام اعشار بعد از ممیز.</p>	<p>M اگر مشخص نشده باشد به صورت پیش فرض</p> <p>۱۸ در نظر گرفته می شود.</p>	<p>NUMERIC(m,d)</p>
	<p>d اگر مشخص نشده باشد، به صورت پیش فرض</p> <p>در نظر گرفته می شود.</p>	
<p>که در آن n تعداد تعداد بیت برای ذخیره به صورت نماد علمی است.</p>	<p>اعداد شناور</p>	<p>FLOAT(n)</p>
	<p>N به صورت پیش فرض ۵۳ در نظر گرفته خواهد شد.</p>	
	<p>از - ۲۱۴,۷۴۸.۳۶۴۸ تا ۲۱۴,۷۴۸.۳۶۴۷</p>	<p>SMALLMONEY</p>
	<p>از - ۹۲۲,۳۳۷,۲۰۳,۶۸۵,۴۷۷.۵۸۰۸ تا ۹۲۲,۳۳۷,۲۰۳,۶۸۵,۴۷۷.۵۸۰۷</p>	<p>MONEY</p>

انواع داده های زمان و تاریخ

جدول ۳-۳ داده های زمان و تاریخ

نوع داده	حداکثر اندازه کاراکتر	توضیحات
DATE	دامنه ی اعداد از '0001-01-01' تا '۹۹۹۹-۱۲-۳۱'	فرمت نمایش 'YYYY-MM-DD'

فرمت نمایش-YYYY-MM DD hh:mm:ss[.mmm]'	دامنه‌ی اعداد از '۰۰:۰۰:۰۰ ۰۱-۰۱-۱۷۵۳' تا '9999-12-31 23:59:59' دامنه‌ی زمان از '۰۰:۰۰:۰۰' تا '۲۳:۵۹:۵۹:۹۹۷'	DATETIME
فرمت نمایش-YYYY-MM DD hh:mm:ss[.fractional seconds]'	دامنه‌ی تاریخ از '0001-01-01' تا '۹۹۹۹- ۱۲-۳۱' دامنه‌ی زمان از '00:00:00' تا '23:59:59:9999999'.	DATETIME2(fractional seconds precision)
فرمت نمایش-YYYY-MM DD hh:mm:ss'	دامنه‌ی تاریخ از '1900-01-01' تا '۲۰۷۹- ۰۶-۰۶' دامنه‌ی زمان از '۰۰:۰۰:۰۰' تا '۲۳:۵۹:۵۹'	SMALLDATETIME
فرمت نمایش-YYYY-MM DD hh:mm:ss[.nnnnnnn]'	دامنه‌ی زمان از '۰۰:۰۰:۰۰.۰۰۰۰۰۰۰۰' تا '23:59:59.9999999'	TIME
فرمت نمایش-YYYY-MM DD hh:mm:ss[.nnnnnnn] [{+ -}hh:mm]	دامنه‌ی تاریخ از '0001-01-01' تا '۹۹۹۹- ۱۲-۳۱' دامنه‌ی زمان از '۰۰:۰۰:۰۰' تا '23:59:59:9999999'. دامنه‌ی منطقه از '۱۴:۰۰+' تا '۱۴:۰۰-'	DATETIMEOFFSET(fractional seconds precision)

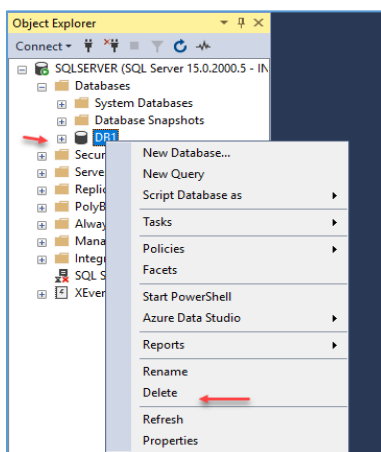
در جدول‌هایی که بررسی کردیم، تمام داده‌های رشته‌ای، عددی، زمان و تاریخ تعریف شده است و توضیحات مربوط به آن داده شده است.

Data Type های دیگری نیز وجود دارد که در نسخه‌های جدید SQL می‌توانید از آنها استفاده کنید:

نوع داده	توضیحات
sql_variant	حداکثر ۸۰۰۰ بایت داده از انواع مختلف داده، به جز متن، ntext و زمان‌سنج ذخیره می‌کند.
uniqueidentifier	شناسه منحصر به فرد جهانی (GUID) را ذخیره می‌کند.
xml	داده‌های قالب‌بندی شده XML را ذخیره می‌کند، حداکثر ۲ گیگابایت.
cursor	این نوع داده برای ذخیره‌ی متغیرها و یا پارامترهای OUTPUT مربوط به store procedure کاربرد دارد.
table	مجموعه‌ای از نتایج را برای پردازش بعدی ذخیره می‌کند.

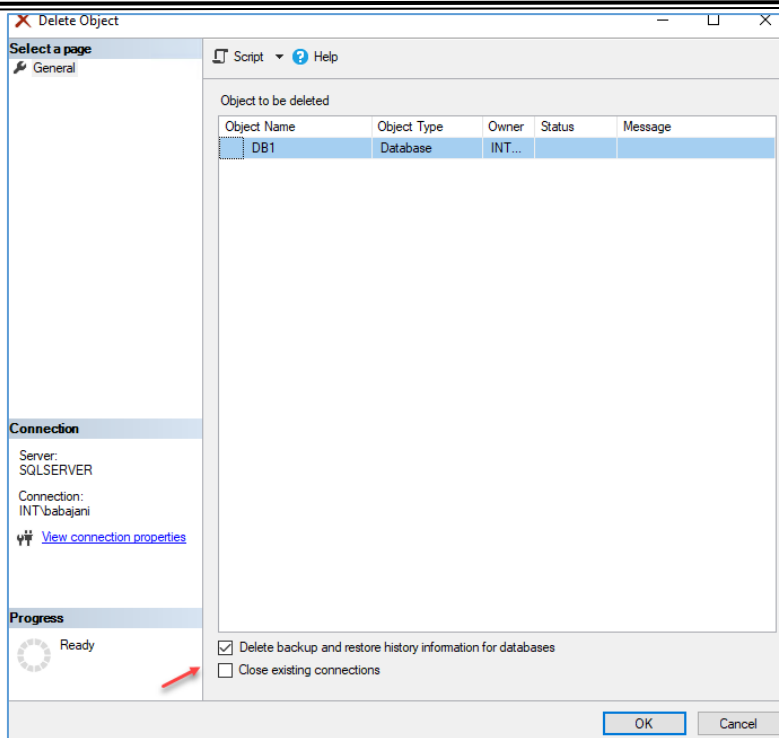
۳-۱-۴ حذف دیتابیس در SQL Server

بعد از اینکه توانستیم یک دیتابیس را ایجاد کنیم نوبت به حذف آن می‌رسد، برای حذف دیتابیس کافی است به‌مانند شکل ۳-۱۳ بر روی دیتابیس مورد نظر کلیک راست کنید و گزینه‌ی Delete را انتخاب کنید.



شکل ۳-۱۳ حذف دیتابیس

در شکل ۳-۱۴ نام دیتابیس DB1 در لیست مشخص شده است و اگر بر روی Delete کلیک کنید، دیتابیس حذف خواهد شد، اما دو گزینه در زیر این صفحه وجود دارد.



شکل ۳-۱۴ حذف دیتابیس

گزینه‌ی `Delete backup and restore history information for databases`: این گزینه به صورت پیش فرض فعال است و تاریخ و اطلاعات این دیتابیس را از دیتابیس `MSDB` حذف می‌کند. گزینه‌ی `Close existing connections`، به صورت پیش فرض فعال نیست و اگر آن را فعال کنید، قبل از حذف دیتابیس، اول بررسی می‌کند که این دیتابیس در جایی دیگر باز شده و یا در حال استفاده است، ابتدا آن ارتباط را می‌بندد و بعد اقدام به حذف می‌کند، سعی کنید این گزینه را فعال نکنید تا حداقل در زمان حذف مشخص شود که در جایی دیگر در حال استفاده است.

۳-۱-۵ تعریف پرس و جو یا Query در SQL

یکی از اصلی ترین موضوعاتی که در SQL باید بررسی شود بحث پرس و جو و یا همان Query است، هر پرس و جو می‌تواند شامل چندین دستور باشد که در زیر آنها را بررسی می‌کنیم.

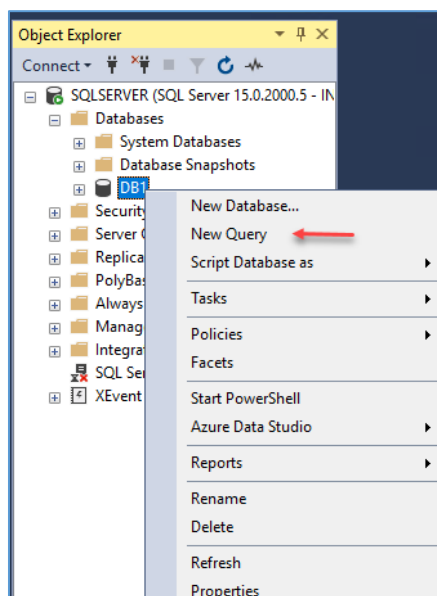
۳-۱-۵-۱ بررسی دستور SELECT

برای اینکه در SQL اطلاعات را از جدول‌های بازایی به قولی فچ کنیم از دستور `SELECT` استفاده می‌کنیم، در زیر نمونه‌ای از این دستور را مشاهده می‌کنید:

```
SELECT column1, column2, ...
FROM table_name;
```

در دستور بالا، دستور SELECT را با حروف بزرگ می‌نویسیم و بعد از آن باید ستون (*column1, column2*) مربوط به جدولی را که می‌خواهید داده‌ها را از آن استخراج کنید می‌نویسید، در سطر دوم با دستور FROM، اسم جدول را به جای *table_name* می‌نویسید.

برای اینکه این دستورات را در سرور SQL تست بگیریم باید به‌مانند شکل ۳-۱۵ وارد Management شوید و بر روی دیتابیس مورد نظر خود کلیک راست کنید و گزینه‌ی New Query را انتخاب کنید.



شکل ۳-۱۵ ایجاد Query

برای شروع کار به جدول ۳-۵ توجه کنید، می‌خواهیم با استفاده از این جدول عملیات مختلف خود را انجام دهیم و داده‌ها را فراخوانی کنیم.

جدول ۳-۵ Customers

CustomerID	Customer Name	ContactName	Address	City	PostalCode	Country
1	Maria	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana	Ana Trujillo	Avda. de la Constitución 2222	México	05021	Mexico
3	Antonio	Antonio Moreno	Mataderos 2312	México	05023	Mexico
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden

برای ایجاد این جدول از طریق SSMS یک جدول ایجاد کنید و آن را با اسم Customers ذخیره کنید و بعد از ایجاد، اطلاعات جدول ۳-۵ را درون آن وارد کنید تا در ادامه بتوانید از آنها استفاده کنید، این موضوع را در شکل ۳-۱۶ مشاهده می‌کنید.

Column Name	Data Type	Allow Nulls
CustomerID	int	<input checked="" type="checkbox"/>
[Customer Name]	nchar(15)	<input checked="" type="checkbox"/>
[Contact Name]	nchar(15)	<input checked="" type="checkbox"/>
Address	varchar(50)	<input type="checkbox"/>
City	nchar(10)	<input checked="" type="checkbox"/>
postalcode	nchar(10)	<input checked="" type="checkbox"/>
country	nchar(10)	<input checked="" type="checkbox"/>

شکل ۱۶-۳ ویرایش جدول Customers

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	Maria	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana	Ana Trujillo	Avda. de la Con...	México	05021	Mexico
3	Antonio	Antonio Moreno	Mataderos 2312	México	05023	Mexico
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden
	NULL	NULL	NULL	NULL	NULL	NULL

شکل ۱۷-۳ ورود اطلاعات در جدول Customers

مثال اول - انتخاب ستون

```
SELECT [Customer Name] , City FROM Customers
```

در دستور بالا، ستون Customer Name و City از جدول Customers انتخاب شده است، توجه داشته باشید علامت [] را بهتر است زمانی قرار دهید که یک جمله‌ی دو حرفی یا بیشتر دارید، مثلاً در مثال بالا اگر city بود، نیاز به علامت [] نبود، اما چون Customer Name از دو قسمت با فاصله تعریف شده است باید حتماً علامت [] را قرار دهید. به‌مانند شکل ۱۸-۳ دستور مورد نظر را بنویسید و برای دریافت خروجی باید کلید F5 را فشار دهید، این دستور دو ستون Customer Name و City را در خروجی نمایش می‌دهد.

```
SQLQuery5.sql - SQ...(INT,babajani (53))*
/***** Script for SelectTopNRows command from SSM
SELECT [Customer Name] , City FROM Customers
```

	Customer Name	City
1	Maria	Berlin
2	Ana	México
3	Antonio	México
4	Thomas	London
5	Christina	Luleå

شکل ۱۸-۳ خروجی دستور SELECT

مثال دوم - نمایش تمام اطلاعات جدول

برای اینکه بتوانید همه‌ی اطلاعات جدول مورد نظر خود را مشاهده کنید باید از دستور زیر استفاده کنید:

```
SELECT * FROM Customers
```

همان‌طور که در شکل ۳-۱۹ مشاهده می‌کنید، تمام اطلاعات جدول در خروجی نمایش داده شده است.

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	1	Maria	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	2	Ana	Ana Trujillo	Avda. de la Constitución 2222	México	05021	Mexico
3	3	Antonio	Antonio Moreno	Mataderos 2312	México	05023	Mexico
4	4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden

شکل ۳-۱۹ نمایش جدول Customers

اگر بخواهیم در یک جدول، تنها اطلاعات متمایز شده و نه تکراری را نمایش دهیم باید از دستور زیر استفاده کنیم:

```
SELECT DISTINCT column1, column2, ...  
FROM table_name;
```

در دستور بالا به‌مانند قبل باید نام ستون‌ها را جلوی دستور SELECT بنویسید، اما تنها باید بعد از دستور SELECT از DISTINCT استفاده کنید تا اطلاعات شبیه به هم را تنها یک‌بار در خروجی نمایش دهد.

مثال سوم – نمایش اطلاعات شهرهای مشابه

دستورات زیر را اجرا کنید:

```
SELECT Country FROM Customers;  
SELECT DISTINCT Country FROM Customers;
```

خروجی دستورات بالا را در شکل ۳-۲۰ مشاهده می‌کنید، دستور اول تمام شهرها را به شما نمایش داده است، اما در دستور دوم تنها یکی از شهرهای Mexico را نمایش داده است و این مورد می‌تواند در نمایش بهتر اطلاعات کمک کند.

	Country
1	Germany
2	Mexico
3	Mexico
4	UK
5	Sweden

	Country
1	Germany
2	Mexico
3	Sweden
4	UK

شکل ۳-۲۰ نمایش جدول

۲-۵-۱-۳ بررسی دستور Insert

این دستور برای وارد کردن اطلاعات در جدول کاربرد دارد و می‌توانید با استفاده از این دستور، اطلاعات خود را وارد جدول کنید.

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

در دستور بالا، ابتدا باید INSERT INTO را بنویسیم و بعد از آن، نام جدول را به جای table_name بنویسیم، سپس باید در پرانتز مورد نظر نام ستون‌هایی که قرار است اطلاعات در آن وارد بشود را بنویسیم. روش دیگری هم برای ورود اطلاعات به صورت زیر وجود دارد:

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

در دستور بالا، ورود اطلاعات به ترتیب ستون‌های موجود انجام می‌شود. مثال اول – ورود اطلاعات با استفاده از نام ستون

```
INSERT INTO Customers (CustomerID, "Customer Name", "Contact Name", Address, City,
PostalCode, Country)
VALUES ('6', 'Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway')
;
SELECT * FROM Customers;
```

در دستور بالا، ابتدا نام ستون‌ها ذکر شده است و بعد از آن، اطلاعاتی که قرار است وارد جدول شود را در قسمت VALUES وارد می‌کنیم.

به این نکته توجه کنید که ستون‌هایی که دو قسمتی هستند، حتماً باید بین دو تا علامت " " قرار بگیرند تا در خروجی با خطا مواجه نشوید.

در آخر، دستور SELECT * FROM Customers; را برای نمایش اطلاعات جدول وارد کردیم که در شکل ۳-۲۱ خروجی نهایی را مشاهده می‌کنید که یک سطر جدید به جدول اضافه شده است.

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	Maria	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana	Ana Trujillo	Avda. de la Constitución 2222	México	05021	Mexico
3	Antonio	Antonio Moreno	Mataderos 2312	México	05023	Mexico
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway

شکل ۳-۲۱ ورود اطلاعات در جدول

مثال دوم – ورود اطلاعات به ترتیب ستون

دستور زیر را اجرا کنید:

```
INSERT INTO Customers
```

```
VALUES ('7','Farshid','Babajani','Seied Khandan','Tehran','8843','Iran');
SELECT * FROM Customers;
```

در دستور بالا، اطلاعات ورودی به ترتیب ستون‌های جدول قرار می‌گیرند و باید در ورود اطلاعات دقت کنید، اگر به شکل ۳-۲۲ توجه کنید خروجی مشخص شده است.

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	Maria	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana	Ana Trujillo	Avda. de la Constitución 2222	México	05021	Mexico
3	Antonio	Antonio Moreno	Mataderos 2312	México	05023	Mexico
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway
7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran

شکل ۳-۲۲ ورود اطلاعات

۳-۵-۱-۳ بررسی دستور Update

با استفاده از این دستور می‌توانید اطلاعات موجود در هر ستون را تغییر دهید؛ برای این کار باید از دستور زیر استفاده کنید:

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

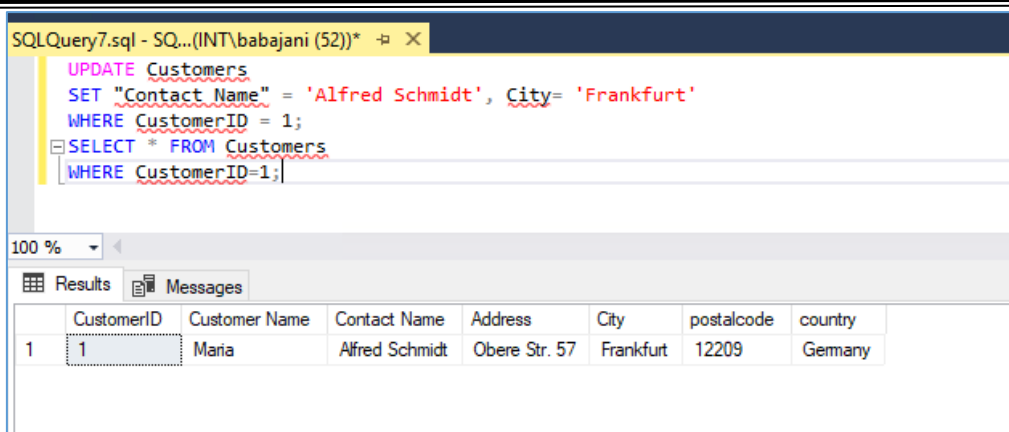
در دستور بالا، ابتدا باید نام جدول را بعد از دستور UPDATE وارد کنید. در قسمت SET باید مشخص کنید که چه گزینه‌هایی باید تغییر کند و در آخر نیز باید از دستور شرطی WHERE استفاده کنید تا مشخص شود که به کدام سطر اشاره می‌کنید.

مثال اول – آپدیت کردن اطلاعات جدول

```
UPDATE Customers
SET "Contact Name" = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;
SELECT * FROM Customers
WHERE CustomerID=1;
```

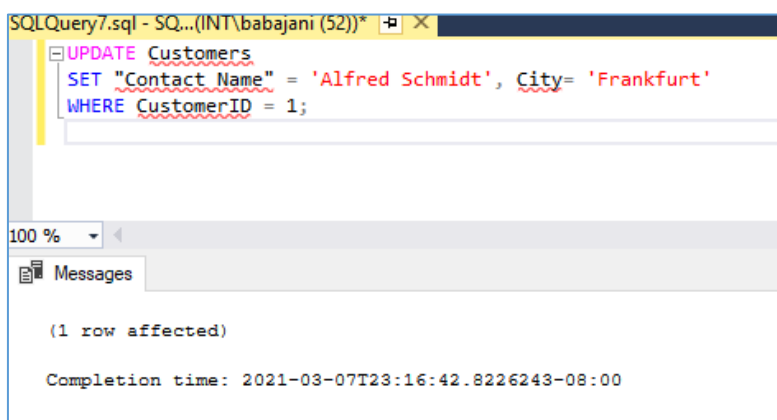
در دستور بالا، جدول Customers انتخاب می‌شود و در قسمت SET ستون‌ها به همراه مقدار آن مشخص می‌شود و در قسمت شرطی WHERE باید شرط مورد نظر را وارد کنید که در اینجا از شماره‌ی CustomerID استفاده کرده است؛ توجه داشته باشید دو خط آخر برای نمایش سطر اول جدول به کار برده شده است.

اگر به شکل ۳-۲۳ توجه کنید، سطر مورد نظر آپدیت شده است و در خروجی به نمایش گذاشته شده است.



شکل ۳-۲۳ آپدیت اطلاعات

اگر دو خط آخر دستور بالا را وارد نکنید، خروجی مانند شکل ۳-۲۴ خواهد شد.



شکل ۳-۲۴ آپدیت اطلاعات

کارهای زیادی می‌توان با دستور UPDATE انجام داد، مثلاً می‌توانید با دستور زیر نام ContactName مربوط به شهرهای Mexico را با استفاده از شرط تغییر دهید:

```

UPDATE Customers
SET "Contact Name"='Juan'
WHERE Country='Mexico';

```

نکته‌ی مهم: اگر چنانچه شرط WHERE را در دستور UPDATE قرار ندهید، تمام ستون‌های انتخاب‌شده آپدیت خواهد شد؛ در دستور زیر تمام ContactName ها به Juan تغییر خواهند کرد.

```

UPDATE Customers
SET ContactName='Juan';

```

۳-۱-۵-۱ بررسی دستور DELETE

این دستور همان‌طور که از نامش پیداست، برای حذف اطلاعات جدول‌ها کاربرد دارد که شکل کلی آن به‌صورت زیر است:

```
DELETE FROM table_name WHERE condition;
```

در دستور بالا به‌جای table_name باید نام جدول و به‌جای condition، شرط مورد نظر خود را وارد کنید.

مثال اول - حذف اطلاعات جدول

```

DELETE FROM Customers WHERE "Customer Name"='Farshid';
SELECT * FROM Customers

```

در دستور بالا، جدول Customers انتخاب شده است و در شرط آن اعلام شده است که CustomersName مشخص شده باید حذف شود که در شکل ۳-۲۵ قبل از اینکه دستور مورد نظر را اجرا کنید، نام Farshid را در قسمت ۷ مشاهده می کنید، اما بعد از اجرای دستور در شکل ۳-۲۶ سطر مورد نظر حذف شد.

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany
2	2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico
3	3	Antonio	Juan	Mataderos 2312	México	05023	Mexico
4	4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden
6	6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway
7	7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran

شکل ۳-۲۵ حذف اطلاعات

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany
2	2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico
3	3	Antonio	Juan	Mataderos 2312	México	05023	Mexico
4	4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden
6	6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway

شکل ۳-۲۶ حذف اطلاعات

برای حذف اطلاعات کامل جدول باید از دستور زیر استفاده کنید:

```
DELETE FROM table_name;
```

در دستور بالا باید به جای table_name نام جدول خود را وارد کنید.

۳-۱-۵-۵ بررسی دستور WHERE

یک دستور شرطی که در قسمت دستور UPDATE مقداری بررسی کردیم، شکل کلی دستور به صورت زیر خواهد بود

```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

در دستور بالا، SELECT نوشتیم که ستونهای column1, column2 را در خروجی نمایش دهد، اما در آخر از دستور WHERE استفاده شده و شرطی بر خروجی قرار داده است.

مثال یک - خروجی دستور SELECT با استفاده از شرط

```
SELECT * FROM Customers
```

```
WHERE Country='Mexico';
```

در دستور بالا، اطلاعات جدول Customers در خروجی نمایش داده می‌شود، به شرطی که ستون Country برابر Mexico باشد، خروجی دستور مورد نظر را در شکل ۳-۲۷ مشاهده می‌کنید.

The screenshot shows a SQL query window with the following text:

```
SELECT * FROM Customers
WHERE Country='Mexico';
```

Below the query window, the 'Results' tab is active, displaying a table with the following data:

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico
2	3	Antonio	Juan	Mataderos 2312	México	05023	Mexico

شکل ۳-۲۷ بررسی دستور WHERE

نکته:

در دستور `WHERE Country='Mexico'`، کلمه `Mexico` بین نقل قول قرار گرفته است، اما اگر به جای `Mexico`، از عدد استفاده می‌کردید، دیگر نیاز به نقل قول نبود، پس به این نکته توجه کنید.

از عملگرهای جدول ۳-۶ می‌توان در بند WHERE استفاده کرد.

جدول ۳-۶ عملگر

توضیحات	عملگر
مساوی	=
بزرگ‌تر از	>
کوچک‌تر از	<
بزرگ‌تر مساوی	>=
کوچک‌تر مساوی	<=
نا برابر، توجه داشته باشید در بعضی از نسخه‌های SQL این علامت شاید به صورت != باشد.	<>
بین یک محدوده‌ی مشخص	BETWEEN
جستجوی یک الگو	LIKE
برای تعیین چندین مقدار ممکن برای یک ستون	IN

مثال دوم – استفاده از عملگر

اگر بخواهیم از این عملگرها در دستور شرطی استفاده کنید، می‌توانید از دستور زیر استفاده کنید:

```
SELECT * FROM Customers
WHERE CustomerID <> 1;
```

```
SELECT * FROM Customers
WHERE CustomerID<=1;
```

در دستور بالا از یک عملگر نابرابر و یک کوچک‌تر مساوی استفاده شده است که خروجی این دستور را در شکل ۲۸-۳ مشاهده می‌کنید؛ در قسمت اول، سطر یک نمایش داده نشده و در قسمت دوم تنها سطر یک نمایش داده شده است.

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany

شکل ۲۸-۳ بررسی دستور WHERE

۶-۵-۱-۳ بررسی دستور AND، OR و NOT

همان‌طور که در درس قبلی خواندید، دستور WHERE یک دستور شرطی است، شما می‌توانید با استفاده از دستورات AND، OR و NOT یک ترکیب جدید در شرط ایجاد کنید که در این قسمت به طور کامل آن را بررسی می‌کنیم.

دستور AND

فرم کلی دستور AND به صورت زیر است:

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 AND condition2 AND condition3 ... ;
```

در دستور بالا با استفاده از SELECT، نام ستون‌ها را انتخاب می‌کنیم و در قسمت WHERE باید شرط خود را با استفاده از AND وارد کنید، منظور از AND این است که یک شرط حتماً باید این چند مورد که با AND تعریف می‌شود را داشته باشد.

مثال اول – AND

```
SELECT * FROM Customers
WHERE Country='Iran' AND City='Tehran';
```

در دستور بالا، خروجی اطلاعات جدول Customers در خط اول نمایش داده خواهد شد، اما در خط دوم شرطی قرار داده که حتماً باید نام Country و City به ترتیب برابر Farshid و Tehran باشد، اگر به شکل ۲۹-۳ توجه کنید خروجی به درستی نمایش داده شده است.

The screenshot shows a SQL query window with the following text:

```
SELECT * FROM Customers
WHERE Country='Iran' AND City='Tehran';
```

Below the query, the results are displayed in a table:

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran

شکل ۳-۲۹ بررسی دستور WHERE

توجه داشته باشید که تمام مثال‌های این فصل از جدول ۳-۱ برداشته شده است و دقیقاً طبق همان جدول عمل کنید.

مثال دوم - دستور OR

```
SELECT * FROM Customers
WHERE Country='Iran' OR City='Shiraz';
```

برای دستور OR نیز همان مثال قبلی را اجرا می‌کنیم با این تفاوت که شهر را به شیراز تغییر می‌دهیم، دستور OR به این نکته اشاره دارد که اگر یکی از این شرایط درست باشد، پس شرط درست است و خروجی آن سطر باید نمایش داده شود که در شکل ۳-۳۰ این موضوع را مشاهده می‌کنید؛ با اینکه شهر Shiraz در دستور وارد شده، چون یکی از شرط‌ها، یعنی Iran درست بوده، خروجی نمایش داده شده است.

The screenshot shows a SQL query window with the following text:

```
SELECT * FROM Customers
WHERE Country='Iran' OR City='Shiraz';
```

Below the query, the results are displayed in a table:

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran

شکل ۳-۳۰ بررسی دستور WHERE

مثال سوم دستور NOT

این دستور، یعنی منفی کردن شرط، یعنی اینکه اگر NOT قبل از آن قرار دهیم، یعنی اینکه آن کلید نباید در خروجی باشد.

```
SELECT * FROM Customers
WHERE NOT Country='Iran';
```

در دستور بالا، اگر کشور Iran در یکی از سطرها باشد آن سطر در خروجی نمایش داده نخواهد شد که این موضوع را در شکل ۳-۳۱ مشاهده می‌کنید.

```
SELECT * FROM Customers
WHERE NOT Country='Iran';
```

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany
2	2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico
3	3	Antonio	Juan	Mataderos 2312	México	05023	Mexico
4	4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden
6	6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway

شکل ۳-۳۱ بررسی دستور WHERE

مثال چهارم- ترکیب سه دستور AND, OR, NOT

برای اینکه درک کامل تری از این سه دستور داشته باشیم، می‌خواهیم یک مثال کلی برای این سه دستور بنویسیم.

```
SELECT * FROM Customers
WHERE Country='Germany' AND (City='Berlin' OR City='Frankfurt');
```

در دستور بالا، کشور مورد نظر Germany در نظر گرفته شده است، اما در شرطی که در داخل پراتز قرار داده شده، گفته که یکی از شهرها می‌تواند Berlin و یا Frankfurt باشد که چون شهر Frankfurt در جدول وجود داشت به این دلیل شرط، درست و به‌مانند شکل ۳-۳۲ در خروجی نمایش داده شده است.

```
SELECT * FROM Customers
WHERE Country='Germany' AND (City='Berlin' OR City='Frankfurt');
```

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany

شکل ۳-۳۲ بررسی WHERE

اگر بخواهیم مثالی از NOT بنویسیم می‌توانیم از دستور زیر استفاده کنیم:

```
SELECT * FROM Customers
WHERE NOT Country='Germany' AND NOT Country='USA';
```

در این دستور، اگر کشور مورد نظر آمریکا و آلمان نباشد باید لیست نهایی در خروجی چاپ شود که این موضوع را در شکل ۳-۳۳ مشاهده می‌کنید.

The screenshot shows a SQL Query Editor window with the following query:

```
SELECT * FROM Customers
WHERE NOT Country='Germany' AND NOT Country='USA';
```

The results table is as follows:

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico
2	3	Antonio	Juan	Mataderos 2312	México	05023	Mexico
3	4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
4	5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden
5	6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway
6	7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran

شکل ۳-۳۳ بررسی دستور WHERE

۳-۱-۵-۷ بررسی دستور LIKE

با این دستور می‌توانید یک Pattern ایجاد کنید، یعنی اینکه مثلاً مشخص کنید که ستون‌هایی که با حرف P شروع می‌شوند در خروجی چاپ شوند؛ در زیر شمای کلی این دستور را مشاهده می‌کنید:

```
SELECT column1, column2, ...
FROM table_name
WHERE column LIKE pattern;
```

در این دستور و در شرط مورد نظر به این نکته اشاره شده است که خروجی کار باید طبق Pattern مورد نظر باشد، در لیست زیر چند نمونه از عملگرهای دستور LIKE را مشاهده می‌کنید که با % و _ اجرا شده‌اند.

جدول ۳-۷ عملگر Like

توضیحات	عملگر LIKE
این دستور مقادیری را پیدا می‌کند که با حرف "a" شروع می‌شوند.	LIKE 'a%'
این دستور مقادیری را پیدا می‌کند که به حرف "a" ختم می‌شوند.	LIKE '%a'
مقادیری را پیدا می‌کند که or داشته باشند.	LIKE '%or%'
این دستور به مقادیری اشاره دارد که حرف دوم آنها r باشد.	LIKE '_r%'
به مقادیری اشاره دارد که با حرف a شروع شده و دارای حداقل دو حرف است.	LIKE 'a_%'
به مقادیری اشاره دارد که با حرف a شروع شده و حداقل دارای سه کاراکتر هستند.	LIKE 'a__%'
مقادیری را پیدا می‌کنند که با حرف a شروع شده و با حرف o پایان می‌یابد.	LIKE 'a%o'
مقادیری را که مقدار دوم آنها، ۲ و مقدار آخر، ۳ باشد.	LIKE '_2%3'

مثال اول - a%

در این قسمت می‌خواهیم گزینه‌هایی را پیدا کنیم که با حرف a شروع می‌شوند، در دستور زیر قسمت Customer Name بررسی می‌شود و اگر کلمه‌ای با a شروع شود را در خروجی به‌مانند شکل ۳-۳۴ چاپ می‌کند:

```
SELECT * FROM Customers
WHERE "Customer Name" LIKE 'a%';
```

The screenshot shows a SQL query window with the following text:

```
SELECT * FROM Customers
WHERE "Customer Name" LIKE 'a%';
```

Below the query window is a results grid with the following data:

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country
1	2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico
2	3	Antonio	Juan	Mataderos 2312	México	05023	Mexico

شکل ۳-۳۴ بررسی دستور LIKE

مثال دوم - %a

در این مثال در خروجی در ستون Customers Name، کلماتی که به a ختم می‌شوند در خروجی چاپ خواهند شد.

۳-۱-۵-۸ بررسی دستور BETWEEN

این دستور مقدار مشخص شده بین دو محدوده را در خروجی نمایش می‌دهد، شکل کلی این دستور به صورت زیر می‌باشد:

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;
```

برای اینکه دستور BETWEEN را اجرا کنیم باید یک ستون جدید به جدول اصلی خود اضافه کنیم؛ برای این کار طبق شکل ۳-۳۵ بر روی جدول مورد نظر خود کلیک راست کنید و قسمت Design را انتخاب کنید و یک ستون با عنوان Price و از نوع int وارد کنید.

The screenshot shows the design view of the Customers table in SQL Server Enterprise Manager. The table structure is as follows:

Column Name	Data Type	Allow Nulls
CustomerID	int	<input checked="" type="checkbox"/>
[Customer Name]	nchar(15)	<input checked="" type="checkbox"/>
[Contact Name]	nchar(15)	<input checked="" type="checkbox"/>
Address	varchar(50)	<input type="checkbox"/>
City	nchar(10)	<input checked="" type="checkbox"/>
postalcode	nchar(10)	<input checked="" type="checkbox"/>
country	nchar(10)	<input checked="" type="checkbox"/>
Price	int	<input checked="" type="checkbox"/>

شکل ۳-۳۵ بررسی دستور BETWEEN

در ادامه باید به مانند شکل ۳-۳۶ اطلاعات ستون Price را وارد کنید؛ برای تمرین می‌توانید از طریق دستور INSERT این کار را انجام دهید.

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	Ana	Juan	Avda. de la Con...	México	05021	Mexico	11
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden	52
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88
7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

شکل ۳-۳۶ نمایش جدول Price

مثال اول - دستور BETWEEN

```
SELECT * FROM Customers
WHERE Price BETWEEN 10 AND 20;
```

در دستور بالا، قیمت بین ۱۰ تا ۲۰ در خروجی نمایش داده خواهد شد که این موضوع را در شکل ۳-۳۷ مشاهده می‌کنید:

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11

شکل ۳-۳۷ بررسی دستور BETWEEN

مثال دوم

```
SELECT * FROM Customers
WHERE "Contact Name" BETWEEN 'A' and 'Z';
```

در دستور بالا و در ستون Contact Name، اسامی که با حروف A تا Z شروع شوند در خروجی چاپ خواهد شد.

مثال سوم - منظم کردن خروجی

```
SELECT * FROM Customers
WHERE "Contact Name" BETWEEN 'Alfred' and 'Juan'
ORDER BY "Contact Name";
```

در دستور بالا، یک دستور جدید اضافه شده است با نام ORDER BY که این دستور برای منظم کردن ستون خروجی است که این موضوع را در شکل ۳-۳۸ مشاهده می‌کنید.

The screenshot shows a SQL query window with the following text:

```
SELECT * FROM Customers
WHERE "Contact Name" BETWEEN 'Alfred' and 'Juan'
ORDER BY "Contact Name";
```

The results grid shows the following data:

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	Mania	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9
3	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden	52
4	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11
5	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25

شکل ۳-۲۸ بررسی دستور BETWEEN

۳-۱-۵-۹ بررسی دستور IN

این دستور اصولاً در یک شرط به کار گرفته می‌شود و مشخص می‌کند چه گزینه‌هایی باید در خروجی نمایش داده شود، شمای کلی دستور IN به صورت زیر است:

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2, ...);
```

روش دیگری نیز برای به کارگیری دستور IN وجود دارد که شکل کلی آن به صورت زیر است:

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (SELECT STATEMENT);
```

مثال اول - بررسی دستور IN

```
SELECT * FROM Customers
WHERE City NOT IN ('London', 'Tehran');
```

در مثال بالا، ستون‌هایی در خروجی چاپ خواهد شد که دارای شهرهای London و Tehran نباشند، چون پیش از دستور IN، دستور NOT به کار برده شده است و به این دلیل شرط منفی خواهد شد.

The screenshot shows a SQL query window with the following text:

```
SELECT * FROM Customers
WHERE City NOT IN ('London', 'Tehran');
```

The results grid shows the following data:

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	Mania	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25
4	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden	52
5	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88

شکل ۳-۲۹ بررسی دستور IN

۳-۱-۵-۱۰ بررسی دستور TOP

این دستور برای نمایش تعداد سطر در خروجی کاربرد دارد که دارای گزینه‌های مختلفی است و شکل کلی آن را در زیر مشاهده می‌کنید:

```
SELECT TOP number|percent column_name(s)
FROM table_name
WHERE condition;
```

مثال اول - بررسی دستور TOP

```
SELECT TOP 4 * FROM Customers;
```

در دستور بالا، ۴ سطر از جدول Customers به‌مانند شکل ۳-۴۰ در خروجی چاپ خواهد شد.

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	Mania	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36

شکل ۳-۴۰ بررسی دستور TOP

مثال دوم - استفاده از OFFSET

```
SELECT * FROM Customers
ORDER BY CustomerID
OFFSET 2 ROWS;
```

برای استفاده از دستور OFFSET حتماً باید قبل از آن از دستور ORDER BY استفاده کنید تا ID مورد نظر منظم شود و بعد می‌توانید از دستور OFFSET استفاده کنید؛ توجه داشته باشید که این دستور از شماره‌ای که در مقابل آن نوشته می‌شود در جدول مورد نظر شماره‌های بعد از آن را در خروجی چاپ می‌کند، مانند دستور بالا که شماره‌ی ۲ نوشته شده است، اما از شماره‌ی ۳ به بعد در خروجی چاپ خواهد شد، اگر به شکل ۳-۴۱ توجه کنید این موضوع را مشاهده می‌کنید؛ توجه داشته باشد که دستور ROWS به ردیف‌ها اشاره دارد.

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden	52
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88
7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9

شکل ۳-۴۱ بررسی OFFSET

مثال سوم - استفاده از دستورات FETCH و PRESENT

```
SELECT * FROM Customers
ORDER BY CustomerID
OFFSET 2 ROWS
FETCH FIRST 3 ROWS ONLY;
```

این دستور از چند قسمت تشکیل شده است؛ در خط اول با دستور SELECT، جدول Customers انتخاب شده و در خط دوم، CustomerID با دستور ORDER BY مرتب می‌شود، بعد از مرتب شدن CustomerID با دستور OFFSET

مشخص می‌کنیم که از سطر سوم به بعد در خروجی چاپ شود و در خط آخر نیز مشخص می‌کنیم که چند سطر در خروجی چاپ شود که این موضوع را در شکل ۳-۴۲ مشاهده می‌کنید.

```
SELECT * FROM Customers
ORDER BY CustomerID
OFFSET 2 ROWS
FETCH FIRST 3 ROWS ONLY;
```

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25
2	4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36
3	5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden	52

شکل ۳-۴۲ بررسی دستور *PRESENT* و *FETCH*

مثال چهارم – استفاده از *WHERE* به همراه *TOP*

```
SELECT TOP 3 * FROM Customers
WHERE Country='Germany';
```

در دستور بالا با استفاده از *TOP* تعداد خروجی کار مشخص شده است، یعنی اگر سطر مورد نظر کشورش Germany باشد در خروجی چاپ خواهد شد، اما تنها ۳ سطر آن چاپ خواهد شد و اگر بیشتر باشد، چاپ نخواهد شد که در شکل ۳-۴۳ این موضوع را مشاهده می‌کنید.

```
SELECT TOP 3 * FROM Customers
WHERE Country='Germany';
```

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	8	Azadeh	Tishebarsar	Monikh	Frankfurt	5565	Germany	20

شکل ۳-۴۳ بررسی دستور *TOP*

۱۱-۵-۱-۳ بررسی دستور *MIN* and *MAX*

این دستور کمترین و بیشترین مقدار در یک ستون را برمی‌گرداند که در زیر شکل کلی آن را مشاهده می‌کنید:

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

MIN

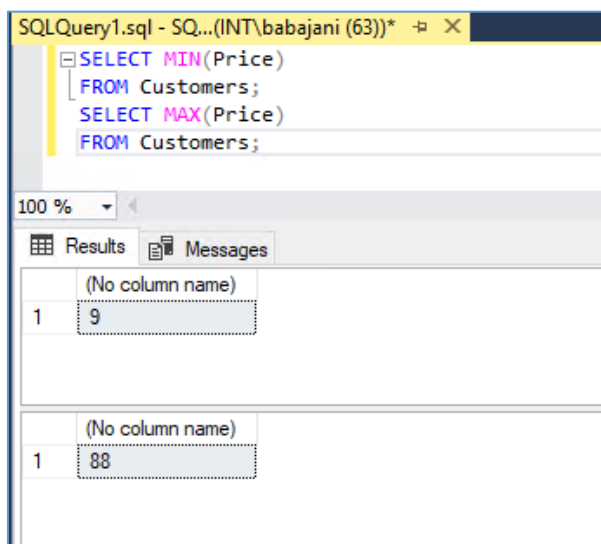
```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

MAX

مثال یک – دستور *MIN* & *MAX*


```
SELECT MIN(Price)
FROM Customers;
SELECT MAX(Price)
FROM Customers;
```

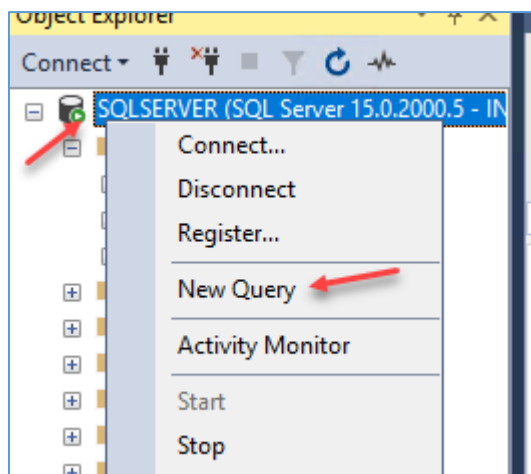
در دستور بالا، دو عدد MIN و MAX از ستون Price در خروجی چاپ خواهد شد که این موضوع را در شکل ۴۴-۳ مشاهده می‌کنید.



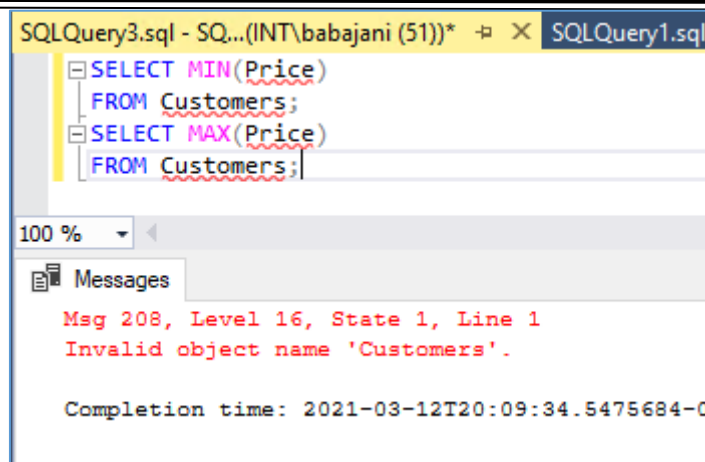
شکل ۴۴-۳ بررسی دستور MIN و MAX

نکته‌ی مهم:

توجه کنید که ایجاد Query باید در مسیر درست انجام شود، اگر به‌مانند شکل ۴۵-۳ بر روی SQLSERVER کلیک راست کنید و گزینه‌ی New Query را انتخاب کنید و بعد دستورات بالا را در آن اجرا کنید با خطای شکل ۴۶-۳ مواجه خواهید شد که دلیل آن نیز این است که جدول Customers در زیرمجموعه‌ی دیگری قرار دارد و حتماً باید بر روی دیتابیس مورد نظر این دستور را اجرا کنید.



شکل ۴۵-۳ ایجاد Query

شکل ۴۶-۳ بررسی دستور *MIN* و *MAX*

۳-۱-۵-۱۲ بررسی دستور *COUNT*, *AVG* و *SUM*

شکل کلی دستور *Count* به صورت زیر است:

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

شکل کلی دستور *AVG* به صورت زیر است:

```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

شکل کلی دستور *SUM* به صورت زیر است:

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

برای هر یک از دستورات بالا، یک مثال را با هم بررسی می‌کنیم:

مثال اول - بررسی دستور *Count*

```
SELECT * From Customers;
```

```
SELECT COUNT(PRICE)
FROM Customers;
```

در خط اول، کل جدول *Customers* در خروجی چاپ خواهد شد و در خط بعدی با دستور *COUNT*، تعداد موجودی در ستون *Price* مشخص خواهد شد که در شکل ۴۷-۳ این موضوع را مشاهده می‌کنید؛ در قسمت دوم عدد ۸ نوشته شده است که تعداد موجودی در ستون *Price* است.

```

SELECT * FROM Customers;
SELECT COUNT(PRICE)
FROM Customers;

```

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden	52
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88
7	Farshid	Babajani	Seled Khandan	Tehran	8843	Iran	9
8	Azadeh	Tishebarsar	Monikh	Frankfurt	5565	Germany	20

(No column name)

1	8
---	---

شکل ۳-۴۷ بررسی دستور COUNT

مثال دوم - بررسی دستور AVG

```

SELECT AVG(PRICE)
FROM Customers;

```

در دستور بالا، میانگین اعدادی ستون Price حساب شده است و در خروجی چاپ خواهد شد که این موضوع را در شکل ۳-۴۸ مشاهده می‌کنید، اگر توجه کنید با ماشین حساب جمع کل ستون Price را محاسبه و تقسیم بر ۸ کردیم که نتیجه‌ی آن عدد ۳۱ شده است.

```

SELECT AVG(PRICE)
FROM Customers;

```

(No column name)	
1	31

Calculator

Standard

251 ÷ 8 =

31.375

MC MR M+ M- MS M*

% CE C

شکل ۳-۴۸ بررسی دستور AVG

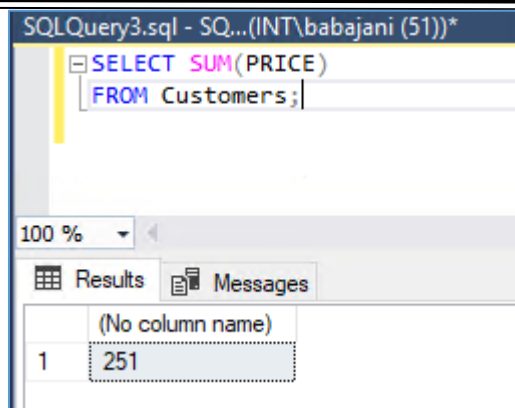
مثال سوم - بررسی دستور SUM

```

SELECT SUM(PRICE)
FROM Customers;

```

این دستور، کل ستون Price را جمع خواهد کرد که در شکل ۳-۴۹ این موضوع را مشاهده می‌کنید.



شکل ۴۹-۳ بررسی دستور SUM

۳-۱-۵-۱۳ بررسی Wildcards

این دستور در کنار دستور LIKE استفاده می‌شود و برای جستجو و جایگزینی در یک رشته کاربرد دارند، در زیر کاراکترهایی که می‌توان به‌عنوان Wildcard در دستورات استفاده کرد.

جدول ۳-۸ بررسی Wildcard

Symbol	توضیحات	مثال
%	صفر یا اکثر نویسه‌ها را نشان می‌دهد.	bl% کلماتی مانند bl, black, blue, blob را پیدا می‌کند.
_	نمایندگی یک شخصیت واحد است	دستور h_t کلماتی مانند hot, hat, hit را پیدا می‌کند.
[]	هر کاراکتر را در داخل براکت نشان می‌دهد.	دستور h[oa]t کلمات hat و hot را نشان می‌دهد، اما کلمه‌ی hit را نمایش نمی‌دهد.
^	هر کاراکتری که در پرانتز نیست را نشان می‌دهد.	دستور h[^oa]t هر کلمه‌ای که در گروه نباشد را نشان می‌دهد، مانند hit.
-	طیف وسیعی از نویسه‌ها را نشان می‌دهد.	دستور c[a-b]t کلمات cat و cbt را پیدا می‌کند.

مثال اول

```
SELECT * FROM Customers
WHERE City LIKE 'fr%';
```

در دستور بالا، شهرهایی که با نام fr شروع می‌شوند در خروجی چاپ خواهند شد که در شکل ۳-۵۰ مشاهده می‌کنید.

```

SELECT * FROM Customers
WHERE City LIKE 'fr%';

```

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	8	Azadeh	Tishebarsar	Monikh	Frankfurt	5565	Germany	20

شکل ۳-۵۰ بررسی Wildcards

مثال دوم

```

SELECT * FROM Customers
WHERE City LIKE '%hr%';

```

دستور بالا، شهرهایی که بین کلماتشان hr دارند در خروجی چاپ خواهد شد که در شکل ۳-۵۱ این موضوع را مشاهده می‌کنید:

```

SELECT * FROM Customers
WHERE City LIKE '%hr%';

```

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9

شکل ۳-۵۱ بررسی دستور Wildcards

مثال سوم

```

SELECT * FROM Customers
WHERE City LIKE '[bspt]%' ;

```

در دستور بالا، شهرهایی که با یکی از حروف bspt شروع شوند در خروجی چاپ خواهد شد که شکل ۳-۵۲ نمایانگر این موضوع است.

```

SELECT * FROM Customers
WHERE City LIKE '[bspt]%' ;

```

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88
2	7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9

شکل ۳-۵۲ بررسی دستور Wildcards

مثال چهارم

```

SELECT * FROM Customers
WHERE City LIKE '[a-f]%' ;

```

در دستور بالا، شهرهایی که با حروف a تا f شروع می‌شوند در خروجی چاپ خواهد شد که در شکل ۳-۵۳ مشخص شده است.

The screenshot shows a SQL query window with the following text:

```
SELECT * FROM Customers
WHERE City LIKE '[a-f]%'
```

The results grid below shows the following data:

	CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	8	Azadeh	Tishebarsar	Monikh	Frankfurt	5565	Germany	20

شکل ۳-۵۳ بررسی دستور Wildcards

۳-۱-۵-۱۴ بررسی دستور Alias

این دستور برای ایجاد یک نام مستعار برای ستون‌های یک جدول کاربرد دارد تا خوانایی آن راحت‌تر شود؛ این نام تنها در زمان جستجو ایجاد می‌شود و بعد از آن حذف خواهد شد، این دستور را باید با AS اجرا کرد که شکل کلی آن به صورت زیر است:

```
SELECT column_name AS alias_name
FROM table_name;
```

در دستور بالا، alias_name یک نام مستعار برای جدول مورد نظر است خواهد بود.

مثال اول

```
SELECT CustomerID AS ID, "Customer Name" AS Customer
FROM Customers;
```

در دستور بالا، دو ستون CustomerID و Customer Name با دو نام جدید در خروجی چاپ خواهند شد که در شکل ۳-۵۴ این موضوع را مشاهده می‌کنید.

The screenshot shows a SQL query window with the following text:

```
SELECT CustomerID AS ID, "Customer Name" AS Customer
FROM Customers;
```

The results grid below shows the following data:

	ID	Customer
1	1	Maria
2	2	Ana
3	3	Antonio
4	4	Thomas
5	5	Christina
6	6	Cardinal
7	7	Farshid
8	8	Azadeh

شکل ۳-۵۴ بررسی دستور Alias

مثال دوم

```
SELECT "Customer Name" AS Customer, "Contact Name" AS "Contact Person"
FROM Customers;
```

این دستور نیز به مانند دستور قبلی است و نکته‌ای که در این دستور وجود دارد، این است که برای ایجاد یک ستون دو حرفی باید کلمه‌ی مورد نظر را در یک گروه قرار دهید؛ این موضوع را در شکل ۳-۵۵ مشاهده می‌کنید.

```
SELECT "Customer Name" AS Customer, "Contact Name" AS "Contact Person"
FROM Customers;
```

	Customer	Contact Person
1	Maria	Alfred Schmidt
2	Ana	Juan
3	Antonio	Juan
4	Thomas	Thomas Hardy
5	Christina	Christina
6	Cardinal	Tom B. Erichsen
7	Farshid	Babajani
8	Azadeh	Tishebarsar

شکل ۳-۵۵ بررسی دستور Alias

مثال سوم

```
SELECT "Customer Name", Address + ', ' + PostalCode + ' ' + City + ', ' + Country AS
Address
FROM Customers;
```

در دستور بالا، دو ستون در خروجی چاپ خواهد شد که در ستون Address، چند گزینه چاپ خواهد شد که شامل Address, PostalCode, City و کشور خواهد بود که در شکل ۳-۵۶ این موضوع را مشاهده می‌کنید.

```
SELECT "Customer Name", Address + ', ' + PostalCode + ' ' + City + ', ' + Country AS Address
FROM Customers;
```

Customer Name	Address
Maria	Obere Str. 57, 12209 Frankfurt , Germany
Ana	Avda. de la Constitución 2222, 05021 Méxi...
Antonio	Mataderos 2312, 05023 México , Mexico
Thomas	120 Hanover Sq., WA1 1DP London , UK
Christina	Berguvsvägen 8, S-958 22 Luleå , Sweden
Cardinal	Skagen 21, 4006 Stavanger , Norway
Farshid	Seied Khandan, 8843 Tehran , Iran
Azadeh	Monikh, 5565 Frankfurt , Germany

شکل ۳-۵۶ بررسی دستور Alias

۳-۱-۵-۱۵ بررسی دستور GROUP BY

دستور GROUP BY برای جمع‌بندی مقادیر یکسان کاربرد دارد و آنها را توسط تابع‌های COUNT(), MAX(), MIN(), SUM(), AVG() در خروجی چاپ خواهد کرد، شکل کلی این دستور به صورت زیر خواهد بود:

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
ORDER BY column_name(s);
```

مثال اول

```
SELECT * FROM Customers
```

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country;
```

در دستور بالا، در خط اول یک خروجی کلی از اطلاعات جدول Customers را خواهیم داشت و در ادامه با استفاده از دستور COUNT، تعداد مشتریان هر کشور شمارش می‌شود و در آخر با دستور GROUP BY، تعداد مشتریان به همراه کشورهای آنها چاپ شده است؛ در شکل ۳-۵۷ این موضوع را مشاهده می‌کنید.

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden	52
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88
7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9
8	Azadeh	Tishebarsar	Monikh	Frankfurt	5565	Germany	20

(No column name)	Country	
1	2	Germany
2	1	Iran
3	2	Mexico
4	1	Norway
5	1	Sweden
6	1	UK

شکل ۳-۵۷ بررسی دستور GROUP BY

مثال دوم

```
SELECT * FROM Customers
```

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country;
```

در دستور بالا به مانند دستورات قبلی مشتریان مورد نظر چاپ خواهد شد، اما در آخر خط با دستور ORDER BY همراه دستور DESC از بیشتر به کمتر مشتریان مرتب خواهند شد که در شکل ۳-۵۸ این موضوع را مشاهده می‌کنید.

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
ORDER BY COUNT(CustomerID) DESC;
```

(No column name)	Country	
1	2	Mexico
2	2	Germany
3	1	Iran
4	1	Norway
5	1	Sweden
6	1	UK

شکل ۳-۵۸ بررسی دستور GROUP BY

HAVING بررسی دستور ۳-۱-۵-۱۶

این دستور، یک دستور کمکی است که می‌تواند در دستورات اضافه شود؛ شکل کل این دستور به صورت زیر است:

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition
ORDER BY column_name(s);
```

مثال اول

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
HAVING COUNT(CustomerID) > 0
ORDER BY COUNT(CustomerID) DESC;
```

در مثال بالا، عدد صفری که مقابل دستور HAVING قرار داده شده به این معنی است که تعداد مشتریان هر کشور اگر بیشتر از صفر بود، در خروجی چاپ کند. شما می‌توانید این عدد را تغییر دهید و یا اینکه عملگر آن را عوض کنید، در شکل ۳-۵۹ این مثال را مشاهده می‌کنید.

The screenshot shows a SQL query window with the following text:

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
HAVING COUNT(CustomerID) > 0
ORDER BY COUNT(CustomerID) DESC;
```

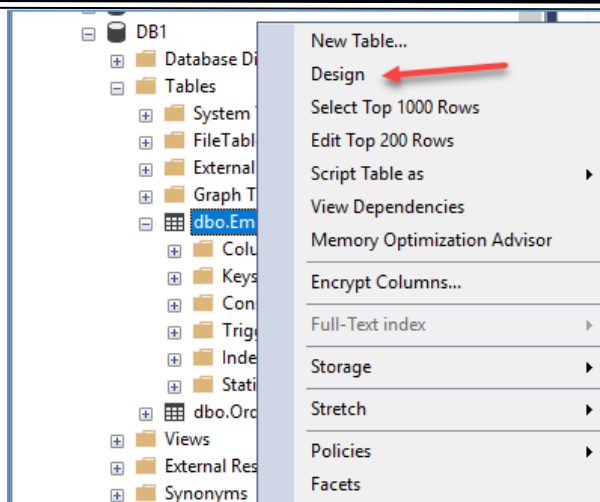
Below the query, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with the following data:

	(No column name)	Country
1	2	Mexico
2	2	Germany
3	1	Iran
4	1	Norway
5	1	Sweden
6	1	UK

شکل ۳-۵۹ بررسی دستور HAVING

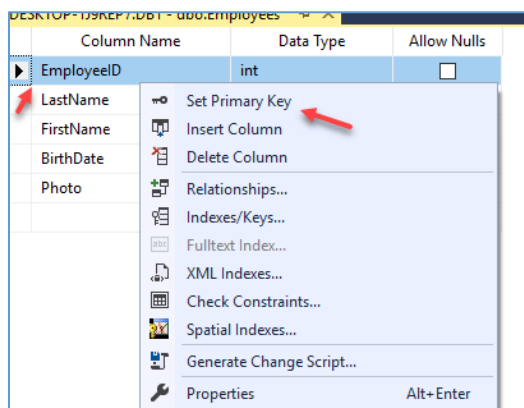
PRIMARY KEY بررسی ۳-۱-۵-۱۷

زمانی که یک جدول ایجاد می‌کنید، می‌توانید مشخص کنید که کدام ستون به عنوان PRIMARY KEY انتخاب شود؛ انتخاب یک ستون به عنوان PRIMARY KEY باعث می‌شود که مقدار آن دیگر، Null یا خالی نباشد و حتماً باید آن را پر کنید؛ برای این کار به مانند شکل ۳-۶۰ بر روی جدول مورد نظر کلیک راست کنید و گزینه Design را انتخاب کنید.



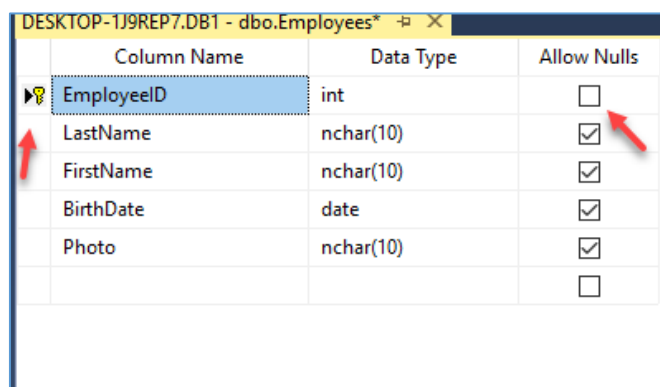
شکل ۳-۶۰ طراحی

به‌مانند شکل ۳-۶۱ بر روی ستونی که می‌خواهید به‌عنوان Primary Key انتخاب شود، کلیک راست کنید و گزینه‌ی Set Primary Key را انتخاب کنید.



شکل ۳-۶۱ بررسی PRIMARY KEY

در بعد از انتخاب Primary Key در شکل ۳-۶۲ یک کلید در کنار ستون مورد نظر قرار خواهد گرفت و در قسمت Allow Nulls، تیک آن برداشته خواهد شد؛ به دلیل اینکه این ستون اصلاً نباید خالی باشد، مانند شماره‌ی دانشجویی در دانشگاه‌ها.

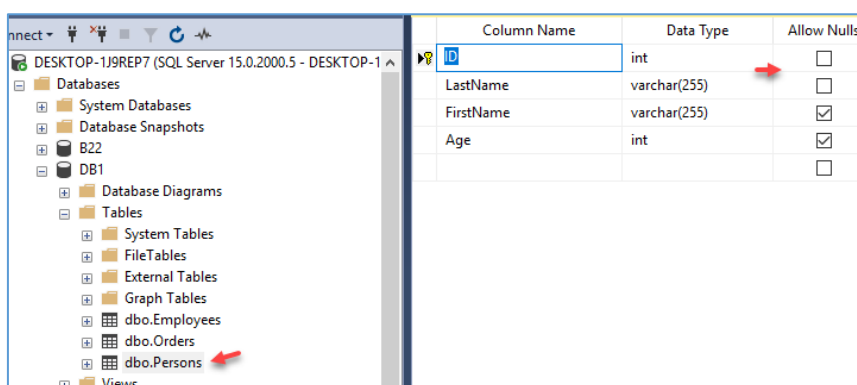


شکل ۳-۶۲ بررسی PRIMARY KEY

برای اینکه بتوانیم از طریق دستورات T-SQL یک جدول ایجاد کنیم که دارای Primary Key باشد باید دستورات زیر را اجرا کنیم:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    PRIMARY KEY (ID)
);
```

در دستورات بالا، یک جدول جدید با نام Persons ایجاد کردیم که ستون‌های ID و LastName نباید خالی باشد و در قسمت آخر PRIMARY KEY را نیز برابر ID قرار دادیم، اگر به شکل ۳-۶۳ توجه کنید نتیجه‌ی دستور را مشاهده خواهید کرد.



شکل ۳-۶۳ بررسی PRIMARY KEY

اگر بخواهیم دستور بالا را تغییر دهیم، می‌توانیم از دستور زیر استفاده کنیم:

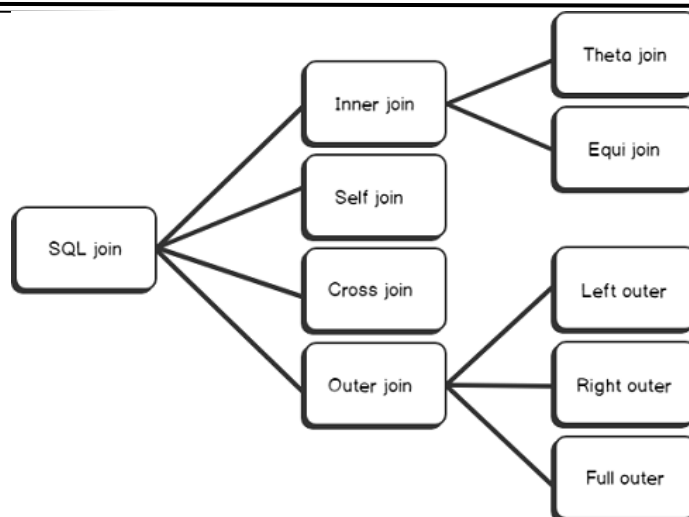
```
CREATE TABLE Persons (
    ID int NOT NULL PRIMARY KEY,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int
);
```

در دستور بالا، Primary Key در کنار ID قرار گرفته شده است.

۳-۱-۵-۱۸ بررسی JOIN در SQL

یکی از مهم‌ترین دستوراتی که در SQL می‌توان نام برد، دستور JOIN است که این دستور اساساً برای ترکیب و بازیابی داده‌ها از دو یا چند جدول مورد استفاده قرار می‌گیرد. در یک پایگاه داده، رابطه‌ای در دنیای واقعی، داده‌ها در تعداد زیادی جداول ساختار می‌یابند و به این دلیل، دائماً باید به این جداول متصل شد. چهار نوع اساسی عضویت در SQL Server وجود دارد: Self، Outer، Inner و Cross join. برای اینکه یک مرور سریع در مورد همه‌ی این پیوستن‌ها داشته باشید در زیر آنها را بررسی می‌کنیم:

در شکل ۳-۶۴ یک شمای کلی از نحوه‌ی ارتباط دستورات JOIN را مشاهده می‌کنید که هر کدام توضیح مخصوص به خود را دارند.



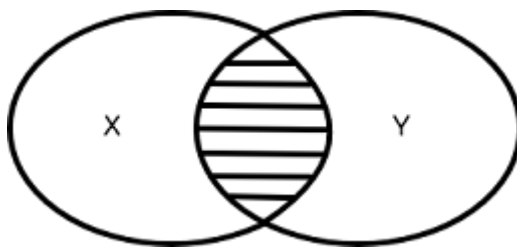
شکل ۳-۶۴ بررسی دستور JOIN

توانایی ترکیب نتایج حاصل از سطرهای مرتبط از چندین جدول، قسمت مهمی در طراحی سیستم پایگاه داده‌ای رابطه‌ای است. در SQL Server این امر با دستور join انجام می‌شود. این ماهیت سیستم‌های پایگاه داده‌ای رابطه‌ای سنتی است که برخی از جدول‌ها حاوی اطلاعات مربوط به جداول دیگر با مقدار کلیدی مشترک هستند. با استفاده از پیوستن SQL، می‌توانید به راحتی از طریق چندین جدول با این کلیدهای مشترک، پرس‌وجوهایی را در مجموعه داده‌های مرتبط انجام دهید.

در ادامه نوع‌های مختلف SQL JOIN را که در شکل ۳-۴۳-۳ مشخص شده است را با هم بررسی می‌کنیم:

۱- SQL inner join

این دستور یکی از ساده‌ترین دستورات JOIN است که نتیجه‌ی خروجی آن، سطری از هر دو جدول است که در آن شرایط پیوستگی یکی باشد؛ در شکل ۳-۶۵ شمای کلی این دستور را مشاهده می‌کنید.



شکل ۳-۶۵ SQL inner join

دستورات کلی Inner Join به صورت زیر است:

```

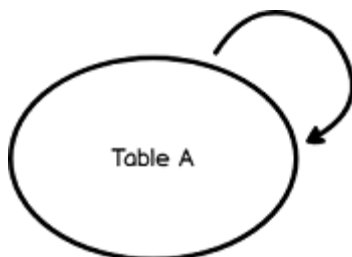
SELECT ColumnList from LeftTable L
INNER join RightTable R
ON L.Column=R.Column
  
```

۲- SQL self join

این دستور باعث می‌شود، یک جدول با خودش Join شود که در ادامه، نحوه‌ی کار آن را بررسی خواهیم کرد.

شکل ۳-۶۶ نمای کلی دستور به صورت زیر است:

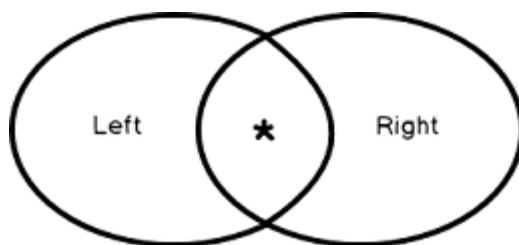
```
SELECT column_name(s)
FROM table1 T1, table1 T2
WHERE condition
```



شکل ۳-۶۶ SQL self join

-۳ SQL cross join

این نوع JOIN حاصل ضرب دکارتی سطرهای تعیین شده در دستور join را در خروجی نمایش می دهد. این دستور، سطرهایی را تولید می کند که ترکیبی از سطر جدول اول و دوم است که نمای کلی آن را در شکل ۳-۶۷ مشاهده می کنید.



شکل ۳-۶۷ SQL cross Join

-۴ SQL outer join

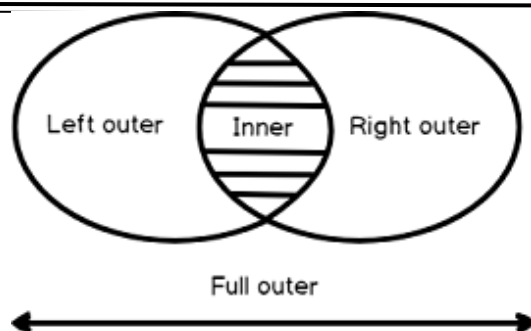
این دستور شامل سه دستور LEFT OUTER, RIGHT OUTER, FULL است که هر کدام برای کار خاص طراحی شده اند.

دستور LEFT OUTER JOIN

شکل کلی این دستور به صورت زیر است:

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```

این دستور، تمام اطلاعات را از جدول سمت چپ، یعنی جدول اول و اطلاعات منطبق را از جدول دوم در خروجی چاپ می کند و اگر هم عدم تطابق ایجاد شود، نتیجه ی صفر را از جدول سمت راست برمی گرداند؛ در شکل ۳-۶۸ این موضوع مشخص شده است.



شکل ۶۸-۳ SQL Full outer

مثال دستور INNER JOIN

برای اینکه بیشتر با این دستورات آشنا شویم یک مثال را با هم بررسی می‌کنیم؛ برای شروع نیاز به دو جدول داریم تا بتوانیم عملیات JOIN را بر روی آنها انجام دهیم.

جدول ۳-۹ Customers

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price
1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10
2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36
5	Christina	Christina	Berguvsvägen 8	Luleå	S-958 22	Sweden	52
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88
7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9
8	Azadeh	Tishebarsar	Monikh	Frankfurt	5565	Germany	20

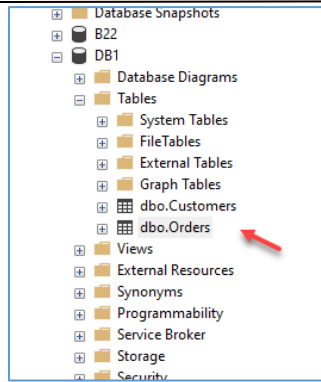
جدول Order

جدول ۳-۱۰ Order

OrderID	CustomerID	OrderDate
10308	8	1996-09-18
10309	6	1996-09-19
10310	4	1996-09-20

دو جدول ۳-۹ و ۳-۱۰ را مشاهده می‌کنید که باید در SQL آنها را ایجاد کنیم تا بتوانیم عملیات Join را بر روی آنها انجام دهیم.

همان طور که در شکل ۳-۶۹ مشاهده می‌کنید، دو جدول مورد نظر در SQL ایجاد شده‌اند.



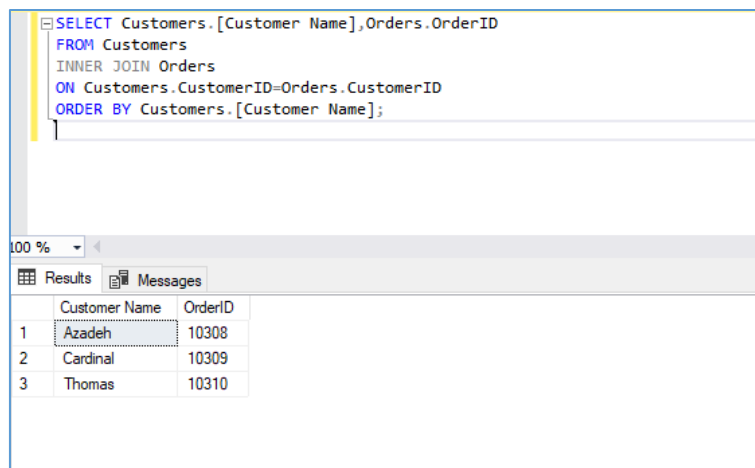
شکل ۳-۶۹ / ایجاد جدول

با استفاده از دستور زیر می‌توانیم بین دو جدول ارتباط داخلی برقرار کنیم:

```
SELECT Customers.[Customer Name],Orders.OrderID
FROM Customers
INNER JOIN Orders
ON Customers.CustomerID=Orders.CustomerID
ORDER BY Customers.[Customer Name];
```

در خط اول با دستور SELECT، ستون‌های خروجی را که قرار است چاپ شوند را مشخص می‌کنیم که ستون Customer Name از جدول Customers و ستون OrderID از جدول Orders چاپ خواهد شد. در خط دوم جدول Customers انتخاب می‌شود و در خط سوم مشخص می‌کنیم که باید با جدول Orders ارتباط INNER JOIN برقرار کند؛ در خط چهارم باید شرط برقراری ارتباط را مشخص کنیم که با دستور ON این کار را انجام می‌دهیم و بعد از آن، CustomerID از جدول Customers را با ستون CustomerID از جدول Orders را مساوی قرار می‌دهیم؛ این بدان معنا است که اگر در این دو ستون، عبارت مساوی هم قرار داشته باشد، آن سطر در خروجی چاپ خواهد شد؛ در خط آخر نیز با دستور ORDER BY، ستون Customer Name را منظم می‌کنیم تا در خروجی، نمایش بهتری داشته باشد.

اگر به شکل ۳-۷۰ توجه کنید، مشاهده خواهید کرد سه سطر در خروجی آپ شده است و آن هم به دلیل برابر بودن اعداد در ستون‌های Customer ID در هر دو جدول است.



شکل ۳-۷۰ / بررسی دستور JOIN

نکته: دستور JOIN و INNER JOIN یک کار را انجام می‌دهند، اما دستور INNER JOIN دارای عملکرد بهتری نسبت به دستور JOIN است و کار را در سریع‌ترین زمان ممکن انجام می‌دهد.

مثال دستور SELF JOIN

همان طور که گفتیم این دستور با خودش ارتباط برقرار می‌کند؛ برای تست این موضوع از جدول Customers استفاده می‌کنیم و دستور زیر را اجرا می‌کنیم:

```
SELECT A.[Customer Name] AS CustomerName1, B.[Customer Name] AS CustomerName2, A.City
FROM Customers A, Customers B
WHERE A.CustomerID <> B.CustomerID
AND A.City = B.City
ORDER BY A.City;
```

در دستور بالا و در خط اول، سه ستون در خروجی چاپ خواهد شد که دستور AS، یک ستون جدید در خروجی نمایش می‌دهد؛ در قسمت FROM، یک جدول به دو قسمت تبدیل شده است که یکی Customers A و دیگری Customers B است؛ در قسمت WHERE شرطی قرار دادیم و گفتیم که در صورتی که A.CustomerID برابر نباشد با B.CustomerID و همچنین شهر جدول A با شهر جدول B یکی باشد شرط درست است و باید در خروجی چاپ شود که در خط آخر نیز طبق ستون A.City خروجی منظم خواهد شد.

در شکل ۳-۷۱ خروجی دستورات بالا را مشاهده می‌کنید که سه ستون ایجاد و خروجی طبق شرطی که قرار دادیم، نمایش داده شده است.

```
SELECT A.[Customer Name] AS CustomerName1, B.[Customer Name] AS CustomerName2, A.City
FROM Customers A, Customers B
WHERE A.CustomerID <> B.CustomerID
AND A.City = B.City
ORDER BY A.City;
```

	CustomerName1	CustomerName2	City
1	Maria	Azadeh	Frankfurt
2	Azadeh	Maria	Frankfurt
3	Ana	Antonio	México
4	Antonio	Ana	México

شکل ۳-۷۱ بررسی دستور SELF JOIN

مثال دستور CROSS JOIN

همان طور که گفتیم، این دستور با استفاده از ضرب دکارتی خروجی را مشخص می‌کند.

```
SELECT *
FROM Customers
CROSS JOIN Orders;
```


در دستور بالا با دستور * SELECT باید همه‌ی ستون‌های جدول Customers در خروجی نمایش داده شوند، اما در خط آخر با دستور CROSS JOIN، تمام سطرهای جدول اول را ضرب در جدول Orders می‌کنیم که خروجی مانند شکل ۳-۷۲ خواهد شد.

همان طور که در شکل ۳-۷۲ مشاهده می‌کنید، هر کدام از سطرهای جدول Customers ضرب در هر کدام از سطرهای جدول Orders خواهد شد که نتیجه‌ی آن یک جدول بزرگ از اطلاعات است.

CustomerID	Customer Name	Contact Name	Address	City	postalcode	country	Price	OrderID	CustomerID	OrderDate
1	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10	10308	8	1996-09-18
2	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11	10308	8	1996-09-18
3	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25	10308	8	1996-09-18
4	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36	10308	8	1996-09-18
5	Christina	Christina	Berguvägen 8	Luleå	S-958 22	Sweden	52	10308	8	1996-09-18
6	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88	10308	8	1996-09-18
7	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9	10308	8	1996-09-18
8	Azadeh	Tsiehbarsar	Monikh	Frankfurt	5565	Germany	20	10308	8	1996-09-18
9	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10	10309	6	1996-09-19
10	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11	10309	6	1996-09-19
11	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25	10309	6	1996-09-19
12	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36	10309	6	1996-09-19
13	Christina	Christina	Berguvägen 8	Luleå	S-958 22	Sweden	52	10309	6	1996-09-19
14	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88	10309	6	1996-09-19
15	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9	10309	6	1996-09-19
16	Azadeh	Tsiehbarsar	Monikh	Frankfurt	5565	Germany	20	10309	6	1996-09-19
17	Maria	Alfred Schmidt	Obere Str. 57	Frankfurt	12209	Germany	10	10310	4	1996-09-20
18	Ana	Juan	Avda. de la Constitución 2222	México	05021	Mexico	11	10310	4	1996-09-20
19	Antonio	Juan	Mataderos 2312	México	05023	Mexico	25	10310	4	1996-09-20
20	Thomas	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK	36	10310	4	1996-09-20
21	Christina	Christina	Berguvägen 8	Luleå	S-958 22	Sweden	52	10310	4	1996-09-20
22	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway	88	10310	4	1996-09-20
23	Farshid	Babajani	Seied Khandan	Tehran	8843	Iran	9	10310	4	1996-09-20
24	Azadeh	Tsiehbarsar	Monikh	Frankfurt	5565	Germany	20	10310	4	1996-09-20

شکل ۳-۷۲ خروجی دستور CROSS JOIN

مثال دستور OUTER JOIN

این دستور همان طور که گفتیم از چندین قسمت تشکیل شده: OUTER:LEFT OUTER, RIGHT OUTER, FULL آنها را با هم بررسی می‌کنیم.

مثال دستور LEFT JOIN

دستور زیر را اجرا کنید:

```
SELECT Customers.[Customer Name], Orders.OrderID
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.[Customer Name];
```

در دستور بالا، در خط اول دستور SELECT ستون Customer Name را از جدول Customers و ستون OrderID را از جدول Orders در خروجی چاپ خواهد کرد؛ در خط دوم جدول Customers انتخاب و در خط سوم با استفاده از دستور LEFT JOIN می‌گوییم که ستون Customers.CustomerID برابر با Orders.CustomerID باشد؛ این دستور تمام اطلاعات جدول Customers را در صورت تطبیق در خروجی چاپ خواهد کرد و اگر در جدول دوم، یعنی Orders مقداری وجود نداشته باشد، کلمه‌ی null را برمی‌گرداند.

همان طور که در شکل ۳-۷۴ مشاهده می‌کنید، خروجی به درستی نمایش داده شده است.

```

SELECT Customers.[Customer Name], Orders.OrderID
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.[Customer Name];

```

	Customer Name	OrderID
1	Ana	NULL
2	Antonio	NULL
3	Azadeh	10308
4	Cardinal	10309
5	Christina	NULL
6	Farshid	NULL
7	Maria	NULL
8	Thomas	10310

شکل ۳-۷۴ بررسی دستور OUTER JOIN

مثال دستور RIGHT JOIN

برای انجام این دستور، دو جدول ۳-۱۱ و ۳-۱۲ را در SQL ایجاد کنید:

جدول ۳-۱۱ Orders

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	1996-09-18	3
10309	37	3	1996-09-19	1
10310	77	8	1996-09-20	2

جدول ۳-۱۲ Employees

EmployeeID	LastName	FirstName	BirthDate	Photo
1	Davolio	Nancy	12/8/1968	EmpID1.pic
2	Fuller	Andrew	2/19/1952	EmpID2.pic
3	Leverling	Janet	8/30/1963	EmpID3.pic

بعد از ایجاد جدول باید دستورات زیر را اجرا کنید:

```

SELECT Orders.OrderID, Employees.LastName, Employees.FirstName
FROM Orders
RIGHT JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID
ORDER BY Orders.OrderID;

```

در خط اول، سه ستون از دو جدول مشخص شده است و در ادامه به جدول Orders و جدول Employees متصل می‌شود که شرط RIGHT JOIN در جدول Employees پیاده‌سازی شده است، با این شرایط که Orders.EmployeeID = Employees.EmployeeID باشد؛ در شکل ۳-۷۵ این موضوع را مشاهده می‌کنید.

```

SELECT Orders.OrderID, Employees.LastName, Employees.FirstName
FROM Orders
RIGHT JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID
ORDER BY Orders.OrderID;

```

	OrderID	LastName	FirstName
1	NULL	Davolio	Nancy
2	NULL	Fuller	Andrew
3	10309	Leverling	Janet

شکل ۳-۷۵ بررسی دستور RIGHT JOIN

مثال دستور Full JOIN

این دستور به طور کامل، خروجی هر دو جدول را طبق شرطی که در دستور زیر مشاهده می‌کنید، نمایش می‌دهد.

```

SELECT Employees.FirstName, Orders.OrderID
FROM Employees
FULL OUTER JOIN Orders ON Employees.EmployeeID=Orders.CustomerID
ORDER BY Orders.EmployeeID;

```

```

SELECT Employees.FirstName, Orders.OrderID
FROM Employees
FULL OUTER JOIN Orders ON Employees.EmployeeID=Orders.CustomerID
ORDER BY Orders.EmployeeID;

```

	FirstName	OrderID
1	Nancy	NULL
2	Janet	NULL
3	NULL	10309
4	Andrew	10308
5	NULL	10310

شکل ۳-۷۶ بررسی دستور Full JOIN

۳-۱-۵-۱۹ بررسی دستور synonym

با دستور synonym می‌توانید برای جداول خود مترادف درست کنید، یعنی می‌توانید به جای نام اصلی جداول و View، یک نام جدید به آنها تخصیص دهید و در کد اجرا کنید؛ در زیر شکل کلی این دستور را مشاهده می‌کنید.

```

CREATE SYNONYM synonym_name
FOR object;

```

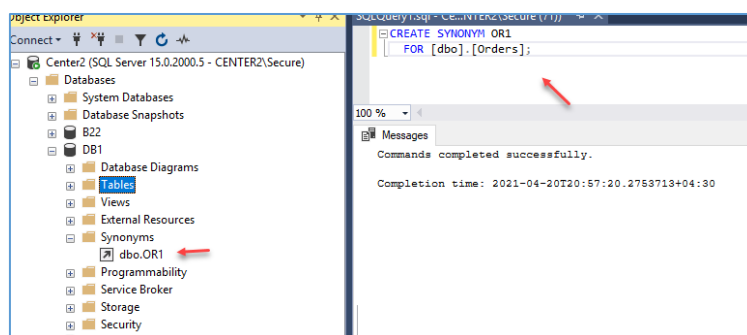
برای اینکه بیشتر با این دستور آشنا شویم، دستور زیر را در SQL اجرا می‌کنیم:

```

CREATE SYNONYM OR1
FOR [dbo].[Orders];

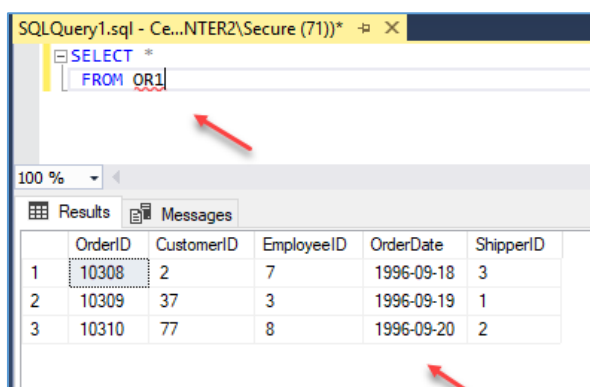
```

با استفاده از دستور CREATE SYNONYM، یک نام جدید با عنوان OR1 برای جدول Orders که در جلوی دستور For قرار دادیم ایجاد می‌شود، اگر به شکل ۳-۷۷ دقت کنید بعد از اجرای دستور در قسمت Synonyms، نام جدید شما ایجاد شده است و حالا می‌توانید به‌مانند جدول اصلی از آن استفاده کنید.



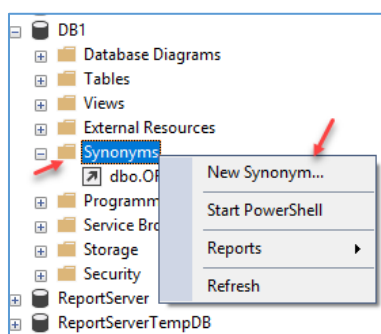
شکل ۳-۷۷ بررسی دستور Synonym

برای اینکه از این جدول خروجی تهیه کنید، می‌توانید به‌مانند شکل ۳-۷۸ عمل کنید؛ همان‌طور که مشاهده می‌کنید با وارد کردن نام OR1 در جلوی دستور FROM، لیست جدول ORDERS در خروجی چاپ شده است؛ یکی از مزایای استفاده از دستور SYNONYM، خلاصه کردن و مشخص کردن اسم مشخص برای جداول و کوتاه کردن دستورات است.



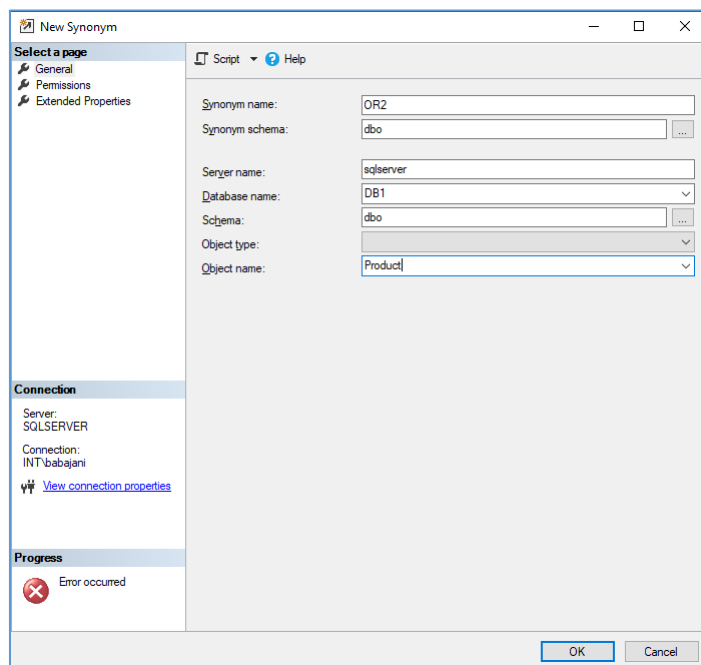
شکل ۳-۷۸ خروجی جدول

خوب اگر بخواهید از طریق GUI، یک SYNONYM ایجاد کنید باید به‌صورت شکل ۳-۷۹ بر روی Synonyms کلیک راست کنید و گزینه‌ی New Synonyms را انتخاب کنید.



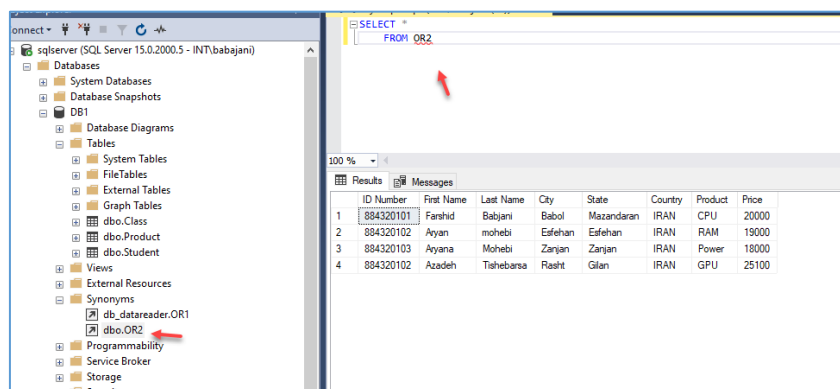
شکل ۳-۷۹ ایجاد Synonym

در شکل ۳-۸۰ و در قسمت Synonyms Name باید نام دلخواه خود را وارد کنید و در قسمت Synonyms schema باید نوع دسترسی آن را مشخص کنید که با وارد کردن dbo، یعنی دسترسی کامل به جدول خواهید داشت؛ در قسمت Database name باید نام دیتابیس را وارد کنید که قرار است جدول مورد نظر را از آن انتخاب کنید و در قسمت Object name نیز باید نام جدول مورد نظر را وارد و بر روی OK کلیک کنید.



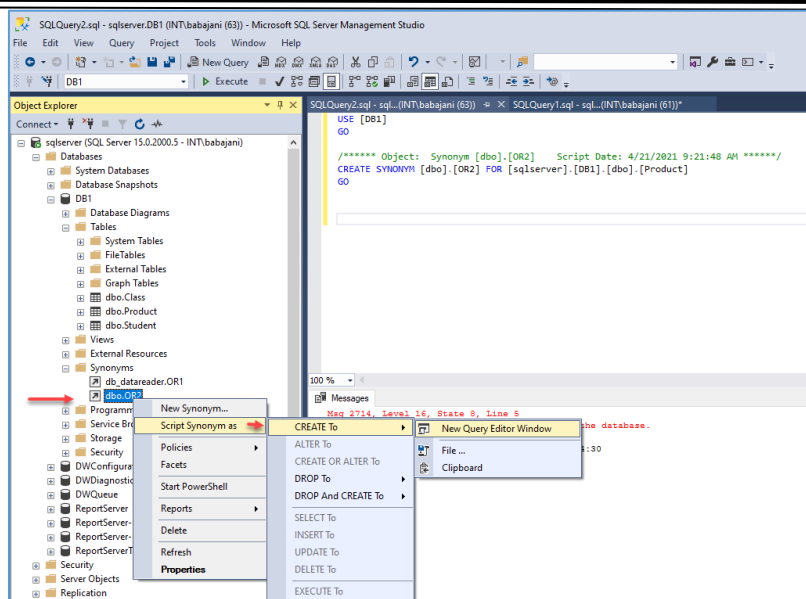
شکل ۳-۸۰ / ایجاد synonym

همان طور که در شکل ۳-۸۱ مشاهده می‌کنید، Synonyms مورد نظر ایجاد شده و خروجی آن نیز جدول Product است.



شکل ۳-۸۱ / خروجی جدول

اگر بخواهید بعد از ایجاد synonyms کد آن را به دست آورید، می‌توانید به‌مانند شکل ۳-۸۲ بر روی synonyms مورد نظر کلیک راست کنید و از قسمت Script و بعد، گزینه‌ی Create to New Query Editor را انتخاب کنید تا کد مورد نظر نمایش داده شود که البته بعد از اجرا به شما خطا خواهد داد که synonyms از قبل وجود دارد.



شکل ۱۲-۳ ایجاد کد Synonym

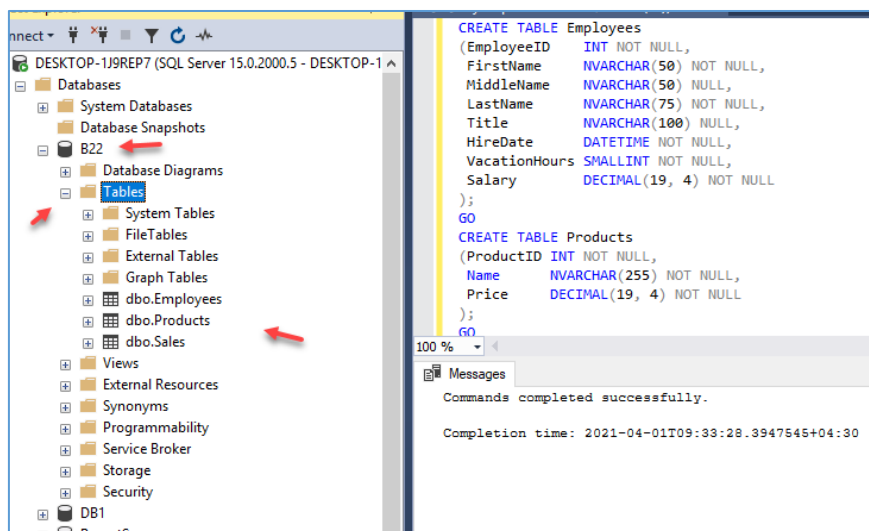
۶-۱-۳ کار با View در SQL

View ها یک سری از جداول هستند که می‌توانند به صورت دقیق‌تر و بهتر نتیجه‌ی دستور SELECT را در خروجی نمایش دهند، البته به این موضوع توجه کنید که VIEW شامل جداول مجازی هستند و ربطی به جدول اصلی در دیتابیس ندارند و به این دلیل هیچ چیزی را در خودشان ذخیره نمی‌کنند، یعنی تنها برای نمایش خروجی دستور SELECT است و اگر هم حذف شوند، تأثیری روی جداول اصلی ندارند. برای شروع کار نیاز به جدالی داریم تا بتوانیم View را بر روی آنها پیاده‌سازی کنیم؛ برای این کار دستور زیر را اجرا کنید:

```
CREATE TABLE Employees
(EmployeeID INT NOT NULL,
FirstName NVARCHAR(50) NOT NULL,
MiddleName NVARCHAR(50) NULL,
LastName NVARCHAR(75) NOT NULL,
Title NVARCHAR(100) NULL,
HireDate DATETIME NOT NULL,
VacationHours SMALLINT NOT NULL,
Salary DECIMAL(19, 4) NOT NULL
);
GO
CREATE TABLE Products
(ProductID INT NOT NULL,
Name NVARCHAR(255) NOT NULL,
Price DECIMAL(19, 4) NOT NULL
);
GO
CREATE TABLE Sales
(SalesID UNIQUEIDENTIFIER NOT NULL,
ProductID INT NOT NULL,
EmployeeID INT NOT NULL,
Quantity SMALLINT NOT NULL,
```

```
SaleDate DATETIME NOT NULL
);
GO
```

با اجرای دستورات بالا، سه جدول با نام‌های Employees, Products, Sales ایجاد می‌شود که در شکل ۳-۸۳ این موضوع را مشاهده می‌کنید.



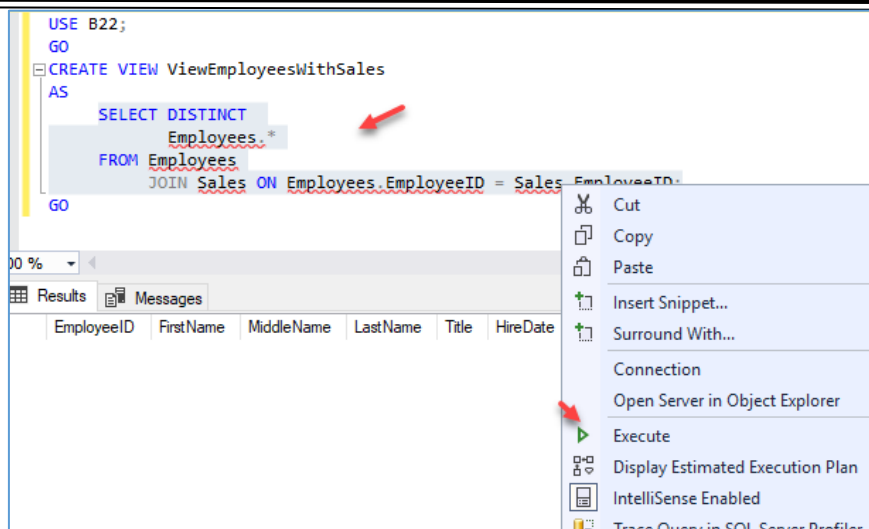
شکل ۳-۸۳ / ایجاد View

برای ایجاد View از دو روش می‌توانیم استفاده کنیم؛ می‌توانیم از دستورات T-SQL استفاده کنیم، یا اینکه از Management Studio استفاده کنیم.

برای ایجاد نمای جدید می‌توانیم از دستورات T-SQL زیر استفاده کنیم:

```
USE B22;
GO
CREATE VIEW ViewEmployeesWithSales
AS
    SELECT DISTINCT
        Employees.*
    FROM Employees
        JOIN Sales ON Employees.EmployeeID = Sales.EmployeeID;
GO
```

این یک نمای (view) ساده با یک عبارت ساده SELECT است که لیستی از کارمندان فروش را در جدول Employees و Sales برمی‌گرداند؛ در حقیقت، شما همیشه می‌توانید قبل از ایجاد نمایش، تنها با اجرای قسمت SELECT، عبارت CREATE VIEW SQL، عبارت جستجو را آزمایش کنید و بهتر است ببینید آیا پرسشی چیزی را برمی‌گرداند؛ برای اینکه یک قسمت از دستور در خروجی اجرا کنید باید به‌مانند شکل ۳-۸۴ دستور مورد نظر را انتخاب کنید و بعد، کلیک راست و گزینه‌ی Execute را انتخاب کنید که نتیجه‌ی آن مشخص خواهد شد؛ از آنجایی که جداول مقدار ندارند، یک جدول خالی را در شکل ۳-۸۴ مشاهده می‌کنید.



شکل ۳-۸۴ خروجی دستور

برای اینکه مقداری را در جداول وارد کنیم، می‌توانیم از دستورات زیر استفاده کنیم:

```
USE B22;
GO
```

```
INSERT INTO Employees VALUES (1, 'Ken', NULL, 'Farshid', 'IT Manager', '1/1/2016',
2080, 45000);
```

```
INSERT INTO Employees VALUES (2, 'Janice', NULL, 'AZADEH', 'Sales Representative',
'12/11/2016', 2080, 45000);
```

```
INSERT INTO Products VALUES (1, 'Long-Sleeve Logo Jersey, S', 12.99);
```

```
INSERT INTO Products VALUES (2, 'Long-Sleeve Logo Jersey, M', 14.99);
```

```
INSERT INTO Products VALUES (3, 'Long-Sleeve Logo Jersey, L', 16.99);
```

```
INSERT INTO Products VALUES (4, 'Long-Sleeve Logo Jersey, XL', 18.99);
```

```
INSERT INTO Sales VALUES (NEWID(), 1, 1, 4, '04/15/2016');
```

```
INSERT INTO Sales VALUES (NEWID(), 2, 1, 1, '02/01/2016');
```

```
INSERT INTO Sales VALUES (NEWID(), 3, 1, 2, '03/12/2016');
```

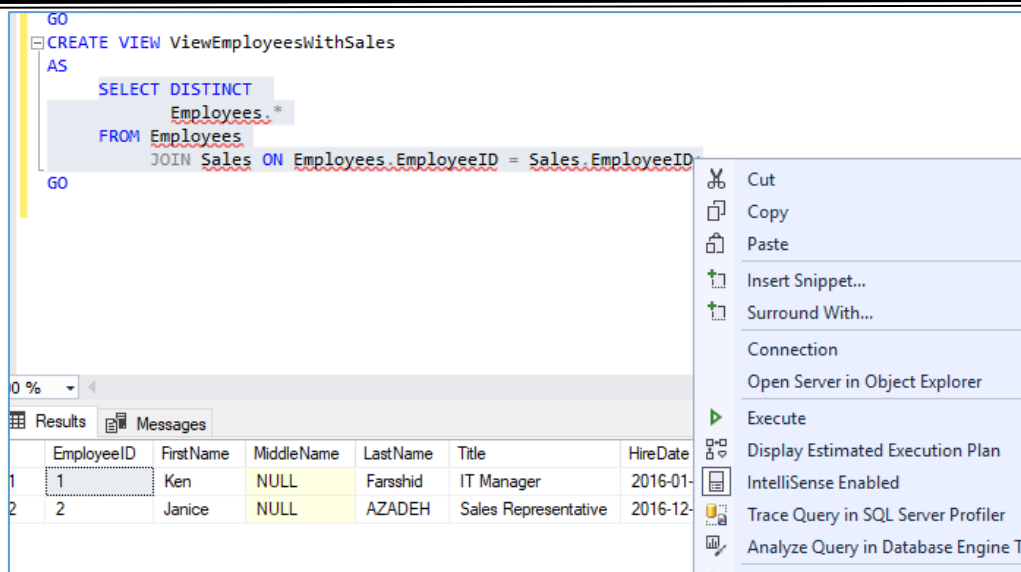
```
INSERT INTO Sales VALUES (NEWID(), 2, 2, 2, '03/18/2016');
```

```
INSERT INTO Sales VALUES (NEWID(), 3, 2, 1, '04/16/2016');
```

```
INSERT INTO Sales VALUES (NEWID(), 4, 2, 2, '04/23/2016');
```

در خط اول دستور باید نام دیتابسی که جداول در آن قرار دارند را وارد کنید، یعنی به جای B22، نام دیتابیس خود را بنویسید.

در ادامه برای تست موضوع باید دوباره قسمت SELECT را به‌مانند شکل ۳-۸۵ انتخاب کنید و دستور Execute را اجرا کنید؛ توجه داشته باشید در دستورات از DISTINCT برای جلوگیری از نمایش سوابق تکراری در خروجی استفاده کردیم.

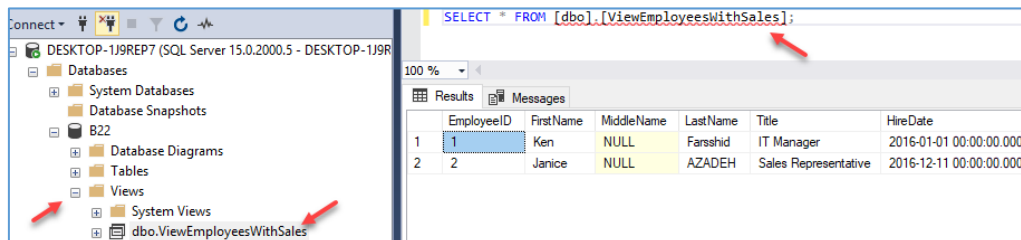


شکل ۳-۱۰۳ خروجی دستور

بعد از اینکه View را ایجاد کردید، می‌توانید به‌مانند شکل ۳-۱۰۴ از سمت چپ وارد قسمت Views شوید و نمای مورد نظر که ایجاد کردید را مشاهده کنید؛ با دستور زیر می‌توانید خروجی این View را مشاهده کنید:

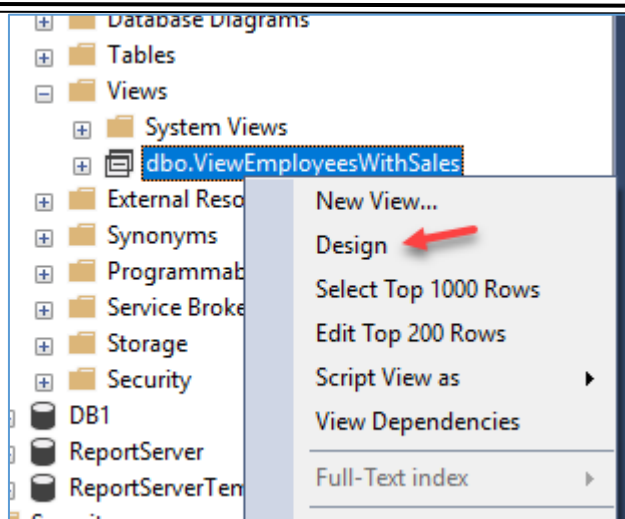
```
SELECT * FROM [dbo].[ViewEmployeesWithSales];
```

توجه داشته باشید به جای اینکه در دستور بالا، نام View را بنویسید، می‌توانید View مورد نظر را بکشید و در دستور رها کنید تا خودش نوشته شود.



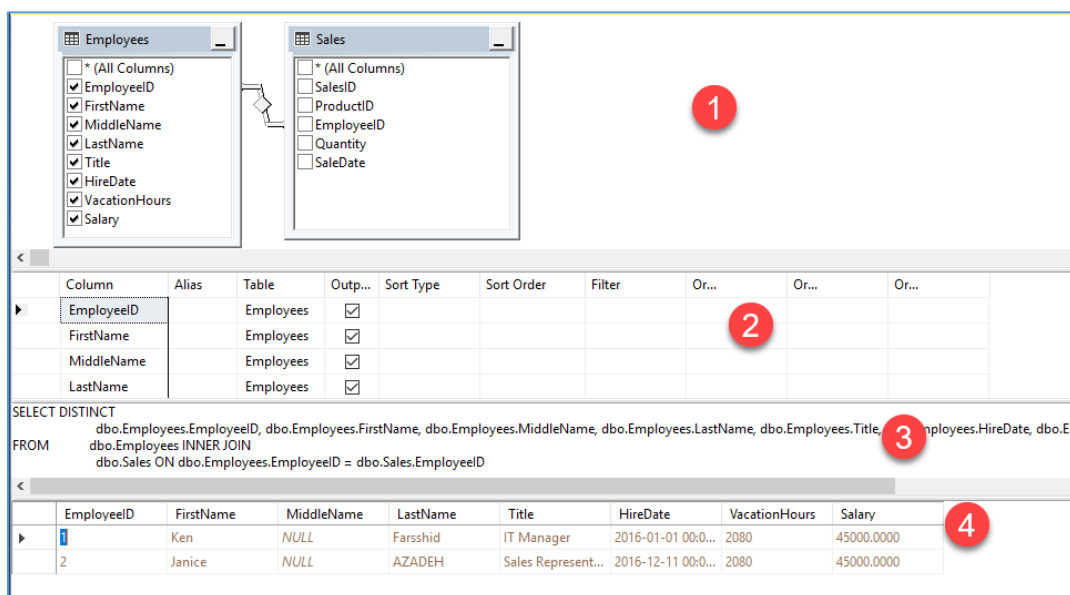
شکل ۳-۱۰۴ خروجی View

در ادامه برای اینکه ارتباط بین جداول را در View مورد نظر به‌صورت گرافیکی مشاهده کنید باید به‌مانند شکل ۳-۱۰۵ بر روی View مورد نظر کلیک راست کنید و گزینه‌ی Design را انتخاب کنید؛ توجه داشته باشید که شما می‌توانستید به‌جای وارد کردن دستورات از همین قسمت با انتخاب New View، نمای مورد نظر خود را انتخاب کنید.



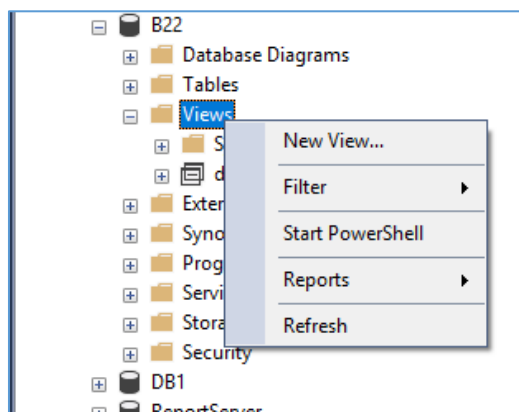
شکل ۳-۱۰۵ بررسی دستور Design

همان طور که در شکل ۳-۱۰۶ مشاهده می‌کنید، این صفحه از چهار قسمت تشکیل شده است؛ در قسمت شماره‌ی نمای کلی، دو جدول را مشاهده می‌کنید؛ در قسمت شماره‌ی دو طبق ستون‌هایی که در دستور قبلی وارد کردیم، ستون‌های مورد نظر تیک خورده شدند؛ در قسمت سوم دستور خروجی SELECT را که قبلاً وارد کردیم را مشاهده می‌کنید؛ کلاً در قسمت شماره‌ی یک، اگر هر یک از ستون‌ها را انتخاب کنید در قسمت شماره‌ی سه دستورات تغییر خواهد کرد؛ در قسمت شماره‌ی چهار نیز خروجی دستور شماره‌ی سوم را مشاهده می‌کنید، البته باید دستور شماره‌ی سوم را انتخاب و کلیک راست کنید و گزینه‌ی Execute را انتخاب کنید؛ توجه داشته باشید در قسمت شماره‌ی یک، پیوند بین جدول اول و دوم ایجاد شده است که آن نیز به دلیل دستور `dbo.Employees.EmployeeID =` `dbo.Sales.EmployeeID` که در قسمت شماره‌ی سوم وارد کردید، است؛ به جای = اگر > قرار دهید شکل آن در قسمت شماره‌ی یک تغییر خواهد کرد.



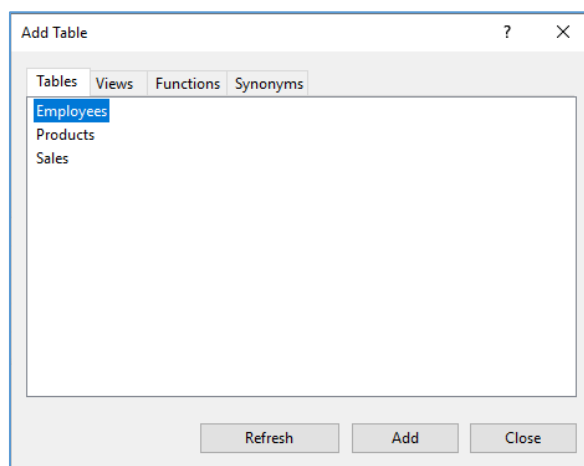
شکل ۳-۱۰۶ ایجاد VIEW

اگر بخواهید از طریق محیط گرافیکی، View ایجاد کنید باید به‌مانند شکل ۳-۱۰۷ بر روی View کلیک راست کنید و گزینه‌ی New View را انتخاب کنید.



شکل ۳-۱۰۷ / ایجاد View

بعد از ورود، شکل ۳-۱۰۸ ظاهر خواهد شد که باید انتخاب کنید که از چه جداولی می‌خواهید خروجی بگیرید؛ در این قسمت، هر سه جدول را انتخاب می‌کنیم، البته می‌توانیم از تب View نیز استفاده کنیم.



شکل ۳-۱۰۸ / ایجاد view

در شکل ۳-۱۰۹ و در قسمت اول می‌توانید ستون‌هایی که قرار است در خروجی چاپ شود را انتخاب کنید؛ به این نکته توجه کنید که انتخاب باید به ترتیب باشد، چون هر انتخابی که در جداول انجام می‌دهید، در قسمت دوم ردیف آن مشخص می‌شود و در قسمت سوم نیز دستورات آن به‌صورت اتوماتیک نوشته خواهد شد؛ کلاً View همان دستور SELECT است که قبلاً با هم روی آن کار کردیم.

Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter	Or...	Or...	Or...
EmployeeID		Employees	<input checked="" type="checkbox"/>						
FirstName		Employees	<input checked="" type="checkbox"/>						
LastName		Employees	<input checked="" type="checkbox"/>						
Price		Products	<input checked="" type="checkbox"/>						

```

SELECT
FROM
    dbo.Employees.EmployeeID, dbo.Employees.FirstName, dbo.Employees.LastName, dbo.Products.Price, dbo.Sales.ProductID, dbo.Sales.Quantity
    dbo.Employees CROSS JOIN
    dbo.Products CROSS JOIN
    dbo.Sales
  
```

EmployeeID	FirstName	LastName	Price	ProductID	Quantity
1	Ken	Farsshid	12.9900	1	4
1	Ken	Farsshid	12.9900	2	1
1	Ken	Farsshid	12.9900	3	2
1	Ken	Farsshid	12.9900	2	2

شکل ۱۰۹ - ۳ ایجاد View

برای اینکه بتوانیم شرط در این دستورات قرار دهیم، می‌توانیم به‌مانند شکل ۱۱۰ - ۳ در قسمت دوم و در ستون فیلتر، شرط خود را وارد کنیم که در اینجا > 2 وارد شده است، یعنی اینکه در خروجی کار سطرهایی که EmployeeID آنها بزرگ‌تر از 2 باشد در خروجی چاپ خواهد شد؛ توجه کنید زمانی که در قسمت فیلتر، شرط مورد نظر خود را وارد می‌کنید، بعد از خروج از آن، قسمت دستور آن در قسمت شماره‌ی سوم نوشته می‌شود که در شکل ۱۱۰ - ۳ مشخص شده است.

برای اینکه دستوری که ایجاد شده را اجرا کنیم باید آن را اجرا کنیم و کلیک راست کنیم و گزینه‌ی Execute SQL را انتخاب کنیم تا در قسمت چهارم، خروجی مشخص شود که همان‌طور که گفتیم خروجی برای EmployeeID هایی که بزرگ‌تر از دو باشد، مشخص شده است.

در ستون‌های Sort نیز می‌توانید مشخص کنید که خروجی به چه صورت تنظیم شود.

Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter	Or...	Or...	Or...
EmployeeID		Employees	<input checked="" type="checkbox"/>			> 1			
FirstName		Employees	<input checked="" type="checkbox"/>						
LastName		Employees	<input checked="" type="checkbox"/>						
Price		Products	<input checked="" type="checkbox"/>						

```

SELECT TOP (100) PERCENT
FROM
    dbo.Employees.EmployeeID, dbo.Employees.FirstName, dbo.Employees.LastName,
    dbo.Products CROSS JOIN
    dbo.Sales
WHERE
    (dbo.Employees.EmployeeID > 1)
  
```

EmployeeID	FirstName	LastName	Price	ProductID	Quantity
2	Janice	AZADEH	12.9900	1	4
2	Janice	AZADEH	12.9900	2	1
2	Janice	AZADEH	12.9900	3	2
2	Janice	AZADEH	12.9900	2	2

شکل ۱۱۰ - ۳ اجرای دستور

۳-۱-۷ بررسی FileStream در SQL Server

در نرم‌افزار SQL قابلیت طراحی شده که بتوانید فایل‌هایی با حجم بالا را در آن ذخیره کنید. همیشه این طور نیست که فقط داده‌های متنی در دیتابیس ذخیره شوند، بلکه داده‌هایی مانند: عکس، موسیقی، فیلم نیز می‌توانند در SQL ذخیره شوند، اما برای این کار باید از قابلیت FileStream در SQL استفاده کنید؛ در نسخه‌های قدیمی SQL افراد برای ذخیره‌ی عکس، موسیقی و فیلم از داده‌هایی از نوع BLOB استفاده می‌کردند که این داده‌ها تا حداکثر ۲ گیگابایت را پشتیبانی می‌کرد و بیشتر از این عدد، دیگر نمی‌توانستند ذخیره کنند و یکی از مهم‌ترین مشکلات این نوع داده‌ها این بود که عملکرد سیستم را به شدت کاهش می‌داد و باعث کندی آن می‌شد، اما با استفاده از FileStream دیگر نگران مقدار فضا و کندی سرعت نرم‌افزار نخواهیم بود.

برای اینکه از FILESTREAM استفاده کنیم باید یک گروه برای آن ایجاد کنیم تا اطلاعات در گروه جدید قرار بگیرد و دیگر با گروه اصلی یا همان Primary کاری نداریم؛ برای استفاده از این قابلیت در جداول نیاز به یک رکورد با نوع varbinary داریم.

در شکل ۳-۱۱۱ نوع داده‌های قدیمی برای ذخیره عکس را مشاهده می‌کنید که از یک دیتابیس استفاده می‌کردند، اما در عکس ۳-۱۱۲ این موضوع از دیتابیس اصلی جدا شده است.

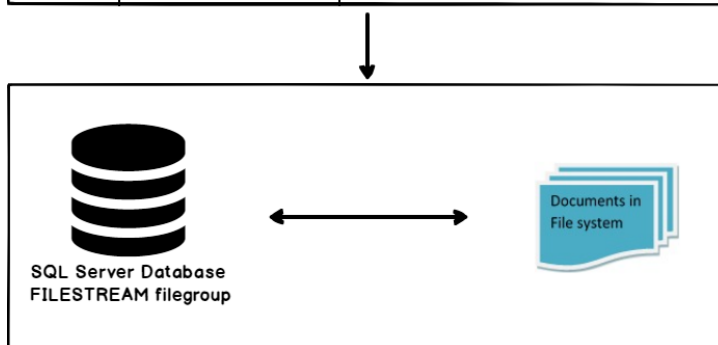
EmpID (int)	EmpName(varchar(100))	EmpPhoto(Varbinary(max))
1	Rajendra	0*E695DD565AC4566
2	Sonu	0*F500045GN68BO47



SQL Server Database
Primary FileGroup

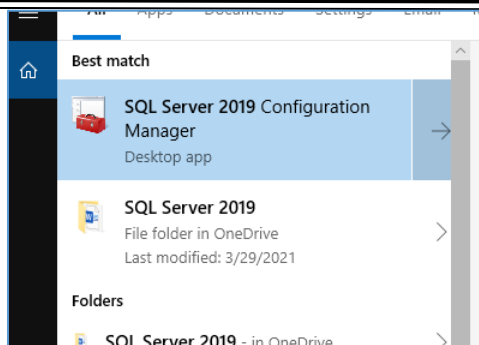
شکل ۳-۱۱۱ بررسی FILESTREAM

EmpID (int)	EmpName(varchar(100))	EmpPhoto(Varbinary(max)) FILESTREAM
1	Rajendra	0*E695DD565AC4566
2	Sonu	0*F500045GN68BO46



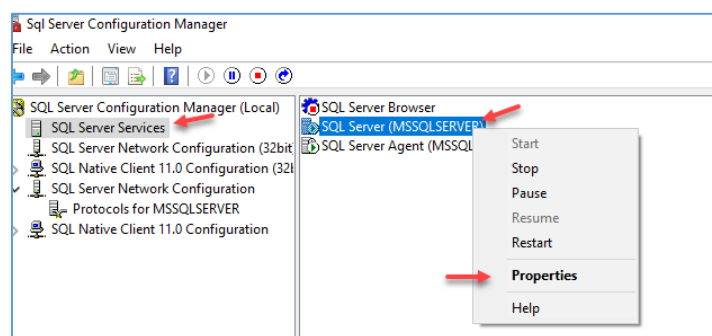
شکل ۳-۱۱۲ بررسی FILESTREAM

برای فعال‌سازی قابلیت FILESTREAM باید به‌مانند شکل ۱۱۳ - ۳ سرویس SQL Server 2019 Configuration Manager را اجرا کنید.



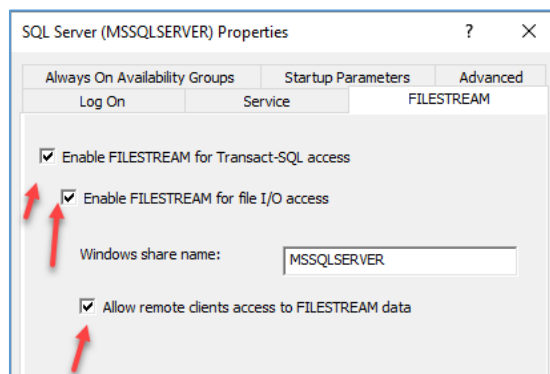
شکل ۳-۱۱۳ اجرای سرویس Configuration

در شکل ۳-۱۱۴ از سمت چپ بر روی SQL Server Services کلیک کنید و در صفحه‌ی باز شده بر روی SQL Server کلیک راست کنید و گزینه‌ی Properties را انتخاب کنید.



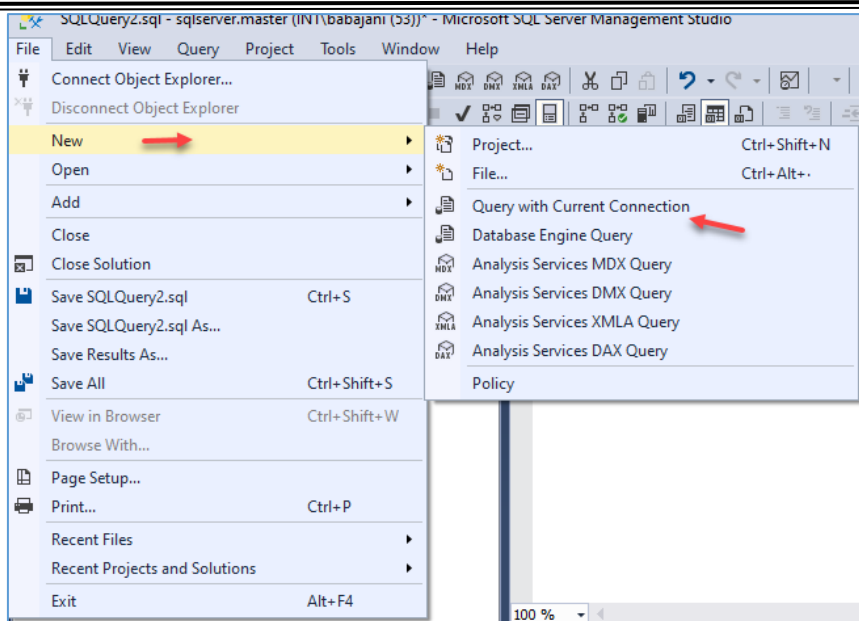
شکل ۳-۱۱۴ بررسی سرویس

در شکل ۳-۱۱۵ وارد تب FILESTREAM شوید و تیک هر سه گزینه را انتخاب کنید؛ گزینه‌ی اول برای دسترسی از طریق دستورات T-SQL است؛ گزینه‌ی دوم برای خواندن و نوشتن از ورودی و خروجی پرونده‌ها است و برای این کار باید یک نام برای به اشتراک‌گذاری آن وارد کنید و گزینه‌ی آخر، اجازه‌ی دسترسی از راه دور به FILESTREAM را می‌دهد.



شکل ۳-۱۱۵ بررسی FileStream

برای اینکه FileStream را در SQL فعال کنیم باید تغییری در تنظیمات آن انجام دهیم؛ برای این کار وارد منوی فایل به‌مانند شکل ۳-۱۱۶ شوید و از قسمت New بر روی Query کلیک کنید.



شکل ۳-۱۱۶ بررسی FileStream

دستور زیر را اجرا کنید:

```
USE master
GO
EXEC sp_configure
GO
```

این دستور تنظیمات کلی سرور SQL را به شما نشان می‌دهد که مربوط به قسمت Advanced است؛ در این قسمت که در شکل ۳-۱۱۷ مشخص شده است باید به قسمت File Stream Access Level مراجعه کنید و در قسمت ستون config_value عدد ۲ قرار دهید؛ با این کار دسترسی کامل به سرویس FILESTREAM فعال خواهد شد.

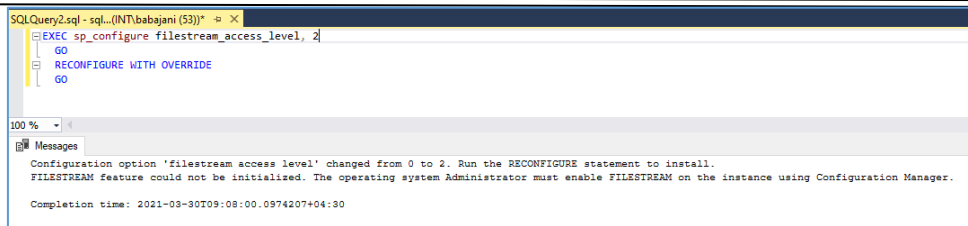
6	column encryption enclave type	0	2	0	0
7	contained database authentication	0	1	1	1
8	cross db ownership chaining	0	1	0	0
9	default language	0	9999	0	0
10	external scripts enabled	0	1	0	0
11	filestream access level	0	2	0	0
12	hadoop connectivity	0	7	0	0
13	max text repl size (B)	-1	2147483647	65536	65536
14	nested triggers	0	1	1	1
15	polybase enabled	0	1	0	0
16	polybase network encryption	0	1	1	1
17	remote access	0	1	1	1
18	remote admin connections	0	1	0	0
19	remote data archive	0	1	0	0

شکل ۳-۱۱۷ بررسی FileStream

برای قراردادن عدد ۲ باید دستور زیر را وارد کنید.

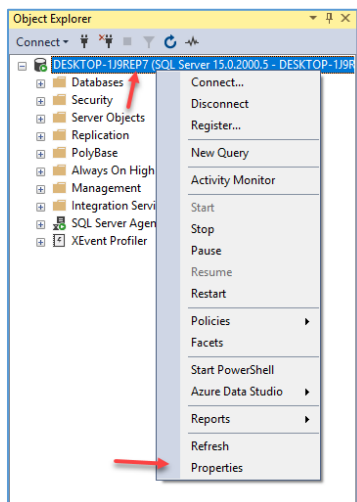
```
EXEC sp_configure filestream_access_level, 2
GO
RECONFIGURE WITH OVERRIDE
GO
```

در دستور بالا، قسمت config_value مربوط به filestream_access_level به عدد ۲ تغییر می‌کند که در شکل ۳-۱۱۸ این موضوع را مشاهده می‌کنید.



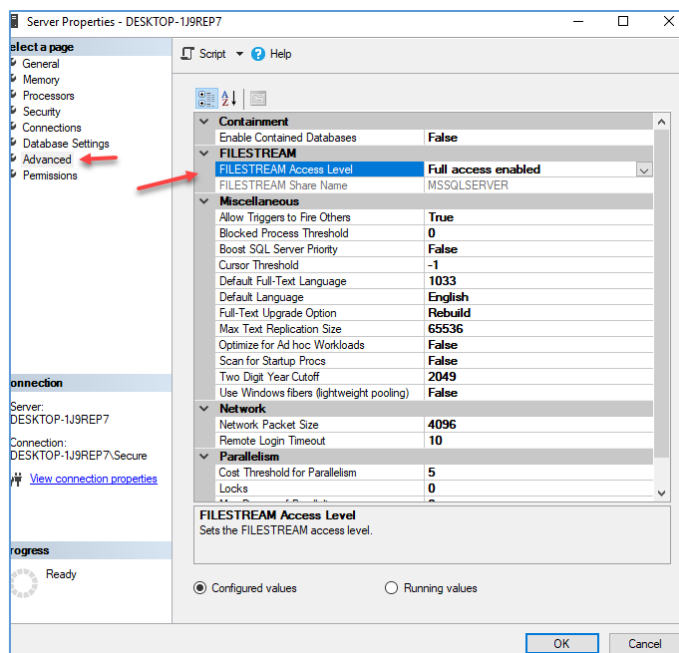
شکل ۱۱۸- بررسی FileStream

برای اینکه متوجه شویم که کار به درستی انجام شده است یا نه باید به مانند شکل ۱۱۹-۳ بر روی سرور SQL کلیک راست کنید و وارد قسمت Properties شوید.



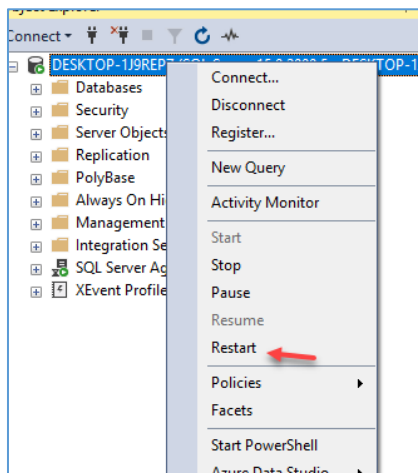
شکل ۱۱۹- بررسی FileStream

در شکل ۱۲۰-۳ از سمت چپ وارد قسمت Advanced شوید و قسمت FILESTREAM Access Level را مشاهده کنید که بر روی FULL access enabled قرار گرفته است، البته شما می‌توانستید دستورات قبلی را وارد نکنید و مستقیم در این قسمت، این گزینه را انتخاب کنید.



شکل ۱۲۰- بررسی FileStream

بعد از اتمام کار، حتماً باید سرور را Restart کنید؛ برای این کار باید به‌مانند شکل ۳-۱۲۱ بر روی سرور SQL کلیک راست کنید و گزینه‌ی Restart را انتخاب کنید تا تنظیمات به درستی اعمال شود.



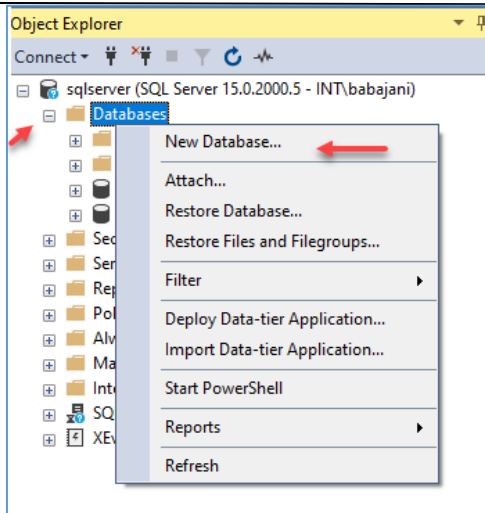
شکل ۳-۱۲۱ بررسی FileStream

نکاتی که قبل از استفاده از FILESTREAM باید بدانیم شامل:

- ۱- می‌توانیم از عبارات SELECT، INSERT، UPDATE و DELETE مشابهی پرس‌وجو استاندارد پایگاه‌داده در FILESTREAM استفاده کنیم.
- ۲- اگر اندازه‌ی شی یا Object بیشتر از ۱ مگابایت باشد باید از FILESTREAM استفاده کنیم.
- ۳- هر سطر باید یک ID ردیف منحصربه‌فرد داشته باشد تا از این قابلیت استفاده کند و نباید حاوی مقادیر NULL باشد.
- ۴- نمی‌توانیم داده‌های FILESTREAM را رمزگذاری کنیم.

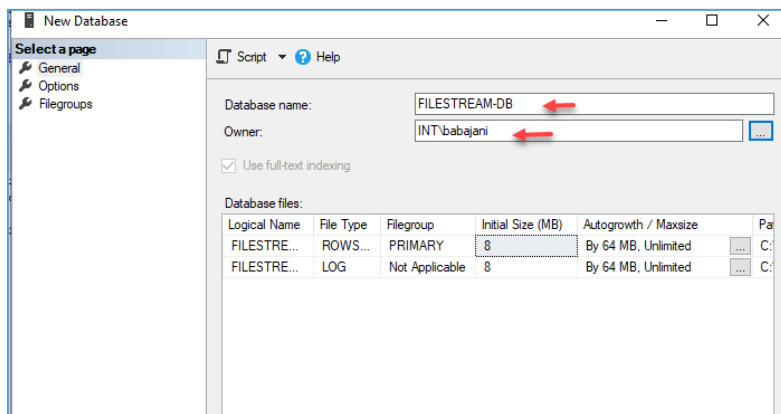
۳-۱-۷-۱ ایجاد دیتابیس برای استفاده از FILESTREAM

برای اینکه بتوانیم از قابلیت FILESTREAM استفاده کنیم باید یک دیتابیس جدید ایجاد و بر روی آن کار کنیم، البته می‌توانیم از دیتابیس‌های قدیمی خود استفاده کنیم؛ به‌مانند شکل ۳-۱۲۲ بر روی Databases کلیک راست کنید و گزینه‌ی New Databases را انتخاب کنید.



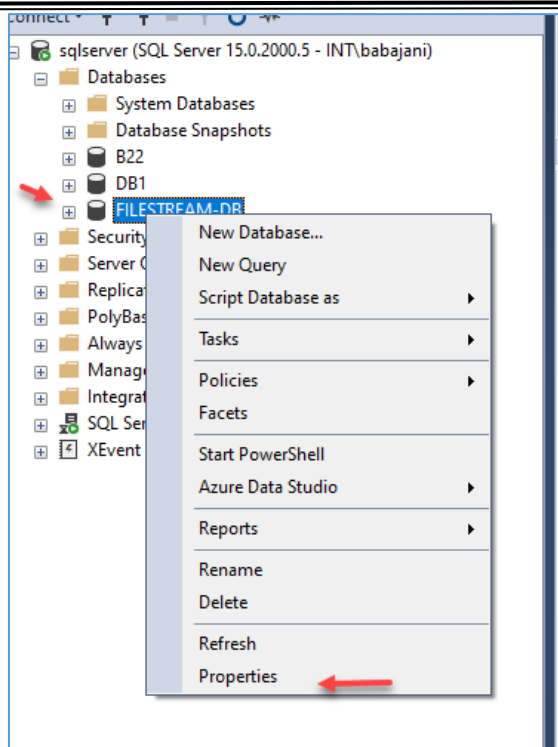
شکل ۱۲۲-۳ ایجاد FileStream

به‌مانند شکل ۱۲۳-۳، یک نام برای دیتابیس خود وارد کنید که در اینجا، FILESTREAM-DB وارد شده است، بعد از این کار بر روی ADD کلیک کنید تا دیتابیس مورد نظر ایجاد شود.



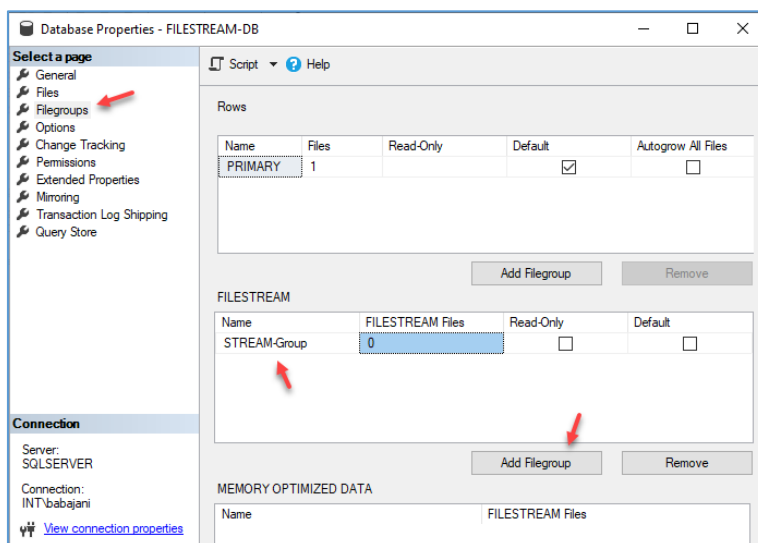
شکل ۱۲۳-۳ ایجاد FileStream

بعد از ایجاد دیتابیس، به‌مانند شکل ۱۲۴-۳ بر روی آن کلیک راست کنید و گزینه‌ی Properties را انتخاب کنید.



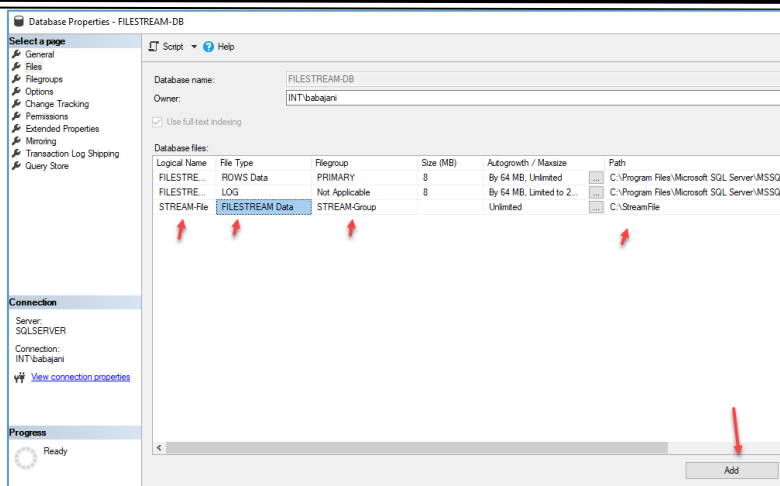
شکل ۱۲۴-۳ ایجاد FileStream

در شکل ۱۲۵-۳ باید وارد FileGroup شوید و در قسمت FILESTREAM بر روی Add کلیک کنید و نام گروه مورد نظر خود را برای تخصیص به FILESTREAM وارد کنید.



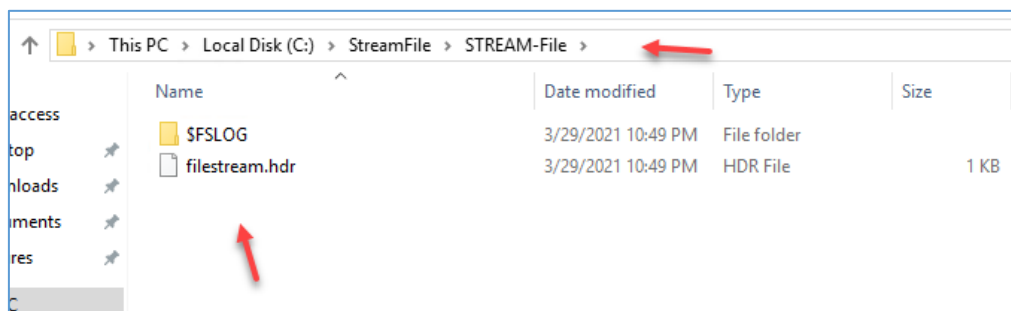
شکل ۱۲۶-۳ ایجاد FileStream

در ادامه‌ی کار به‌مانند شکل ۱۲۷-۳ وارد قسمت Files شوید و بر روی Add کلیک کنید؛ در ستون Logical Name نام به‌دلخواه وارد کنید؛ در ستون File Type نوع فایل را FILESTREAM Data در نظر بگیرید و Filegroup را نیز انتخاب کنید، البته به‌صورت اتوماتیک خودش انتخاب خواهد شد؛ در قسمت Path نیز باید مسیری را انتخاب کنید که از ظرفیت خوبی برخوردار باشد؛ در آخر بر روی OK کلیک کنید تا تنظیمات ذخیره شود.



شکل ۱۲۷-۳ ایجاد FileStream

بعد از انجام تنظیمات، اگر به مانند شکل ۱۲۸-۳ وارد آدرسی مربوط به Stream شوید، مشاهده خواهید کرد که فایل STREAM در این مسیر ایجاد شده است.



شکل ۱۲۸-۳ ایجاد FileStream

۲-۷-۱-۳ ایجاد جدول در دیتابیس FILESTREAM

برای اینکه یک جدول با داده‌های FILESTREAM داشته باشیم باید به صورت زیر عمل کنیم؛ همان طور که گفتیم برای استفاده از FILESTREAM باید داده از نوع VARBINARY باشد که در دستور زیر مشخص شده است و یکی دیگر از شرط‌های FILESTREAM این است که حتماً باید یک کلید منحصر به فرد داشته باشد و حتماً نیز باید NOT NULL باشد؛ در دستور زیر سه فیلد ایجاد شده است که در Fileid نوع داده، UNIQIDENTIFIER و NOT NULL است؛ داده‌ی FileName نیز از نوع معمولی VARCHAR با حداکثر کاراکتر ۲۵ است و FILE نیز از نوع VARBINARY است.

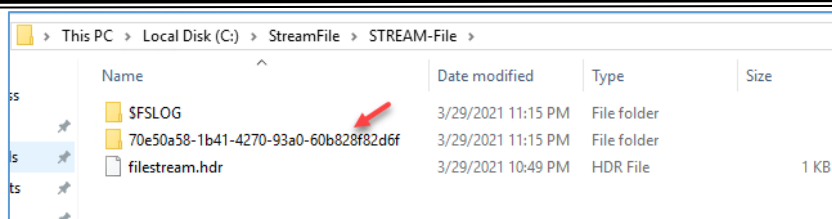
```
Use [FILESTREAM-DB]
```

```
Go
```

```
CREATE TABLE [FileStreamTable_1] (
  [FileId] UNIQUEIDENTIFIER ROWGUIDCOL NOT NULL UNIQUE,
  [FileName] VARCHAR (25),
  [File] VARBINARY (MAX) FILESTREAM);
```

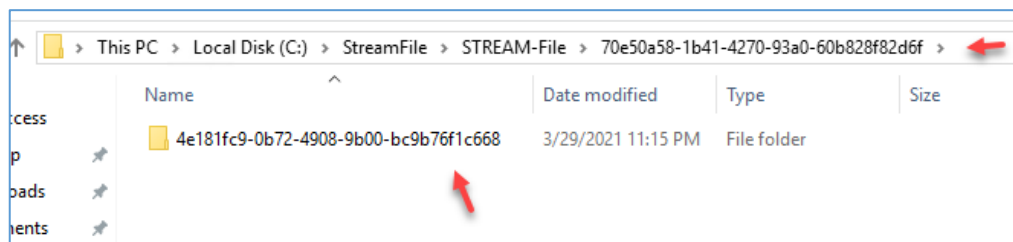
```
GO
```

بعد از اجرای دستور در مسیر مشخص شده که در شکل ۱۲۹-۳ مشخص شده است، یک پوشه با مقدار GUID ایجاد شده است.



شکل ۳-۱۲۹ بررسی فایل FileStream

اگر وارد این پوشه در شکل ۳-۱۳۰ شوید، یک پوشه‌ی دیگر با GUID جدید مشاهده می‌کنید که مربوط به ستون FILESTREAM در جدول جدید است؛ این موضوع را در شکل ۳-۱۳۰ مشاهده می‌کنید.



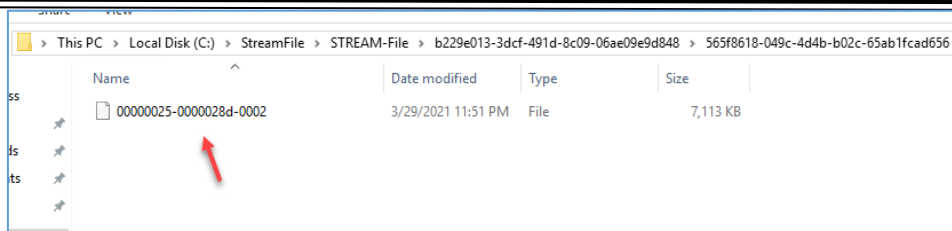
شکل ۳-۱۳۰ بررسی فایل FileStream

برای اینکه بیشتر با FILESTREAM کار کنیم، می‌خواهیم یک فایل موسیقی را درون دیتابیس مورد نظر ذخیره کنیم؛ اولین کاری که باید انجام دهید این است که فایل موسیقی را در مسیر C:\Music ذخیره کنیم، بعد از آن باید از دستور زیر استفاده کنید؛ در این دستور، یک جدول جدید با نام FileStreamTable_2 ایجاد می‌شود و در ادامه با دستور DECLARE، متغیر FILE با نوع varbinary صدا زنده می‌شود، فایل MP3 در آن قرار می‌گیرد.

```
Use [FILESTREAM-DB]
Go
CREATE TABLE [FileStreamTable_2] (
[FileId] UNIQUEIDENTIFIER ROWGUIDCOL NOT NULL UNIQUE,
[FileName] VARCHAR (25),
[File] VARBINARY (MAX) FILESTREAM);
GO
DECLARE @File varbinary(MAX);
SELECT
@File = CAST(
bulkcolumn as varbinary(max)
)
FROM
OPENROWSET(BULK 'c:\music\a.mp3', SINGLE_BLOB) as MyData;

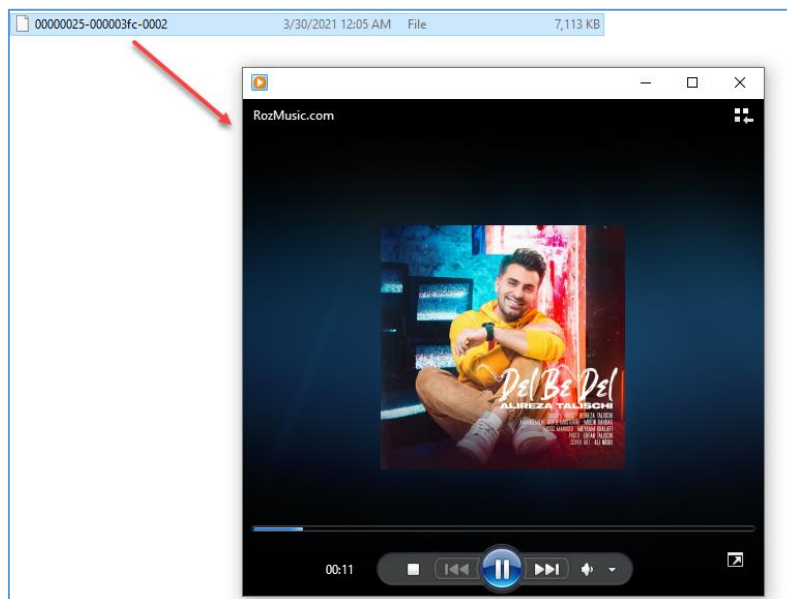
INSERT INTO FileStreamTable_2
VALUES
(
NEWID(),
'Sample Music',
@File
)
```

اگر به شکل ۳-۱۳۱ توجه کنید، یک فایل جدید به حجم تقریبی ۷ مگابایت که همان فایل MP3 است در مسیر مورد نظر ایجاد شده است.



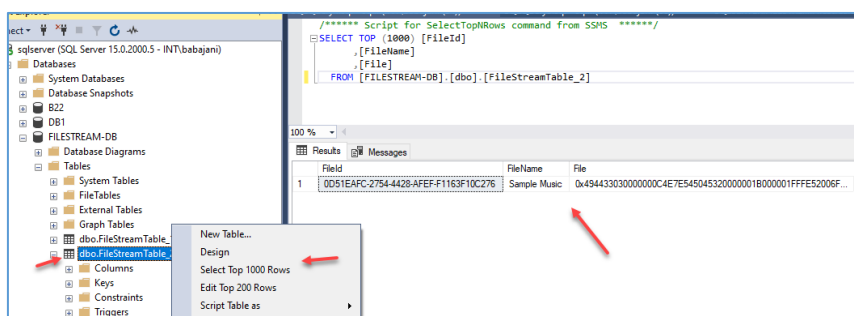
شکل ۳-۱۳۱ بررسی فایل FileShare

توجه داشته باشید، اگر فایل موسیقی در شکل ۳-۱۳۱ را در نرم‌افزار مورد نظر قرار دهید به راحتی اجرا خواهد شد؛ این موضوع را در شکل ۳-۱۳۲ مشاهده می‌کنید.



شکل ۳-۱۳۲ بررسی فایل FileShare

اگر وارد SQL شوید و بر روی جدول مورد نظر کلیک راست کنید و گزینه **Select Top 1000 Rows** را انتخاب کنید، خروجی جدول مورد نظر را نشان می‌دهد که در شکل ۳-۱۳۳ مشخص شده است.



شکل ۳-۱۳۳ بررسی فایل FileShare

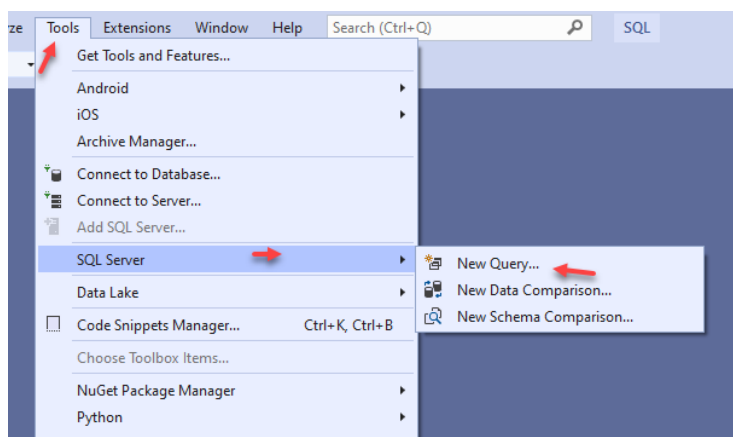
۳-۱-۸ ارتباط با SQL از طریق Visual Studio

یکی از ابزارهای مهم در صنعت برنامه‌نویسی و پایگاه‌داده، نرم‌افزار عالی Visual Studio است که ما را در ارائه‌ی راحت‌تر و بهتر کار یاری می‌کند؛ برای اینکه از آخرین نسخه‌ی این نرم‌افزار استفاده کنید، می‌توانید از لینک زیر آن را دانلود کنید:

<https://dl2.soft98.ir/programing/Microsoft.VisualStudio.2019.16.9.1.html>

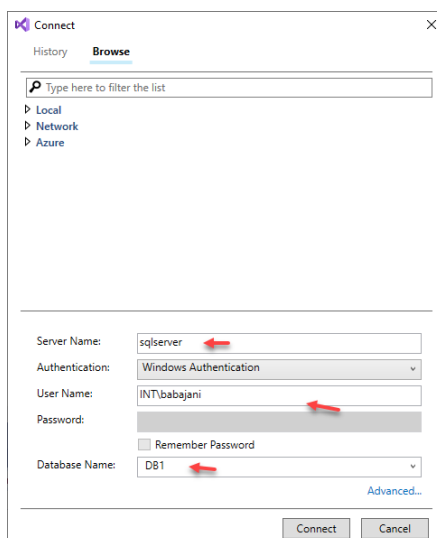
بعد از دانلود، آن را بر روی سیستم خود نصب کنید؛ توجه داشته باشید برای نصب کامل این نرم‌افزار حداقل نیاز به ۵۰ گیگابایت فضای هارد دارید که واقعاً فضای زیادی را اشغال می‌کند.

بعد از اجرای نرم‌افزار، اولین کاری که انجام می‌دهیم این است که از طریق Visual Studio به SQL Server متصل شویم و Query خود را اجرا کنیم؛ برای این کار به‌مانند شکل ۳-۱۳۴ وارد منوی Tools شوید و از قسمت SQL Server، گزینه‌ی New Query را انتخاب کنید.



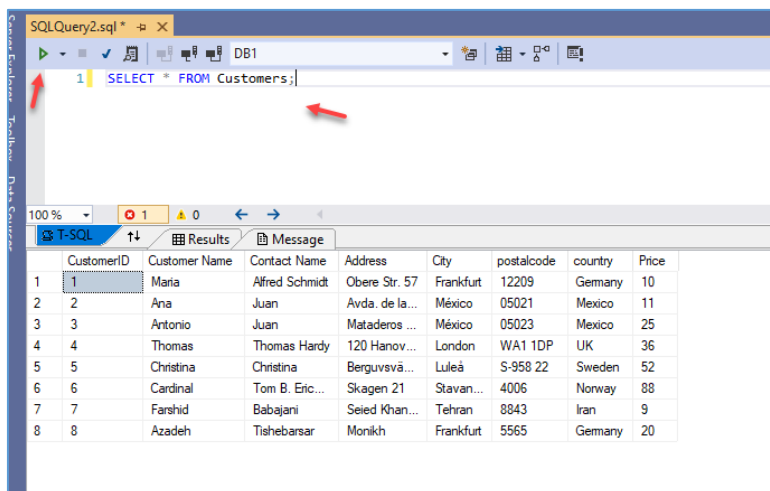
شکل ۳-۱۳۴ بررسی Visual Studio

در شکل ۳-۱۳۵ باید در قسمت Server name نام سرور SQL را وارد کنید و بعد از آن، مشخص کنید که نوع احراز هویت به چه صورت باشد، اگر با همان کاربری که در حال کار با visual studio هستید، می‌خواهید با SQL ارتباط برقرار کنید، می‌توانید Windows Authentication را انتخاب کنید و یا اگر می‌خواهید با کاربران موجود در SQL وارد آن شوید باید SQL Authentication را انتخاب کنید؛ در قسمت آخر نیز باید نام دیتابیس خود را انتخاب کنید و بر روی Connect کلیک کنید.



شکل ۳-۱۳۵ بررسی Visual Studio

همان طور که در شکل ۳-۱۳۶ مشاهده می‌کنید با استفاده از دستور SELECT توانستیم جدول Customers را در خروجی چاپ کنیم؛ این بدان معنا است که از طریق Visual Studio توانستیم با SQL Server ارتباط برقرار کنیم.



شکل ۳-۱۳۷ بررسی Visual Studio

۳-۱-۸-۱ ایجاد دیتابیس از طریق دستورات در Visual Studio

برای اینکه کار خود را گسترش دهیم، می‌خواهیم از طریق خط فرمان، یک دیتابیس ایجاد و در آن جدول دلخواه خود را ایجاد کنیم؛ برای این کار باید در query مورد نظر در Visual Studio دستورات زیر را وارد کنیم:

```
CREATE DATABASE B22
GO
USE B22
CREATE TABLE B22T (
    IDNumber int identity primary key,
    FirstName varchar(50),
    LastName varchar(50),
    City varchar(10),
    Country varchar(10),
    Address varchar(50)
);
INSERT INTO B22T(FirstName, LastName, City, Country, Address) VALUES
('ahmad', 'Mohamad', 'babol', 'IR', 'Kordmahaleh'),
('Azadeh', 'Mohebi', 'Shiraz', 'IR', 'Shiraz1'),
('Negar', 'Sistani', 'Karaj', 'IR', 'MohamadShahr'),
('alireza', 'nafeei', 'khuzestan', 'IR', 'khuzestan'),
('Elham', 'bozorgi', 'ardebil', 'IR', 'Azarbaijan')
```

در دستورات بالا، در خط اول با دستور CREATE DATABASE، یک دیتابیس یا نام B22 ایجاد کردیم و در ادامه، حتماً دستور GO را قرار دهید تا بعد از ایجاد دیتابیس B22، یک refresh صورت بگیرد؛ در ادامه و در خط سوم باید مشخص کنیم که از چه دیتابیسی می‌خواهیم استفاده کنیم تا جدول را داخل آن ایجاد کنیم که این کار را با دستور USE B22 انجام می‌دهیم.

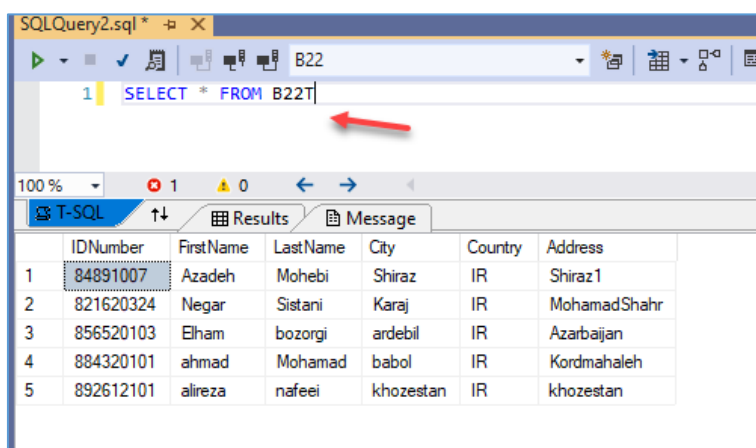
در خط چهارم با استفاده از دستور CREATE TABLE، یک جدول با نام B22T در دیتابیس B22 ایجاد می‌کنیم و اطلاعات آن را نیز در ادامه مشخص می‌کنیم؛ توجه داشته باشید که IDNumber از نوع Primary Key است و نباید خالی باشد.

بعد از ایجاد جدول B22T باید با دستور INSERT INTO، اطلاعات مورد نظر را وارد ستون‌های جدول کنیم؛ به این نکته‌ی مهم توجه کنید که IDNumber در دستور INSERT INTO وجود ندارد؛ این موضوع به این دلیل است که در خط پنجم از دستور identity استفاده کردیم و باعث می‌شود که به‌صورت اتوماتیک در هر سطر، یک شماره به آن سطر اختصاص دهد و اگر بخواهید از IDNumber در داخل جدول استفاده کنید و شماره‌ی دلخواه خود را وارد کنید باید کد بالا را به‌صورت زیر تغییر دهید:

```
CREATE DATABASE B22
GO
USE B22
CREATE TABLE B22T (
    IDNumber int primary key,
    FirstName varchar(50),
    LastName varchar(50),
    City varchar(10),
    Country varchar(10),
    Address varchar(50)
);
INSERT INTO B22T(IDNumber,FirstName, LastName, City, Country, Address) VALUES
('884320101','ahmad','Mohamad','babol','IR','Kordmahaleh'),
('84891007','Azadeh','Mohebi','Shiraz','IR','Shiraz1'),
('821620324','Negar','Sistani','Karaj','IR','MohamadShahr'),
('892612101','alireza','nafeei','khuzestan','IR','khuzestan'),
('856520103','Elham','bozorgi','ardebil','IR','Azarbaijan')
```

در کد بالا و در خط پنجم، دستور identity حذف شده است و حتماً باید در ورودی به IDNumber یک مقدار بدهید تا با خطا مواجه نشوید؛ دلیل آن نیز این است که این گزینه از نوع Primary Key است و نباید خالی باشد؛ به این دلیل در خط دوازدهم و در داخل پرانتز، IDNumber تعریف و مقدار آن نیز داده شده است.

همان‌طور که در شکل ۳-۱۳۸ مشاهده می‌کنید، دستورات به درستی اجرا شده و خروجی جدول، B22T را چاپ کرده است؛ توجه داشته باشید که این خروجی مربوط به کد دومی است که داخل آن از IDNumber استفاده شده است.

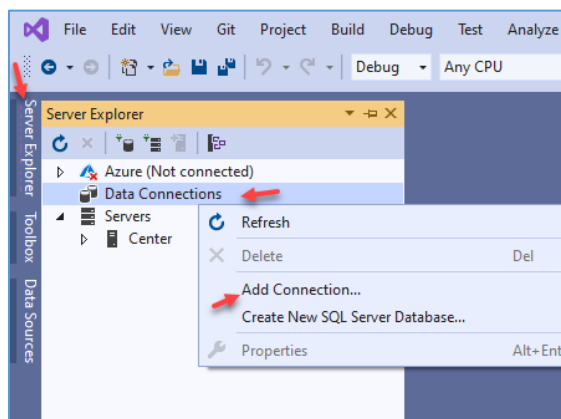


IDNumber	FirstName	LastName	City	Country	Address
84891007	Azadeh	Mohebi	Shiraz	IR	Shiraz1
821620324	Negar	Sistani	Karaj	IR	MohamadShahr
856520103	Elham	bozorgi	ardebil	IR	Azarbaijan
884320101	ahmad	Mohamad	babol	IR	Kordmahaleh
892612101	alireza	nafeei	khuzestan	IR	khuzestan

شکل ۳-۱۳۸ خروجی جدول B22T

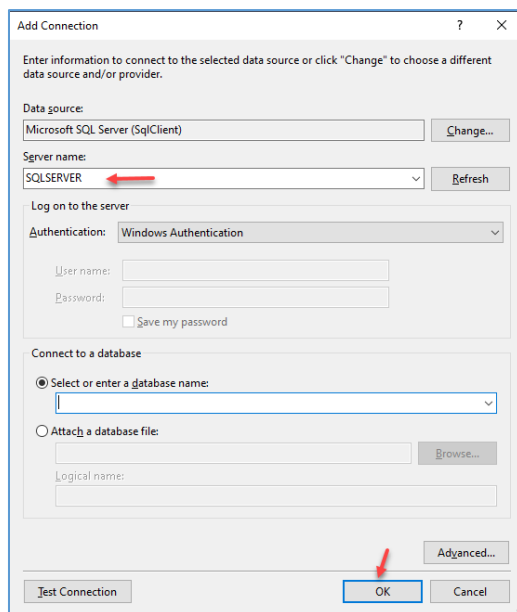
۳-۱-۸-۲ ایجاد دیتابیس از طریق ابزار Visual Studio

در قسمت قبلی توانستیم با استفاده از Query، یک دیتابیس جدید ایجاد و جدول آن را به همراه مقادیر آن ایجاد کنیم؛ در این قسمت نیز می‌خواهیم از طریق ابزارهای موجود در Visual Studio این کار را انجام دهیم. برای شروع به‌مانند شکل ۳-۱۳۹ وارد Server Explorer شوید و بر روی Data Connections کلیک راست کنید و بر روی Add Connection کلیک کنید.



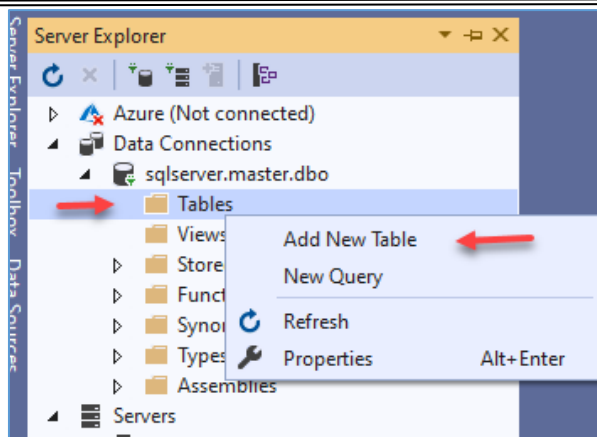
شکل ۳-۱۳۹ / ایجاد دیتابیس

بعد از باز شدن پنجره در شکل ۳-۱۴۰، نام سرور SQL را وارد و بر روی Connect کلیک کنید.



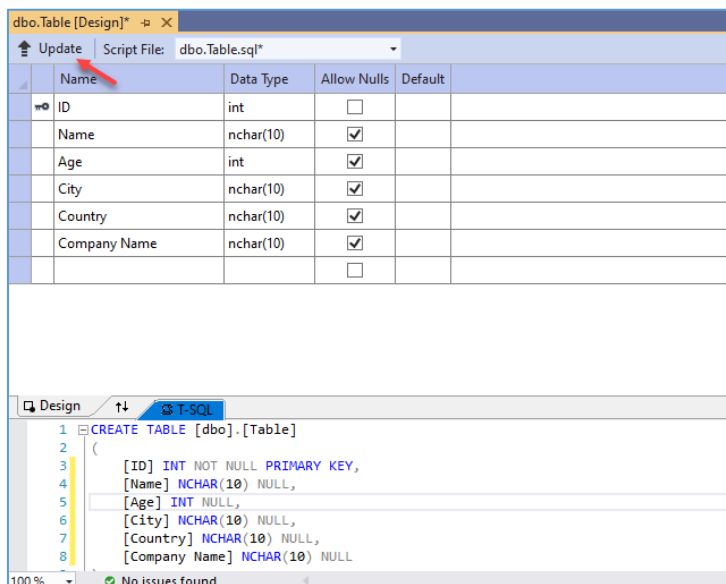
شکل ۳-۱۴۰ / ایجاد دیتابیس

همان‌طور که در شکل ۳-۱۴۱ مشاهده می‌کنید به SQL مورد نظر متصل شدیم و برای اینکه بتوانیم جدول مورد نظر خود را ایجاد کنیم، می‌توانیم بر روی Tables کلیک راست کنیم و گزینه‌ی Add New Table را انتخاب کنیم.



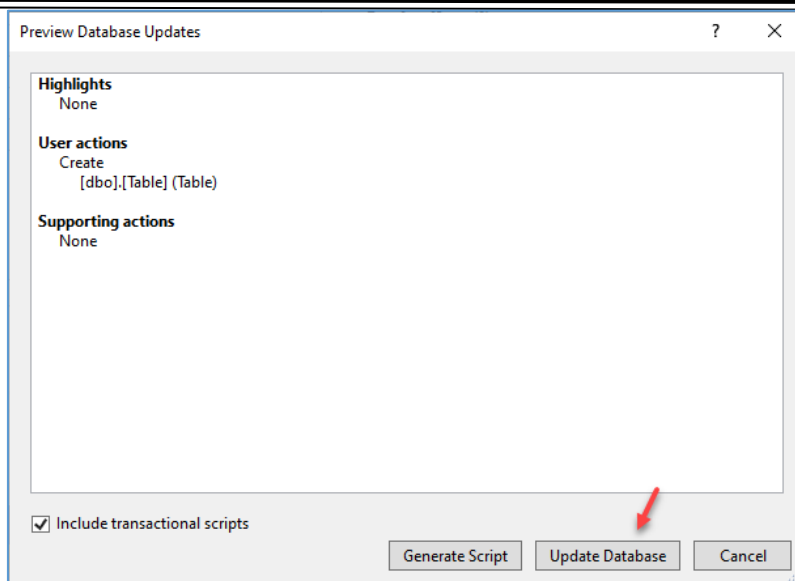
شکل ۳-۱۴۱ / ایجاد جدول

همان طور که در شکل ۳-۱۴۲ مشاهده می‌کنید به راحتی می‌توانید اطلاعات مورد نظر جدول خود را وارد کنید و هم‌زمان که این کار را انجام می‌دهید در زیر شکل ۳-۱۴۲، Query آن نیز نوشته می‌شود؛ برای تایید اطلاعات باید بر روی Update کلیک کنید.



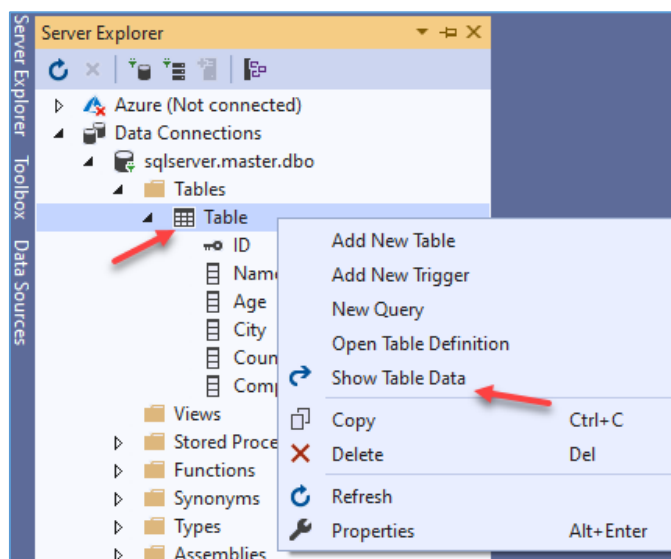
شکل ۳-۱۴۲ / ایجاد جدول

بعد از کلیک بر روی Update، شکل ۳-۱۴۳ ظاهر می‌شود که باید بر روی Update Database کلیک کنید تا اطلاعات به سرور SQL ارسال شود و در آخر پیغام تأیید را برای شما ارسال می‌کند.



شکل ۳-۱۴۳ / ایجاد دیتابیس

همان طور که در شکل ۳-۱۴۴ مشاهده می‌کنید، جدول مورد نظر ایجاد شده است و برای اینکه اطلاعات خود را وارد کنید باید بر روی جدول کلیک راست کنید و گزینه‌ی Show Table Data را انتخاب کنید.



شکل ۳-۱۴۴ / ایجاد جدول و نمایش

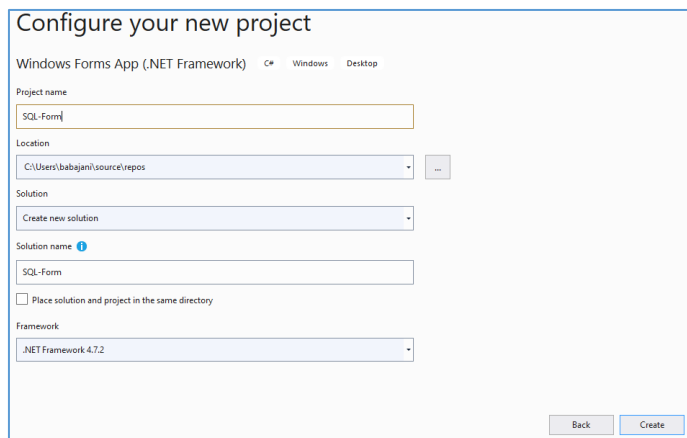
همان طور که در شکل ۳-۱۴۵ مشاهده می‌کنید، اطلاعات را می‌توانید وارد ستون‌های جدول کنید.

ID	Name	Age	City	Country	Company Name
1	reza	12	babol	iran	3isco
NULL	NULL	NULL	NULL	NULL	NULL

شکل ۳-۱۴۵ / نمایش جدول

۳-۱-۸-۳ ایجاد فرم در Visual Studio و ثبت اطلاعات در جدول SQL

در این قسمت می‌خواهیم یک فرم طراحی کنیم تا کاربر بعد از وارد کردن اطلاعات، آن را در SQL ثبت کند؛ برای اینکه یک فرم ایجاد کنیم و اطلاعات ورودی کاربر را در SQL ثبت کنیم باید به‌مانند شکل ۳-۱۴۶ یک پروژه‌ی جدید از نوع Windows Forms App ایجاد کنیم.



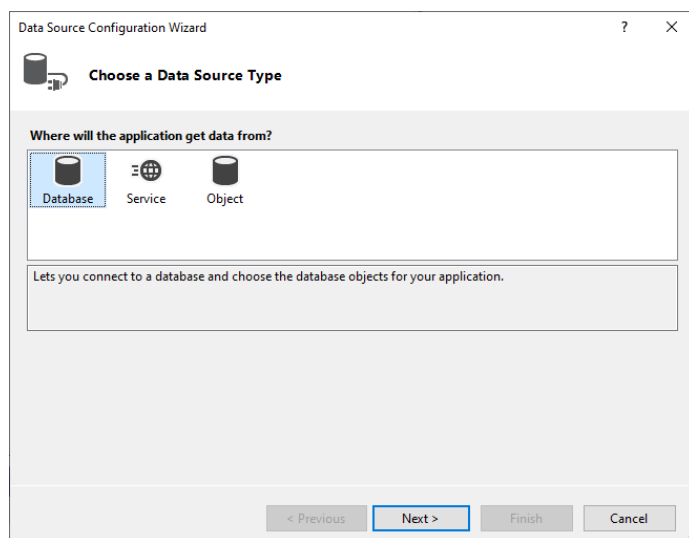
شکل ۳-۱۴۶ ایجاد فایل جدید

در شکل ۳-۱۴۷ وارد Data Sources شوید و بر روی آیکن مورد نظر کلیک کنید.



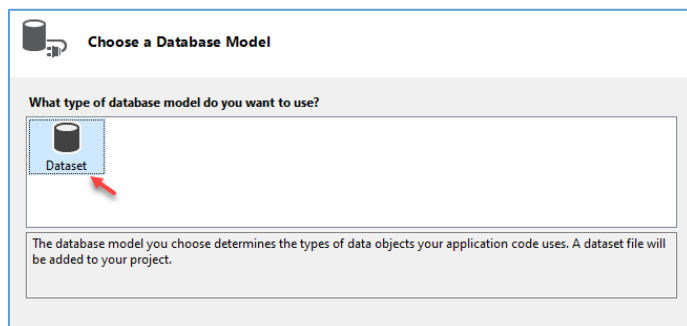
شکل ۳-۱۴۷ ایجاد Data Source

در شکل ۳-۱۴۸ گزینه‌ی Database را انتخاب کنید.



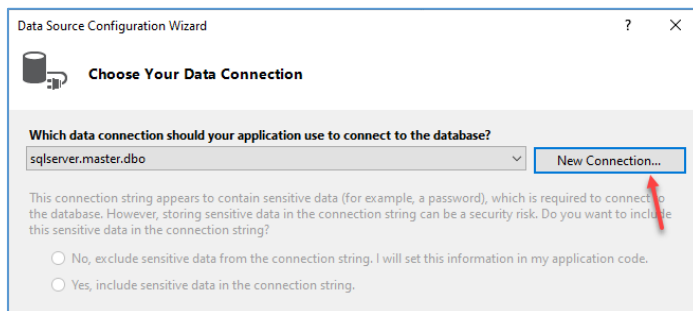
شکل ۳-۱۴۸ ارتباط با SQL

در شکل ۳-۱۴۹ گزینه‌ی Dataset را انتخاب و بر روی Next کلیک کنید.



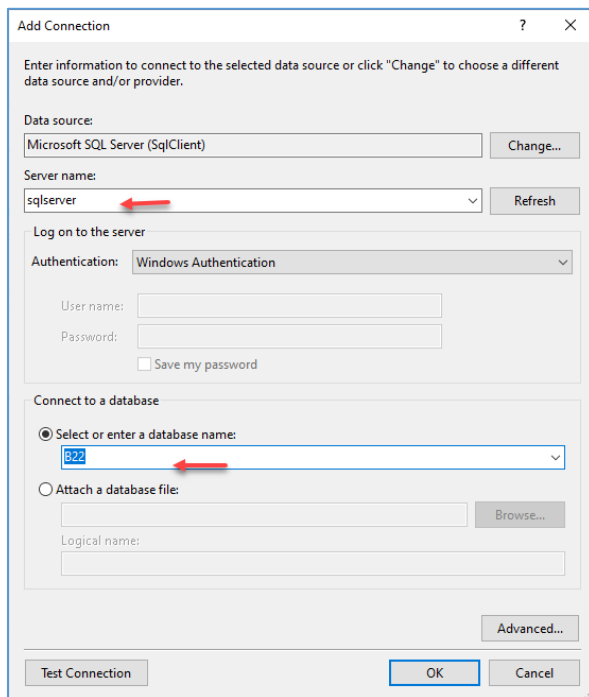
شکل ۳-۱۴۹ ارتباط با SQL

در شکل ۳-۱۵۰ بر روی New Connection کلیک کنید.



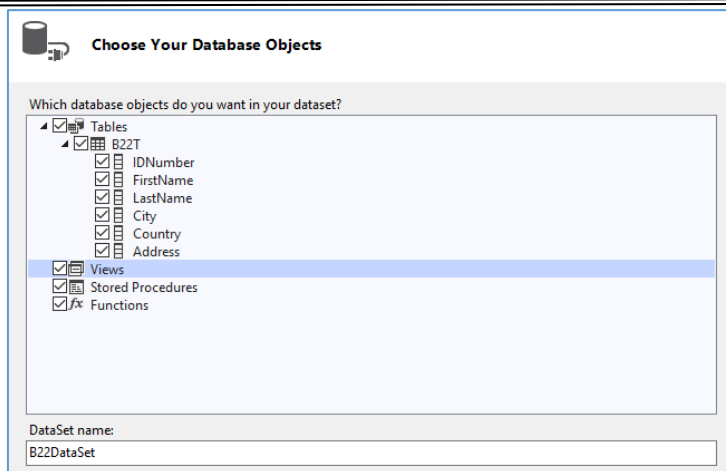
شکل ۳-۱۵۰ ارتباط با SQL

در شکل ۳-۱۵۱ باید نام سرور را وارد و دیتابیس مورد نظر را انتخاب کنید.



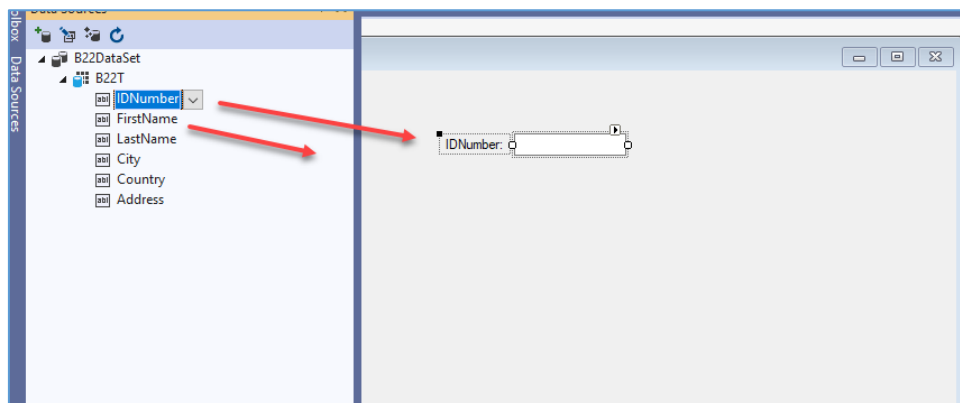
شکل ۳-۱۵۱ ارتباط با SQL

در شکل ۳-۱۵۲ تیک همه‌ی گزینه‌ها را انتخاب و بر روی Next کلیک کنید.



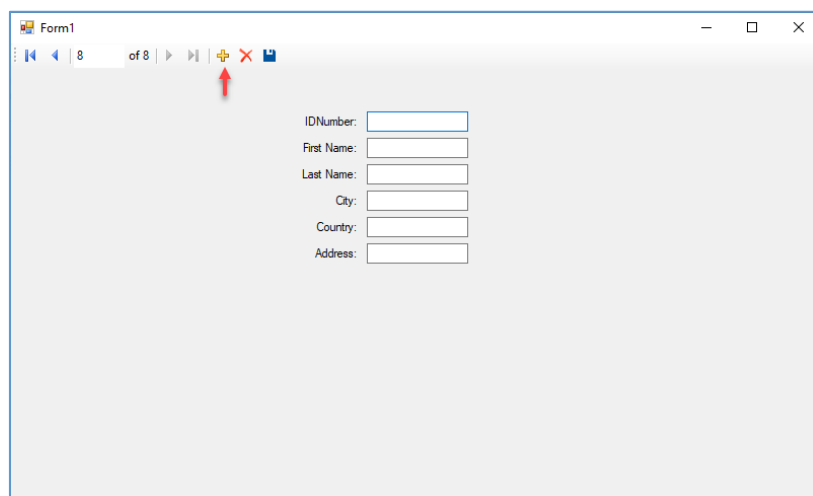
شکل ۱۵۲-۳ / ارتباط با SQL

در شکل ۱۵۳-۳ همه‌ی ستون‌های جدول مورد نظر را بکشید و در فرم رها کنید.



شکل ۱۵۳-۳ / ایجاد فرم

در شکل ۱۵۴-۳ برای وارد کردن اطلاعات در جدول مورد نظر بر روی + کلیک و اطلاعات را وارد و در آخر بر روی آیکون Save کلیک کنید.



شکل ۱۵۴-۳ / ایجاد فرم

۳-۱-۹ وارد کردن فایل اکسل در SQL

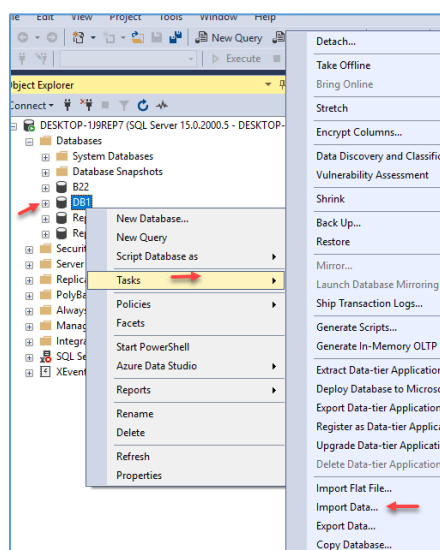
شما حتماً با فایل‌های اکسل کار کردید و کارهای روزمره‌ی خود را در آن انجام می‌دهید، مانند: اطلاعات حسابداری، ورود و خروج به شرکت و ... که این نرم‌افزار را می‌توان یکی از نرم‌افزارهای پرکاربرد در سطح جهان دانست. اگر شما یک فایل اکسلی داشته باشید و بخواهید آن را وارد جداول SQL کنید، باید چه کاری انجام دهید؛ آیا این کار شدنی است؟

بله، این کار توسط SQL شدنی است و به راحتی می‌توانید فایل‌های اکسل را وارد جداول و دیتابیس SQL کنید؛ برای این کار یک فایل اکسل را در شکل ۳-۱۵۵ مشاهده می‌کنید که دارای ۸ ستون و چندین سطر است که می‌خواهیم آن را وارد SQL Server کنیم.

Account	Name	Rep	Manager	Product	Quantity	Price	Status
714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
737550	Fritsch, Russel and Anderson	Craig Booker	Debra Henley	CPU	1	35000	declined
146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won
218895	Kulas Inc	Daniel Hilton	Debra Henley	CPU	2	40000	pending
218895	Kulas Inc	Daniel Hilton	Debra Henley	Software	1	10000	presented
412290	Jerde-Hilpert	John Smith	Debra Henley	Maintenance	2	5000	pending
740150	Barton LLC	John Smith	Debra Henley	CPU	1	35000	declined
141962	Herman LLC	Cedric Moss	Fred Anderson	CPU	2	65000	won
163416	Purdy-Kunde	Cedric Moss	Fred Anderson	CPU	1	30000	presented
239344	Stokes LLC	Cedric Moss	Fred Anderson	Maintenance	1	5000	pending
239344	Stokes LLC	Cedric Moss	Fred Anderson	Software	1	10000	presented
307599	Kassulke, Ondricka and Metz	Wendy Yule	Fred Anderson	Maintenance	3	7000	won
68981	Keeling LLC	Wendy Yule	Fred Anderson	CPU	5	100000	won
729833	Koepf Ltd	Wendy Yule	Fred Anderson	CPU	2	65000	declined
729833	Koepf Ltd	Wendy Yule	Fred Anderson	Monitor	2	5000	presented

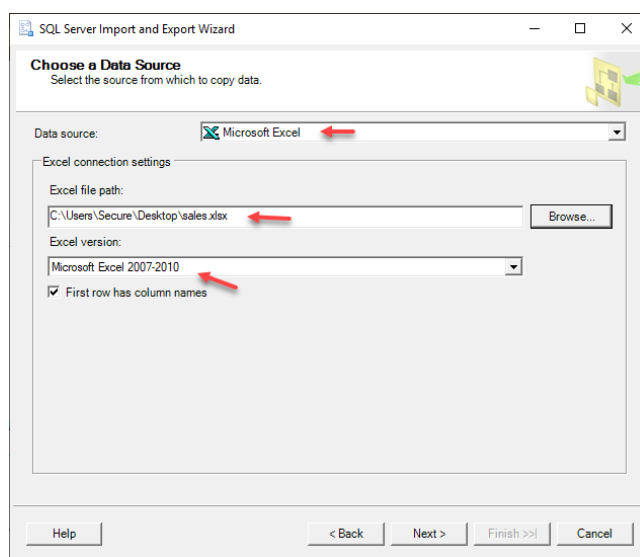
شکل ۳-۱۵۵ Excel

در ادامه‌ی کار به‌مانند شکل ۳-۱۵۶ وارد SQL Server شوید و بر روی دیتابیس مورد نظر خود که قرار است فایل اکسل را وارد آن کنید، کلیک راست کنید و از قسمت Tasks، گزینه‌ی Import Data را انتخاب کنید.



شکل ۳-۱۵۶ وارد کردن فایل اکسل

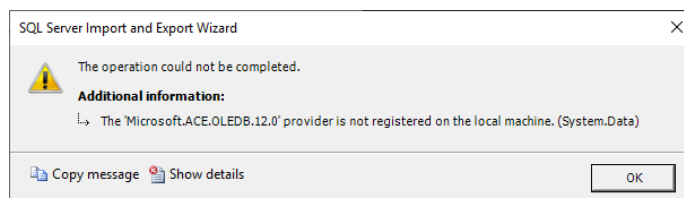
در شکل ۳-۱۵۷ و در قسمت Data source باید Microsoft Excel را انتخاب کنید و بعد بر روی Browse کلیک کنید و فایل مورد نظر خود را انتخاب کنید؛ بعد از انتخاب در قسمت سوم، ورژن آن مشخص خواهد شد؛ برای ادامه‌ی کاربر روی Next کلیک کنید.



شکل ۳-۱۵۷ وارد کردن فایل Excel

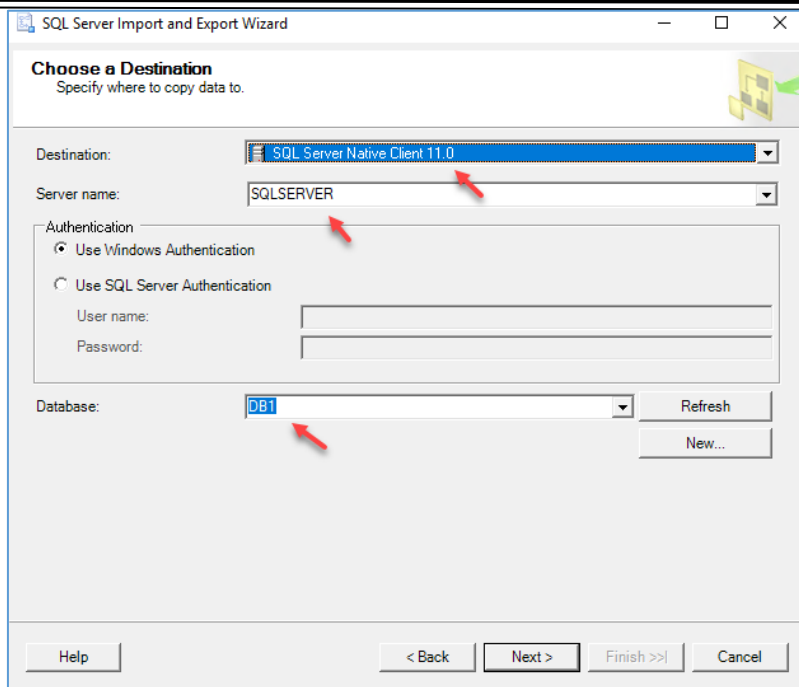
بعد از کلیک بر روی Next در شکل ۳-۱۵۷ با خطای شکل ۳-۱۵۹ روبرو می‌شوید که برای حل آن باید از لینک زیر، نرم‌افزار Microsoft Access Database Engine 2010 Redistributable را دانلود و نصب کنید تا این خطا برطرف شود.

<https://www.microsoft.com/en-us/download/details.aspx?id=13255>



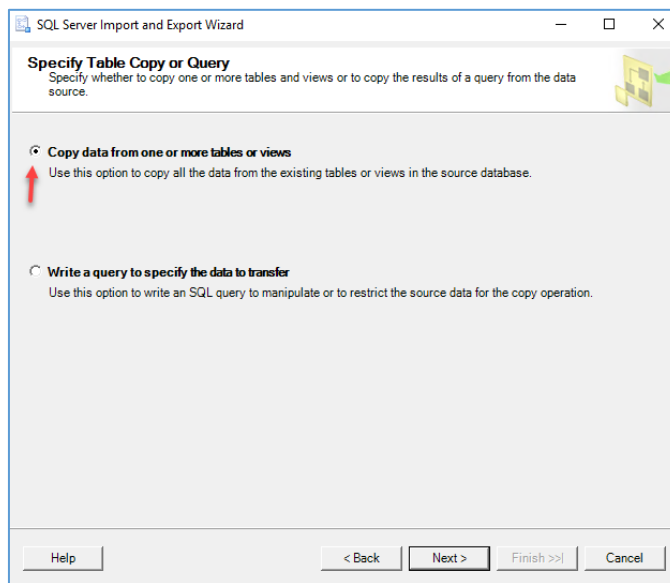
شکل ۳-۱۵۹ خطای فایل Excel

بعد از نصب، دوباره مراحل بالا را اجرا کنید و به‌مانند شکل ۳-۱۶۰ در قسمت Destination گزینه‌ی SQL Server Native Client 11.0 را انتخاب کنید؛ در قسمت Server name، نام سرور SQL خود را وارد و در قسمت Database دیتابسی که قرار است اطلاعات اکسل در آن وارد شود را انتخاب و بر روی Next کلیک کنید.



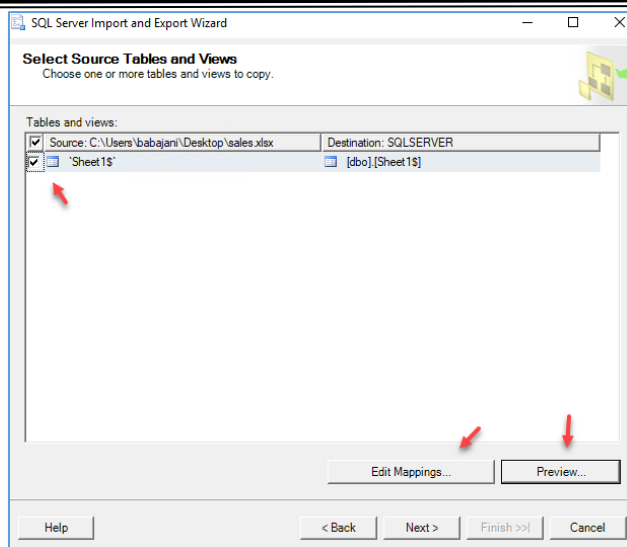
شکل ۳-۱۶۰ وارد کردن فایل Excel

در شکل ۳-۱۶۱ باید گزینه‌ی اول را انتخاب و بر روی Next کلیک کنید.



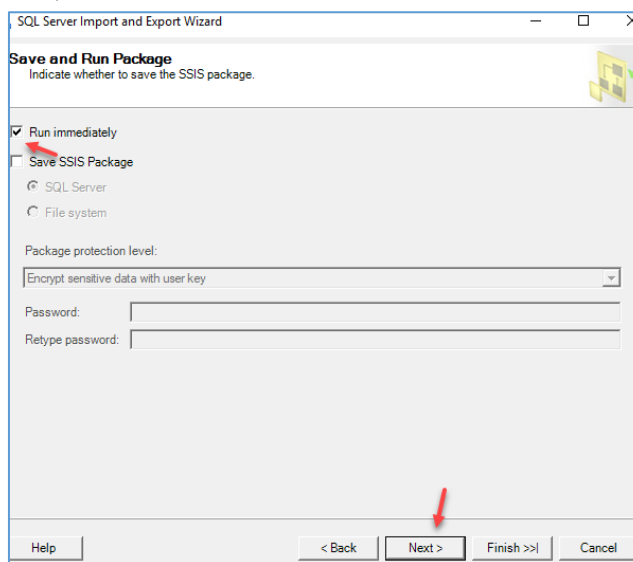
شکل ۳-۱۶۱ وارد کردن فایل Excel

همان طور که در شکل ۳-۱۶۲ مشاهده می‌کنید، Sheet مورد نظر در اکسل پیدا شده است و شما می‌توانید با کلیک بر روی دکمه‌ی Edit Mappings، ستون‌های آن را ویرایش کنید و همچنین برای نمایش اطلاعات بر روی دکمه‌ی Preview کلیک کنید.



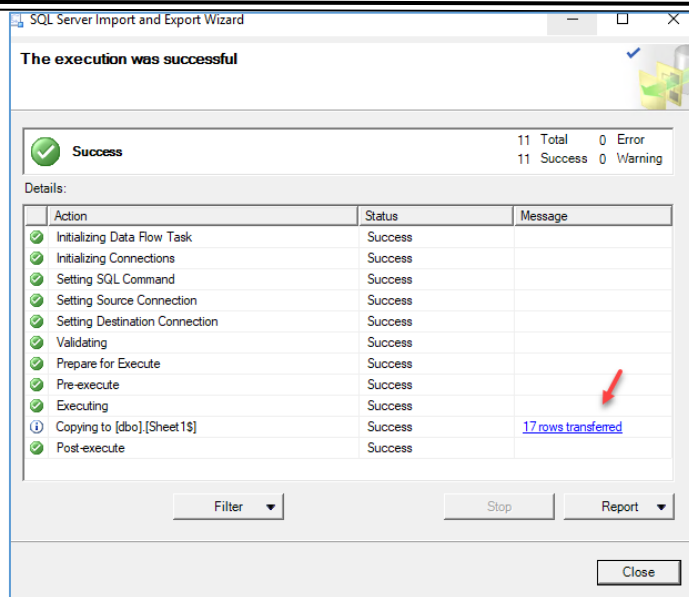
شکل ۳-۱۶۲ وارد کردن فایل Excel

در شکل ۳-۱۶۳ گزینه‌ی اول را انتخاب کنید و اگر می‌خواهید بر روی اطلاعات، رمز عبور قرار دهید باید گزینه‌ی دوم را انتخاب کنید و در صفحه‌ی آخر بر روی Finish کلیک کنید تا کار انتقال انجام شود.



شکل ۳-۱۶۳ وارد کردن فایل Excel

همان طور که در شکل ۳-۱۶۴ مشاهده می‌کنید، عملیات انتقال با موفقیت انجام شده است.



شکل ۱۶۴-۳ وارد کردن فایل Excel

همان طور که در شکل ۱۶۵-۳ مشاهده می‌کنید، جدول مورد نظر در دیتابیس DB1 ایجاد شده است و اگر از آن خروجی بگیرید، تمام اطلاعات آن را مشاهده خواهید کرد.

Account	Name	Rep	Manager	Product	Quantity	Price	Status
714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
737550	Fritsch, Russel a...	Craig Booker	Debra Henley	CPU	1	35000	declined
146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won
218895	Kulas Inc	Daniel Hilton	Debra Henley	CPU	2	40000	pending
218895	Kulas Inc	Daniel Hilton	Debra Henley	Software	1	10000	presented
412290	Jerde-Hilpert	John Smith	Debra Henley	Maintenance	2	5000	pending
740150	Barton LLC	John Smith	Debra Henley	CPU	1	35000	declined
141962	Herman LLC	Cedric Moss	Fred Anderson	CPU	2	65000	won
163416	Purdy-Kunde	Cedric Moss	Fred Anderson	CPU	1	30000	presented
239344	Stokes LLC	Cedric Moss	Fred Anderson	Maintenance	1	5000	pending
239344	Stokes LLC	Cedric Moss	Fred Anderson	Software	1	10000	presented
307599	Kassulke, Ondri...	Wendy Yule	Fred Anderson	Maintenance	3	7000	won
688981	Keeling LLC	Wendy Yule	Fred Anderson	CPU	5	100000	won
729833	Koeppe Ltd	Wendy Yule	Fred Anderson	CPU	2	65000	declined
729833	Koeppe Ltd	Wendy Yule	Fred Anderson	Monitor	2	5000	presented
714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
737550	Fritsch, Russel a...	Craig Booker	Debra Henley	CPU	1	35000	declined
146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won
218895	Kulas Inc	Daniel Hilton	Debra Henley	CPU	2	40000	pending
218895	Kulas Inc	Daniel Hilton	Debra Henley	Software	1	10000	presented
412290	Jerde-Hilpert	John Smith	Debra Henley	Maintenance	2	5000	pending
740150	Barton LLC	John Smith	Debra Henley	CPU	1	35000	declined
141962	Herman LLC	Cedric Moss	Fred Anderson	CPU	2	65000	won
163416	Purdy-Kunde	Cedric Moss	Fred Anderson	CPU	1	30000	presented
239344	Stokes LLC	Cedric Moss	Fred Anderson	Maintenance	1	5000	pending
239344	Stokes LLC	Cedric Moss	Fred Anderson	Software	1	10000	presented

شکل ۱۶۵-۳ اطلاعات فایل Excel در SQL

۳-۱-۱۰ بررسی دستور Stored Procedures

زمانی که شما یک کد T-SQL برای یک قسمت از بانک اطلاعاتی خود می‌نویسید، شاید دوست داشته باشید همیشه از آن کد استفاده کنید و به این دلیل همیشه باید آن را چندین و چند بار بنویسید یا کپی کنید؛ روشی وجود دارد با عنوان Stored Procedures که دستورات پرکاربرد T-SQL شما را در نام ذخیره می‌کند و دیگر نیاز نیست، کل کد را بنویسید و تنها کافی است آن نام مورد نظر را صدا بزنید.

شکل کلی دستور به صورت زیر است:

```
CREATE PROCEDURE procedure_name
AS
sql_statement
GO;
```

برای اینکه PROCEDURE که ایجاد کردید را فراخوانی کنید باید از دستور زیر استفاده کنید:

```
EXEC procedure_name;
```

برای اینکه این دستور را تست بگیریم، یک مثال را با هم بررسی می‌کنیم؛ برای شروع دستورات زیر را اجرا کنید:

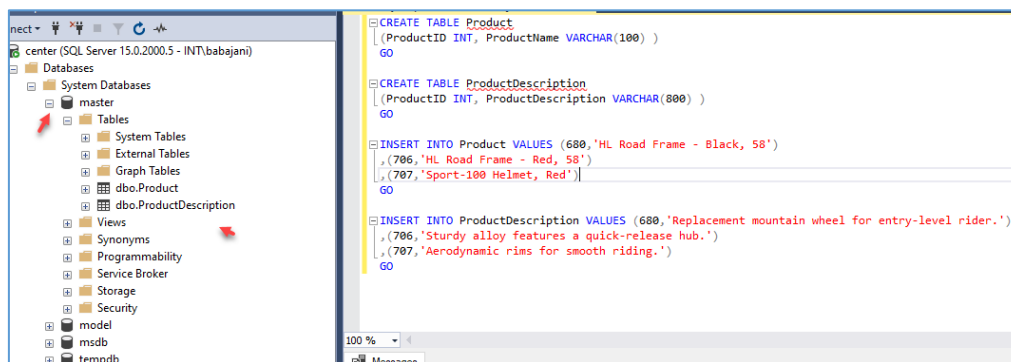
```
CREATE TABLE Product
(ProductID INT, ProductName VARCHAR(100) )
GO
```

```
CREATE TABLE ProductDescription
(ProductID INT, ProductDescription VARCHAR(800) )
GO
```

```
INSERT INTO Product VALUES (680, 'HL Road Frame - Black, 58')
, (706, 'HL Road Frame - Red, 58')
, (707, 'Sport-100 Helmet, Red')
GO
```

```
INSERT INTO ProductDescription VALUES (680, 'Replacement mountain wheel for entry-level
rider.')
, (706, 'Sturdy alloy features a quick-release hub.')
, (707, 'Aerodynamic rims for smooth riding.')
GO
```

با اجرای دستورات بالا، دو جدول Product و ProductDescription در دیتابیس master ایجاد می‌شود و داده‌های داخل جدول نیز مشخص شده است که در شکل ۱۶۶-۳ این موضوع نشان داده شده است.



شکل ۱۶۶-۳ / ایجاد جدول

بعد از ایجاد جدول، حال می‌خواهیم یک Procedure ایجاد کنیم و اطلاعات را در آن قرار دهیم؛ برای این موضوع از دستور زیر استفاده کنید:

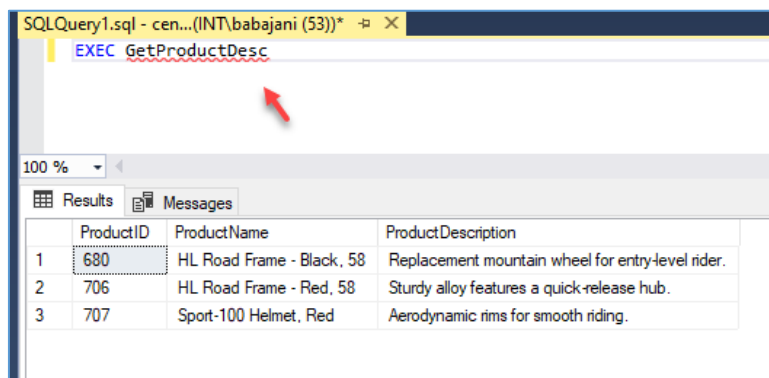
```
CREATE PROCEDURE GetProductDesc
AS
BEGIN
SET NOCOUNT ON

SELECT P.ProductID, P.ProductName, PD.ProductDescription FROM
Product P
INNER JOIN ProductDescription PD ON P.ProductID=PD.ProductID
```

END

در دستور بالا، یک Procedure با نام GetProductDesc ایجاد می‌کنیم و در ادامه‌ی دستورات با استفاده از INNER JOIN، دو جدول را به هم متصل می‌کنیم که اطلاعات خروجی در Procedure مورد نظر ذخیره می‌شود. بعد از ایجاد Procedure باید آن را با دستور زیر فراخوانی کنیم:

EXEC GetProductDesc



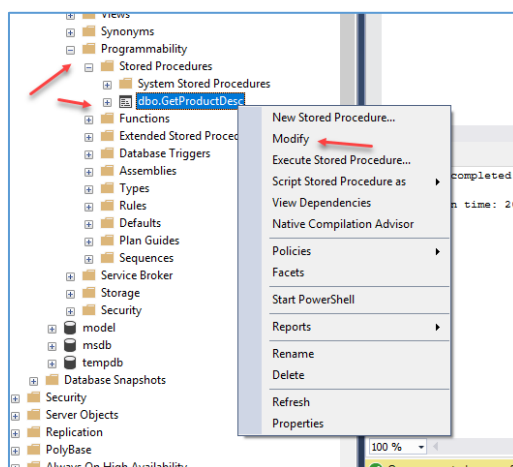
شکل ۱۶۷-۳ اجرای Procedure

نکته:

به این نکته باید توجه کنید که حتی با بستن پنجره Query و بستن SQL Management Studio باز هم Procedure مورد نظر قابل استفاده است.

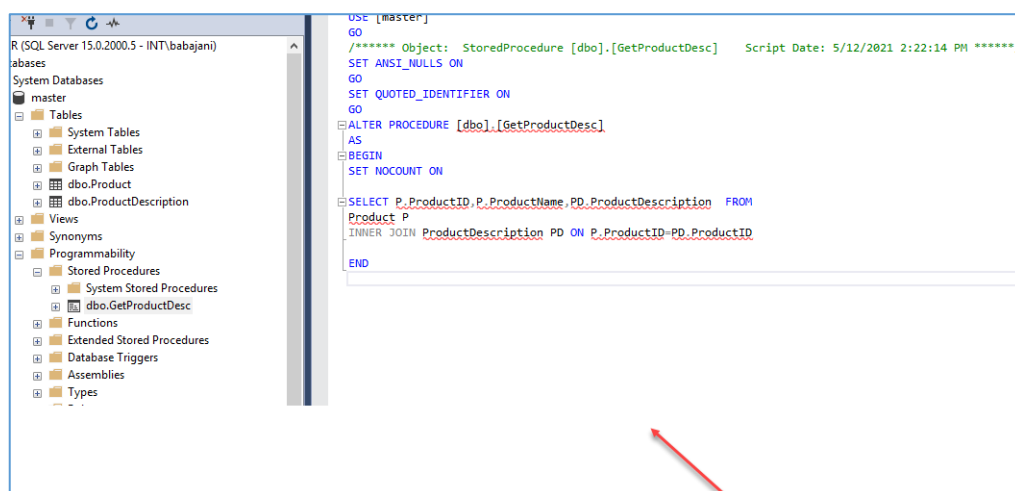
۲-۱۰-۱-۳ تغییر یک Stored procedure

برای تغییر یا اصلاح یک stored procedure موجود از دستور ALTER PROCEDURE استفاده می‌کنیم. ابتدا فولدر stored procedure را باز کنید تا محتوای آن را ببینید، سپس بر روی نام stored procedure مورد نظر کلیک راست کنید و از آیتم‌های داخل منو، گزینه‌ی Modify را انتخاب کنید:



شکل ۱۷۰-۳ ویرایش Procedure

همان طور که در شکل ۳-۱۷۱ مشاهده می‌کنید، می‌توانید کد مورد نظر را تغییر و اطلاعات را ذخیره و اجرا کنید.



شکل ۳-۱۷۱ ویرایش Procedure

۳-۱-۱۰-۳ حذف یک Stored procedure

برای حذف یک stored procedure می‌توانید از دستور DROP PROCEDURE یا DROP PROC استفاده کنید:

```
DROP PROCEDURE sp_name;
```

یا

```
DROP PROC sp_name;
```

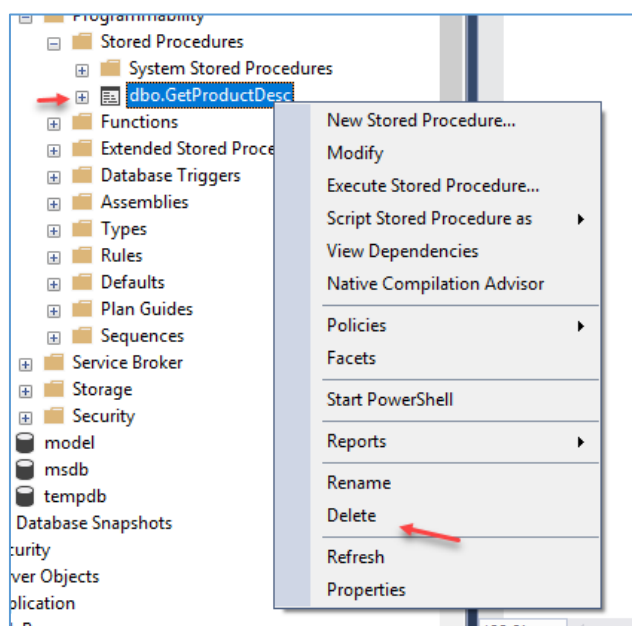
که `sp_name` نام stored procedure مورد نظر شماست که می‌خواهید حذف شود.

برای مثال، برای حذف stored procedure به نام `uspProductList` باید دستور زیر را اجرا کنیم:

```
DROP PROCEDURE uspProductList;
```

اگر بخواهید به صورت گرافیکی این کار را انجام دهید باید بر روی Procedure مورد نظر کلیک راست کنید و گزینه‌ی

DELETE را انتخاب کنید که در شکل ۴۰ مشخص شده است.



شکل ۳-۱۷۲ حذف Procedure

در نتیجه، چگونگی مدیریت stored procedure ها در SQL Server به خصوص اجرا، تغییر و حذف stored procedure را آموختید.

۱-۱۰-۳ استفاده از پارامتر در دستور Procedure

در قسمت قبل، چگونگی ایجاد یک stored procedure ساده که یک دستور SELECT را پوشش می‌داد را آموختید؛ وقتی stored procedure را فراخوانی می‌کنید، خیلی ساده query داخل آن اجرا می‌شود و یک مجموعه نتیجه را بازمی‌گرداند.

در این قسمت، بحث stored procedure را گسترش می‌دهیم به صورتی که خواهیم توانست یک یا چند مقدار را به آن انتقال دهیم؛ نتیجه‌ی stored procedure بر اساس مقادیر پارامترها تغییر خواهد کرد.

ایجاد یک stored procedure با یک پارامتر

Query زیر یک لیست محصول از جدول products در پایگاه داده‌ی Bikestores بازمی‌گرداند:

```
SELECT
    product_name,
    list_price
FROM
    production.products
ORDER BY
    list_price;
```

می‌توانید یک stored procedure ایجاد کنید که با استفاده از دستور CREATE PROCEDURE این query را احاطه کند:

```
CREATE PROCEDURE uspFindProducts
AS
BEGIN
    SELECT
        product_name,
        list_price
    FROM
        production.products
    ORDER BY
        list_price;
END;
```

در هر صورت، این دفعه می‌توانیم یک پارامتر به stored procedure اضافه کنیم تا محصولاتی که لیست قیمت‌های آنها بیشتر از یک قیمت ورودی هستند را بیابد:

```
ALTER PROCEDURE uspFindProducts(@min_list_price AS DECIMAL)
AS
BEGIN
    SELECT
        product_name,
        list_price
    FROM
        production.products
    WHERE
```



```
list_price >= @min_list_price
ORDER BY
list_price;
END;
```

در این مثال:

ابتدا، یک پارامتر به نام `@min_list_price` به stored procedure به نام `uspFindProducts` اضافه کردیم؛ هر پارامتر باید با علامت `@` آغاز شود. کلیدواژه‌های `AS DECIMAL`، نوع داده‌ی پارامتر `@min_list_price` را مشخص می‌کند؛ پارامترها باید با آکولادهای باز و بسته احاطه شوند.

سپس از پارامتر `@min_list_price` در دستور `WHERE`، درون دستور `SELECT` برای فیلتر محصولاتی که لیست قیمت‌های آنها بیشتر یا برابر با `@min_list_price` هستند، استفاده کردیم.

اجرای یک stored procedure با یک پارامتر

برای اجرای stored procedure باید به‌مانند شکل زیر یک آرگومان به `uspFindProducts` ارسال کنید:

```
EXEC uspFindProducts 100 ;
```

product_name	list_price
Sun Bicycles Lil Kitt'n - 2017	109.99
Trek Boy's Kickster - 2015/2017	149.99
Trek Girl's Kickster - 2017	149.99
Trek Kickster - 2018	159.99
Trek Precaliber 12 Boys - 2017	189.99
Trek Precaliber 12 Girls - 2017	189.99
Trek Precaliber 12 Boy's - 2018	199.99
Trek Precaliber 12 Girl's - 2018	199.99
Trek Precaliber 16 Boy's - 2018	209.99
Trek Precaliber 16 Girl's - 2018	209.99
Trek Precaliber 16 Boys - 2017	209.99
Trek Precaliber 16 Girls - 2017	209.99
Haro Shredder 20 - 2017	209.99
Haro Shredder 20 Girls - 2017	209.99

stored procedure، تمام محصولاتی که لیست قیمت‌های آنها بیشتر یا برابر با ۱۰۰ هستند را بازمی‌گرداند.

اگر آرگومان را به ۲۰۰ تغییر دهید، مجموعه‌نتایج متفاوتی دریافت خواهید کرد:

product_name	list_price
Haro Shredder 20 - 2017	209.99
Haro Shredder 20 Girls - 2017	209.99
Trek Precaliber 16 Boys - 2017	209.99
Trek Precaliber 16 Girls - 2017	209.99
Trek Precaliber 16 Boy's - 2018	209.99
Trek Precaliber 16 Girl's - 2018	209.99
Trek Precaliber 20 Boy's - 2018	229.99
Trek Precaliber 20 Girl's - 2018	229.99
Trek MT 201 - 2018	249.99
Strider Sport 16 - 2018	249.99
Haro Shredder Pro 20 - 2017	249.99
Sun Bicycles Revolutions 24 - 2017	250.99
Sun Bicycles Revolutions 24 - Girl's - 2017	250.99
Electra Cruiser 1 (24-Inch) - 2016	269.99

ایجاد یک stored procedure با چندین پارامتر

stored procedure می‌توانند یک یا چند پارامتر بگیرند؛ پارامترها با ویرگول از هم جدا می‌شوند.

کد زیر stored procedure به نام uspFindProducts را با اضافه کردن یک پارامتر دیگر به نام @max_list_price به آن، تغییر می‌دهد:

```
ALTER PROCEDURE uspFindProducts(
    @min_list_price AS DECIMAL
    ,@max_list_price AS DECIMAL
)
AS
BEGIN
    SELECT
        product_name,
        list_price
    FROM
        production.products
    WHERE
        list_price >= @min_list_price AND
        list_price <= @max_list_price
    ORDER BY
        list_price;
END;
```

وقتی stored procedure با موفقیت تغییر کرد، می‌توانید آن را با ارسال دو آرگومان (یکی برای @min_list_price و دیگری برای @max_list_price) اجرا کنید:

```
EXECUTE uspFindProducts 900, 1000;
```

تصویر زیر خروجی را نشان می‌دهد:

product_name	list_price
Electra Straight 8 3i - 2018	909.99
Trek X-Caliber 7 - 2018	919.99
Trek Stache Carbon Frameset - 2018	919.99
Trek Domane AL 3 - 2018	919.99
Trek Domane AL 3 Women's - 2018	919.99
Trek CrossRip 1 - 2018	959.99
Electra Delivery 3i - 2016/2017/2018	959.99
Surly Wednesday Frameset - 2016	999.99
Surly Big Dummy Frameset - 2017	999.99
Trek X-Caliber 8 - 2017	999.99
Surly Ice Cream Truck Frameset - 2017	999.99
Trek X-Caliber 8 - 2018	999.99
Trek Farley Carbon Frameset - 2018	999.99

استفاده از پارامترهای دارای نام

در صورتی که stored procedureها چندین پارامتر داشته باشند، بهتر و به صرفه تر است که stored procedureها را با استفاده از پارامترهای دارای نام اجرا کنید.

برای مثال، کد زیر stored procedure به نام uspFindProducts را با استفاده از پارامترهای دارای نام (یعنی @min_list_price و @max_list_price) اجرا می کند:

```
EXECUTE uspFindProducts
    @min_list_price = 900,
    @max_list_price = 1000;
```

نتیجه‌ی stored procedure، همان نتیجه‌ی قبلی است، با این تفاوت که حالا کد ما خیلی واضح تر و مرتب تر است.

ایجاد پارامترهای متنی

کد زیر پارامتر @name را به عنوان یک پارامتر رشته حروف به stored procedure اضافه می کند.

```
ALTER PROCEDURE uspFindProducts(
    @min_list_price AS DECIMAL
    , @max_list_price AS DECIMAL
    , @name AS VARCHAR(max)
)
AS
BEGIN
    SELECT
        product_name,
        list_price
    FROM
        production.products
    WHERE
        list_price >= @min_list_price AND
        list_price <= @max_list_price AND
        product_name LIKE '%' + @name + '%'
```

```
ORDER BY
    list_price;
END;
```

در دستور WHERE از کد SELECT، شرط زیر را اضافه می‌کنیم:

```
product_name LIKE '%' + @name + '%'
```

با این کار، stored procedure محصولاتی که لیست قیمت‌های آن‌ها در بازه‌ی حداقل و حداکثر لیست قیمت‌ها هستند را بازمی‌گرداند و نام محصولات نیز شامل یک قطعه‌متن هستند که ارسال می‌کنید.

وقتی stored procedure با موفقیت تغییر کرد، می‌توانید آن را به‌مانند شکل زیر اجرا کنید:

```
EXECUTE uspFindProducts
    @min_list_price = 900,
    @max_list_price = 1000,
    @name = 'Trek';
```

در این کد از stored procedure به نام uspFindProducts برای یافتن محصولاتی که لیست قیمت آنها در بازه‌ی ۹۰۰ و ۱۰۰۰ و نام آن‌ها شامل واژه Trek باشد استفاده کردیم.

تصویر زیر خروجی را نشان می‌دهد:

product_name	list_price
Trek X-Caliber 7 - 2018	919.99
Trek Stache Carbon Frameset - 2018	919.99
Trek Domane AL 3 - 2018	919.99
Trek Domane AL 3 Women's - 2018	919.99
Trek CrossRip 1 - 2018	959.99
Trek X-Caliber 8 - 2017	999.99
Trek X-Caliber 8 - 2018	999.99
Trek Farley Carbon Frameset - 2018	999.99

ایجاد پارامترهای اختیاری

وقتی uspFinProducts را اجرا می‌کنید، باید تمام سه آرگومان متناظر با سه پارامتر آن را نیز ارسال کنید.

SQL Server به شما اجازه می‌دهد مقادیر پیش‌فرض برای پارامترها مشخص کنید تا وقتی stored procedure را فراخوانی می‌کنید، می‌توانید پارامترها را با مقادیر پیش‌فرض ارسال کنید.

stored procedure زیر را ببینید:

```
ALTER PROCEDURE uspFindProducts(
    @min_list_price AS DECIMAL = 0
    , @max_list_price AS DECIMAL = 999999
    , @name AS VARCHAR(max)
)
AS
BEGIN
    SELECT
        product_name,
```

```

list_price
FROM
production.products
WHERE
list_price >= @min_list_price AND
list_price <= @max_list_price AND
product_name LIKE '%' + @name + '%'
ORDER BY
list_price;
END;

```

در این stored procedure، ما ۰ را به عنوان مقدار پیش فرض برای پارامتر @min_list_price و ۹۹۹.۹۹۹ را به عنوان مقدار پیش فرض برای پارامتر @max_list_price قرار داده ایم.

وقتی stored procedure کامپایل می شود، می توانید آن را بدون ارسال آرگومان به پارامترهای @min_list_price و @max_list_price اجرا کنید:

```

EXECUTE uspFindProducts
@name = 'Trek';

```

product_name	list_price
Trek Boy's Kickster - 2015/2017	149.99
Trek Girl's Kickster - 2017	149.99
Trek Kickster - 2018	159.99
Trek Precaliber 12 Boys - 2017	189.99
Trek Precaliber 12 Girls - 2017	189.99
Trek Precaliber 12 Boy's - 2018	199.99
Trek Precaliber 12 Girl's - 2018	199.99
Trek Precaliber 16 Boy's - 2018	209.99
Trek Precaliber 16 Girl's - 2018	209.99
Trek Precaliber 16 Boys - 2017	209.99
Trek Precaliber 16 Girls - 2017	209.99
Trek Precaliber 20 Boy's - 2018	229.99

در این صورت، stored procedure در زمان اجرای query از ۰ برای پارامتر @min_list_price و از ۹۹۹.۹۹۹ برای پارامتر @max_list_price استفاده می کند.

پارامترهای @min_list_price و @max_list_price پارامترهای اختیاری فراخوانی کرده اند.

مسلماً، همچنین می توانید آرگومان ها را به پارامترهای اختیاری نیز ارسال کنیم. برای مثال، کد زیر تمام محصولات که لیست قیمت های آن ها بیشتر یا برابر با ۶۰۰۰ و نام آن ها شامل واژه Trek است را باز می گرداند:

```

EXECUTE uspFindProducts
@min_list_price = 6000,
@name = 'Trek';

```

product_name	list_price
Trek Silque SLR 8 Women's - 2017	6499.99
Trek Domane SL Frameset - 2018	6499.99
Trek Domane SL Frameset Women's - 2018	6499.99
Trek Emonda SLR 8 - 2018	6499.99
Trek Domane SLR 8 Disc - 2018	7499.99
Trek Domane SLR 9 Disc - 2018	11999.99

استفاده از NULL به عنوان مقدار پیش فرض

در aspFindProducts، ما از ۹۹۹.۹۹۹ به عنوان قیمت حداکثر پیش فرض استفاده کردیم. این کار جالبی نیست، چون در آینده ممکن است محصولات با قیمت‌های بیشتر از این مقدار داشته باشید.

یک تکنیک مرسوم برای پرهیز از این امر، استفاده از NULL به عنوان مقدار پیش فرض برای پارامترها است:

```
ALTER PROCEDURE uspFindProducts(
    @min_list_price AS DECIMAL = 0
    , @max_list_price AS DECIMAL = NULL
    , @name AS VARCHAR(max)
)
AS
BEGIN
    SELECT
        product_name,
        list_price
    FROM
        production.products
    WHERE
        list_price >= @min_list_price AND
        (@max_list_price IS NULL OR list_price <= @max_list_price) AND
        product_name LIKE '%' + @name + '%'
    ORDER BY
        list_price;
END;
```

در دستور WHERE، ما شرط را تغییر دادیم تا بتوانیم از مقدار NULL برای پارامتر @max_list_price استفاده کنیم:

```
(@max_list_price IS NULL OR list_price <= @max_list_price)
```

کد زیر uspFindProducts را برای یافتن محصولاتی که قیمت آن‌ها بزرگ‌تر یا برابر با ۵۰۰ و نام آن‌ها شامل واژه Haro هستند، اجرا می‌کند.

```
EXECUTE uspFindProducts
    @min_list_price = 500,
    @name = 'Haro';
```

product_name	list_price
Haro SR 1.1 - 2017	539.99
Haro Flightline Two 26 Plus - 2017	549.99
Haro SR 1.2 - 2017	869.99
Haro SR 1.3 - 2017	1409.99
Haro Shift R3 - 2017	1469.99

در این بخش، چگونگی ایجاد و اجرای stored procedure ها با یک یا چند پارامتر را آموختید. همچنین چگونگی ایجاد پارامترهای اختیاری و استفاده از NULL به عنوان مقادیر پیش فرض برای پارامترها را نیز آموختید.

۳-۱-۱۱ بررسی Trigger در SQL Server

همان طور که می دانید SQL معدن Log است و هر کاری که در دیتابیس و دیگر جاهای SQL انجام دهید، یک Log از آن تولید می شود که شما می توانید با قابلیت Trigger این Log ها را بهتر مدیریت کنید و آن چیزی را که بخواهید در خروجی نمایش دهید.

در کل Trigger شامل سه نوع مختلف است:

۱- DML

که این نوع Trigger زمانی اجرا می شود که یکی از سه عملیات (Insert, delete, Update) در جدول مورد نظر شما انجام شود.

۲- DDL

در این نوع Trigger زمانی اجرا می شود که سه عملیات (Create, alter, drop) اجرا شود.

۳- Logon

این نوع Trigger زمانی فعال خواهد شد که کاربر وارد SQL و خارج شود.

برای اینکه بهتر با موضوع آشنا شویم یک مثال را با هم بررسی می کنیم و نحوه کارکرد Trigger را بررسی می کنیم. شکل کلی دستور DML Triggers به صورت زیر می باشد:

```
CREATE TRIGGER [schema_name.]trigger_name
ON table_name
{FOR | AFTER | INSTEAD OF} {[INSERT] [,] [UPDATE] [,] [DELETE]}
AS
{sql_statements}
```

برای اینکه دستور بالا را بررسی کنیم یک مثال را با هم انجام می دهیم، در دستور زیر یک دیتابیس با نام Showroom ایجاد می کنیم که این دیتابیس دارای دو جدول CAR و CARLOG است و اطلاعات داخل آنها هم مشخص شده است.

```
CREATE DATABASE Showroom
```

```
GO
```

```
Use Showroom
```

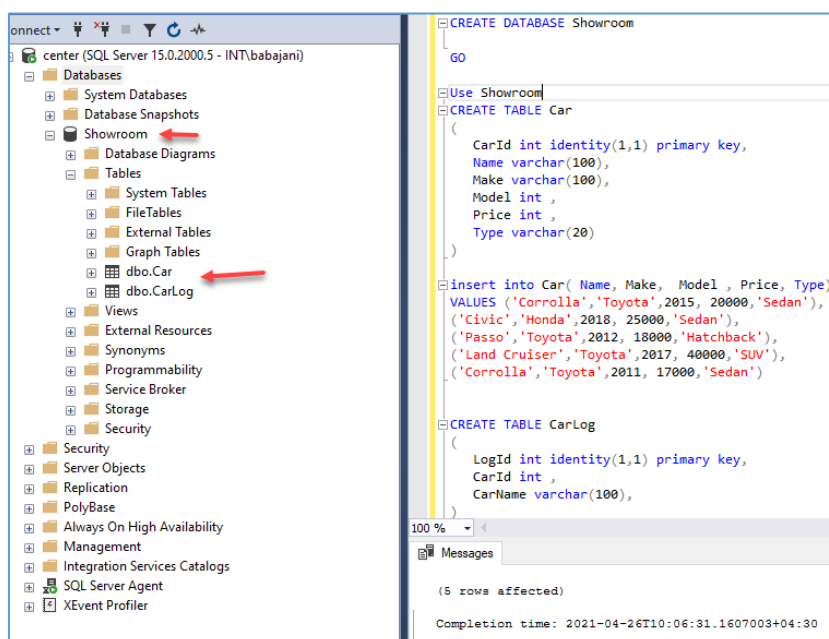
```
CREATE TABLE Car
```

```
(
  CarId int identity(1,1) primary key,
  Name varchar(100),
  Make varchar(100),
  Model int ,
  Price int ,
  Type varchar(20)
)
```

```
insert into Car( Name, Make, Model , Price, Type)
VALUES ('Corrolla', 'Toyota', 2015, 20000, 'Sedan'),
('Civic', 'Honda', 2018, 25000, 'Sedan'),
('Passo', 'Toyota', 2012, 18000, 'Hatchback'),
('Land Cruiser', 'Toyota', 2017, 40000, 'SUV'),
('Corrolla', 'Toyota', 2011, 17000, 'Sedan')
```

```
CREATE TABLE CarLog
(
    LogId int identity(1,1) primary key,
    CarId int ,
    CarName varchar(100),
)
```

دستورات بالا در شکل ۱۷۰-۳ اجرا شده است، و دیتابیس به همراه دو جدول ایجاد شده است.



شکل ۱۷۰-۳ / ایجاد دیتابیس

در ادامه می‌خواهیم از DML Trigger استفاده کنید تا رویدادهایی مانند Insert, Update, Delete در جدول اصلی یعنی در CAR صورت می‌گیرد به صورت اتوماتیک در جدول Carlog هم ثبت شود، برای این کار از دستورات زیر استفاده کنید.

```
CREATE TRIGGER [dbo].[CarLOG_INSERT] ON [dbo].[CarLog]
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @car_id INT, @car_name VARCHAR(50)
    SELECT @car_id = INSERTED.CarId, @car_name = INSERTED.CarName
    FROM INSERTED
    INSERT INTO CarLog
    VALUES(@car_id, @car_name)
END
```

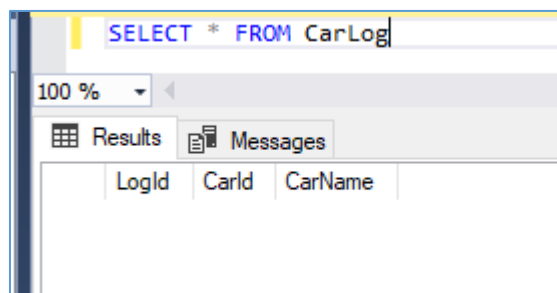
در دستورات بالا یک Trigger جدید با نام CarLOG_INSERT ایجاد می‌شود که زیرمجموعه جدول Carlog است، در قسمت بعدی با استفاده از دستور INSTEAD OF INSERT مشخص می‌کنیم که فقط داده‌هایی که وارد جدول

می‌شوند باید در جدول Carlog ثبت شوند بعد از این کار با دستور DECLARE دو متغیر با نام‌های @car_id و @car_name تعریف کردیم که و در ادامه با دستور SELECT این دو متغیر را با ستون‌های جدول اصلی برابر قرار دادیم و در ادامه گفتیم که اطلاعات را در جدول Carlog قرار دهد.

برای اینکه عملکرد Trigger را تست بگیریم قبل از هر کاری محتوای جدول Carlog را با دستور زیر مشاهده کنید:

```
SELECT * FROM CarLog
```

همان‌طور که در شکل ۳-۱۷۱ مشاهده می‌کنید جدول کاملاً خالی است



شکل ۳-۱۷۱

برای اینکه کارایی را مشاهده کنیم باید با دستور زیر اطلاعاتی را در جدول Car که جدول اصلی است وارد کنیم و بعد از آن به صورت اتوماتیک اطلاعات در جدول Carlog ثبت خواهد شد.

```
insert into Car( Name, Make, Model , Price, Type)
VALUES ('Pride', 'Kia', 2021, 100000, 'Sedan')
```

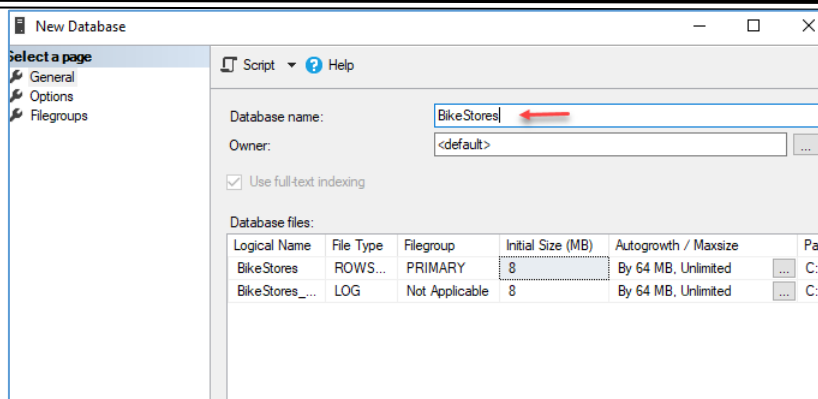
بعد از وارد کردن دستورات بالا اطلاعات جدید در جدول Car ثبت خواهد شد و بعد از آن به صورت اتوماتیک Car ID و Car Name در جدول Carlog ثبت خواهند شد.

۳-۱-۱۲ توابع در SQL Server 2019

زمانی که یک پروژه سنگین را به اتمام می‌رسانید و این پروژه دارای هزار خط کد است که اگر چنانچه با مشکل روبرو شود پیدا کردن خطا در آن بسیار پیچیده خواهد شد، با استفاده از توابع در SQL شما می‌توانید یک برنامه را بهبود ببخشید و برنامه را بهتر کنترل کنید، اگر بخواهیم به صورت کلی بگوییم با استفاده از توابع، برنامه ما به قطعه‌های منطقی جداگانه تبدیل می‌شود و یک تابع به این صورت عمل می‌کند که یک سری پارامتر را از ورودی می‌گیرد و عملیاتی را بر روی آن انجام می‌دهد و در خروجی نمایش می‌دهد، تابع را می‌توان یک شی در نظر گرفت که بعد از ایجاد در SQL ذخیره شده و هر زمان بخواهید می‌توانید آن را صدا بزنید.

برای اینکه توابع را بررسی کنیم یک دیتابیس به همراه جداول و مقادیر آن را در SQL ایجاد و در ادامه توابع را بر روی آنها اجرا خواهیم کرد:

به‌مانند شکل زیر یک Database با نام BikeStores ایجاد کنید، توجه کنید که دقیقاً این اسم را وارد کنید تا در ادامه تمرینات با مشکل روبرو نشویم.



بعد از ایجاد Database بالا با دستورات زیر جداول آن را ایجاد کنید:
برای دانلود آن هم می‌توانید به ID من در تلگرام (@farshidbabajani) پیام دهید.

```
CREATE SCHEMA production;
go
```

```
CREATE SCHEMA sales;
go
```

```
-- create tables
```

```
CREATE TABLE production.categories (
    category_id INT IDENTITY (1, 1) PRIMARY KEY,
    category_name VARCHAR (255) NOT NULL
);
```

```
CREATE TABLE production.brands (
    brand_id INT IDENTITY (1, 1) PRIMARY KEY,
    brand_name VARCHAR (255) NOT NULL
);
```

```
CREATE TABLE production.products (
    product_id INT IDENTITY (1, 1) PRIMARY KEY,
    product_name VARCHAR (255) NOT NULL,
    brand_id INT NOT NULL,
    category_id INT NOT NULL,
    model_year SMALLINT NOT NULL,
    list_price DECIMAL (10, 2) NOT NULL,
    FOREIGN KEY (category_id) REFERENCES production.categories (category_id) ON
DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (brand_id) REFERENCES production.brands (brand_id) ON DELETE
CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE sales.customers (
    customer_id INT IDENTITY (1, 1) PRIMARY KEY,
    first_name VARCHAR (255) NOT NULL,
    last_name VARCHAR (255) NOT NULL,
    phone VARCHAR (25),
    email VARCHAR (255) NOT NULL,
    street VARCHAR (255),
    city VARCHAR (50),
    state VARCHAR (25),
    zip_code VARCHAR (5)
);
```

```

CREATE TABLE sales.stores (
    store_id INT IDENTITY (1, 1) PRIMARY KEY,
    store_name VARCHAR (255) NOT NULL,
    phone VARCHAR (25),
    email VARCHAR (255),
    street VARCHAR (255),
    city VARCHAR (255),
    state VARCHAR (10),
    zip_code VARCHAR (5)
);

CREATE TABLE sales.staffs (
    staff_id INT IDENTITY (1, 1) PRIMARY KEY,
    first_name VARCHAR (50) NOT NULL,
    last_name VARCHAR (50) NOT NULL,
    email VARCHAR (255) NOT NULL UNIQUE,
    phone VARCHAR (25),
    active tinyint NOT NULL,
    store_id INT NOT NULL,
    manager_id INT,
    FOREIGN KEY (store_id) REFERENCES sales.stores (store_id) ON DELETE CASCADE ON
UPDATE CASCADE,

FOREIGN KEY (manager_id) REFERENCES sales.staffs (staff_id) ON DELETE NO ACTION ON
UPDATE NO ACTION
);

CREATE TABLE sales.orders (
    order_id INT IDENTITY (1, 1) PRIMARY KEY,
    customer_id INT,
    order_status tinyint NOT NULL,
    -- Order status: 1 = Pending; 2 = Processing; 3 = Rejected; 4 = Completed
    order_date DATE NOT NULL,
    required_date DATE NOT NULL,
    shipped_date DATE,
    store_id INT NOT NULL,
    staff_id INT NOT NULL,
    FOREIGN KEY (customer_id) REFERENCES sales.customers (customer_id) ON DELETE
CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (store_id) REFERENCES sales.stores (store_id) ON DELETE CASCADE ON
UPDATE CASCADE,
    FOREIGN KEY (staff_id) REFERENCES sales.staffs (staff_id) ON DELETE NO ACTION
ON UPDATE NO ACTION
);

CREATE TABLE sales.order_items (
    order_id INT,
    item_id INT,
    product_id INT NOT NULL,
    quantity INT NOT NULL,
    list_price DECIMAL (10, 2) NOT NULL,
    discount DECIMAL (4, 2) NOT NULL DEFAULT 0,
    PRIMARY KEY (order_id, item_id),
    FOREIGN KEY (order_id) REFERENCES sales.orders (order_id) ON DELETE CASCADE ON
UPDATE CASCADE,
    FOREIGN KEY (product_id) REFERENCES production.products (product_id) ON DELETE
CASCADE ON UPDATE CASCADE
);

CREATE TABLE production.stocks (
    store_id INT,

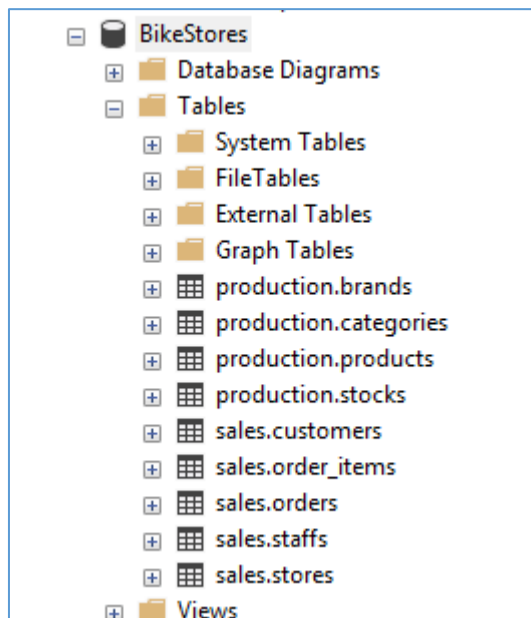
```

```

product_id INT,
quantity INT,
PRIMARY KEY (store_id, product_id),
FOREIGN KEY (store_id) REFERENCES sales.stores (store_id) ON DELETE CASCADE ON
UPDATE CASCADE,
FOREIGN KEY (product_id) REFERENCES production.products (product_id) ON DELETE
CASCADE ON UPDATE CASCADE
);

```

همان‌طور که در شکل زیر مشاهده می‌کنید جداول مورد نظر ایجاد شده‌اند:



در ادامه کار باید اطلاعات این جداول را پر کنیم، برای این کار باید از دستورات زیر استفاده کنیم: به‌خاطر اینکه تعداد دستورات زیاد است بهتر است از لینک زیر آن را دانلود و در SQL اجرا کنید:

<http://3isco.ir/Books/Load-data.sql>

```

-- Constraints
-- Triggers
-- Indexes
-- Statistics
production.products
  Columns
    product_id (PK int, not null)
    product_name (varchar(255), not null)
    brand_id (FK int, not null)
    category_id (FK int, not null)
    model_year (smallint, not null)
    list_price (decimal(10,2), not null)
  Keys
  Constraints
  Triggers
  Indexes
  Statistics
production.stocks
sales.customers
sales.order_items
sales.orders
sales.staffs
sales.stores
Views
External Resources
Synonyms
Programmability
Service Broker
Storage

SET IDENTITY_INSERT production.brands ON;

INSERT INTO production.brands (brand_id, brand_name) VALUES (1, 'Electra')
INSERT INTO production.brands (brand_id, brand_name) VALUES (2, 'Haro')
INSERT INTO production.brands (brand_id, brand_name) VALUES (3, 'Heller')
INSERT INTO production.brands (brand_id, brand_name) VALUES (4, 'Pure Cycles')
INSERT INTO production.brands (brand_id, brand_name) VALUES (5, 'Ritchey')
INSERT INTO production.brands (brand_id, brand_name) VALUES (6, 'Strider')
INSERT INTO production.brands (brand_id, brand_name) VALUES (7, 'Sun Bicycles')
INSERT INTO production.brands (brand_id, brand_name) VALUES (8, 'Surly')
INSERT INTO production.brands (brand_id, brand_name) VALUES (9, 'Trek')

SET IDENTITY_INSERT production.brands OFF;

SET IDENTITY_INSERT production.categories ON;

INSERT INTO production.categories (category_id, category_name) VALUES (1, 'Children Bicycles')
INSERT INTO production.categories (category_id, category_name) VALUES (2, 'Comfort Bicycles')
INSERT INTO production.categories (category_id, category_name) VALUES (3, 'Cruisers Bicycles')
INSERT INTO production.categories (category_id, category_name) VALUES (4, 'Cyclocross Bicycles')
INSERT INTO production.categories (category_id, category_name) VALUES (5, 'Electric Bikes')
INSERT INTO production.categories (category_id, category_name) VALUES (6, 'Mountain Bikes')
INSERT INTO production.categories (category_id, category_name) VALUES (7, 'Road Bikes')

SET IDENTITY_INSERT production.categories OFF;

SET IDENTITY_INSERT production.products ON;

INSERT INTO production.products (product_id, product_name, brand_id, category_id, model_year, list_price) VALUES (1, 'Trek')
INSERT INTO production.products (product_id, product_name, brand_id, category_id, model_year, list_price) VALUES (2, 'Rit')
INSERT INTO production.products (product_id, product_name, brand_id, category_id, model_year, list_price) VALUES (3, 'Sur')
INSERT INTO production.products (product_id, product_name, brand_id, category_id, model_year, list_price) VALUES (4, 'Hel')
INSERT INTO production.products (product_id, product_name, brand_id, category_id, model_year, list_price) VALUES (5, 'Hel')

```

توابع ایجاد شده توسط کاربر در SQL Server به شما کمک می‌کنند تا با کپسوله‌سازی منطق‌های کسب‌وکار پیچیده و ایجاد قابلیت استفاده مجدد از آن‌ها در هر query، فرایند توسعه نرم‌افزار خود را تسهیل کنید. در این بخش مباحث زیر مورد بررسی قرار می‌گیرند:

- توابع اسکالر تعریف شده توسط کاربر: توابع اسکالر تعریف شده توسط کاربر را پوشش می‌دهد. این توابع به شما امکان می‌دهند تا فرمول‌ها یا منطق‌های کسب‌وکار پیچیده را کپسوله‌سازی کنید و در همه queryها مجدداً از آنها استفاده کنید.
- متغیرهای جدول: چگونگی استفاده از متغیرهای جدول به‌عنوان یک مقدار بازگشتی از توابع تعریف شده توسط کاربر را خواهید آموخت.
- توابع دارای مقدار جدول: شما را با تابع دارای مقدار جدول تک خطی و تابع دارای مقدار جدول چند کدی آشنا می‌کند تا بتوانید توابع تعریف شده توسط کاربر ایجاد کنید که داده‌های نوع‌های جدول را بازگردانند.
- حذف توابع تعریف شده توسط کاربر: چگونگی حذف یک یا چند تابع تعریف شده توسط کاربر موجود از پایگاه داده را توضیح می‌دهند.

۱-۱۲-۱-۳ توابع اسکالر در SQL Server

تابع اسکالر در SQL Server یک یا چند پارامتر می‌گیرد و یک مقدار باز می‌گرداند. توابع اسکالر به شما کمک می‌کنند کدتان را ساده‌سازی کنید. برای مثال، ممکن است یک محاسبه پیچیده داشته باشید که در queryهای زیادی از آن استفاده شده است. به‌جای درج فرمول در همه queryها، می‌توانید یک تابع اسکالر ایجاد کنید که فرمول را کپسوله‌سازی کرده و از آن در queryها استفاده کند. ایجاد یک تابع اسکالر

برای ایجاد یک تابع اسکالر، می‌توانید از کد CREATE FUNCTION به شکل زیر استفاده کنید:

```
CREATE FUNCTION [schema_name.]function_name (parameter_list)
RETURN data_type AS
BEGIN
    statements
    RETURN value
END
```

در این syntax:

- اول، نام تابع بعد از کلیدواژه‌های CREATE FUNCTION مشخص شده است. نام شما اختیاری است. اگر صریحاً نام شما را مشخص نکنید، SQL Server به‌صورت پیش‌فرض از dbo استفاده می‌کند.
- دوم، لیستی از پارامترها در بین پرانتزها بعد از نام تابع مشخص شده‌اند.
- سوم، نوع داده از مقدار بازگشتی در کد RETURN مشخص شده است.

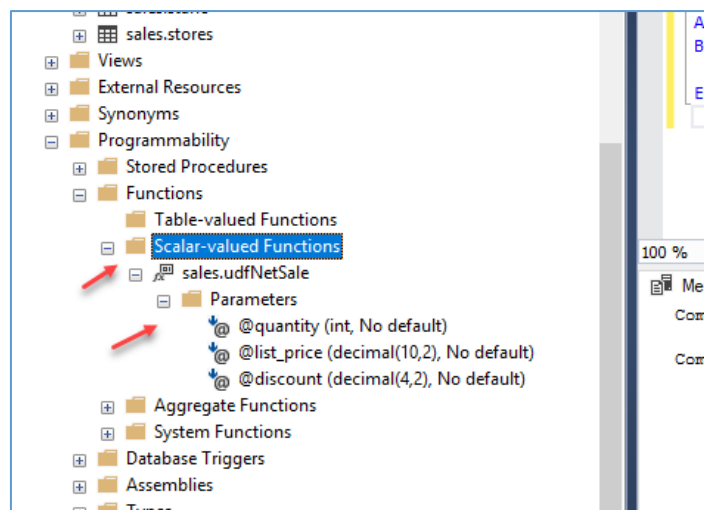
- چهارم، داخل بدنه تابع، از دستور RETURN باید استفاده شود تا یک مقدار بازگردانده شود.

مثال زیر یک تابع ایجاد می‌کند که فروش خالص را بر اساس کمیت، قیمت و تخفیف محاسبه می‌کند:

```
CREATE FUNCTION sales.udfNetSale(
    @quantity INT,
    @list_price DEC(10,2),
    @discount DEC(4,2)
)
RETURNS DEC(10,2)
AS
BEGIN
    RETURN @quantity * @list_price * (1 - @discount);
END;
```

سپس، می‌توانید از این کد برای محاسبه فروش خالص هر سفارش فروش در جدول order_items از پایگاه داده Bikestores استفاده کنید.

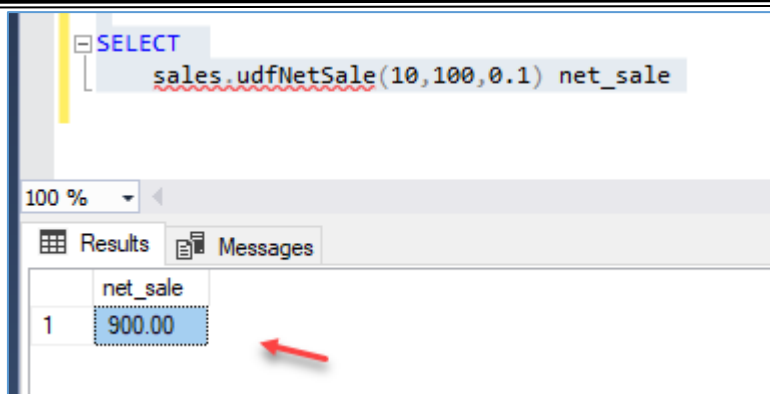
بعد از ایجاد تابع اسکالر، می‌توانید آن را در آدرس programmability > Functions > Scalar-valued Functions در تصویر زیر مشاهده می‌کنید بیابید:



فراخوانی تابع اسکالر دقیقاً همانند فراخوانی یک تابع معمولی است. برای مثال، کد زیر چگونگی فراخوانی تابع udfNetSale را نشان می‌دهد:

```
SELECT
    sales.udfNetSale(10,100,0.1) net_sale
```

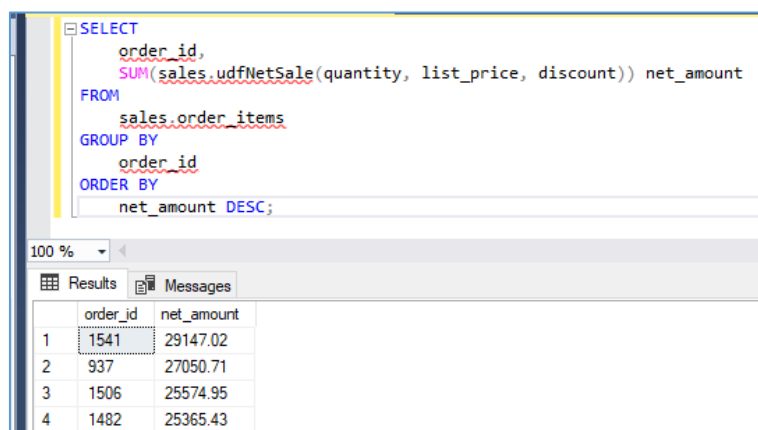
خروجی به این شکل است:



مثال زیر، چگونگی استفاده از تابع Sales.udfNetSale برای به دست آوردن فروش خالص سفارش‌های فروش در جدول order_items را نشان می‌دهد:

```
SELECT
    order_id,
    SUM(sales.udfNetSale(quantity, list_price, discount)) net_amount
FROM
    sales.order_items
GROUP BY
    order_id
ORDER BY
    net_amount DESC;
```

تصویر زیر بخشی از خروجی را نمایش می‌دهد:



تغییر دادن یک تابع اسکالر

برای تغییر دادن یک تابع اسکالر، می‌توانید به جای کلیدواژه CREATE از دستور ALTER استفاده کنید. بقیه کدها دست‌نخورده باقی می‌مانند:

```
ALTER FUNCTION [schema_name.]function_name (parameter_list)
    RETURN data_type AS
BEGIN
    statements
    RETURN value
END
```

توجه کنید که اگر تابع تعریف شده توسط کاربر وجود ندارد، می‌توانید از دستور CREATE OR ALTER برای ایجاد یک تابع تعریف شده توسط کاربر یا برای تغییر دادن یک تابع اسکالر از پیش موجود استفاده کنید:

```
CREATE OR ALTER FUNCTION [schema_name.]function_name (parameter_list)
RETURN data_type AS
BEGIN
    statements
RETURN value
END
```

حذف یک تابع اسکالر

برای حذف یک تابع اسکالر از پیش موجود، می‌توانید از دستور DROP FUNCTION استفاده کنید:

```
DROP FUNCTION [schema_name.]function_name;
```

برای مثال، برای حذف تابع Sales.udfNetSale می‌توانید از کد زیر استفاده کنید:

```
DROP FUNCTION sales.udfNetSale;
```

نکات تابع اسکالر در SQL Server

در لیست زیر نکات کلیدی توابع اسکالر را مشاهده می‌کنید:

- توابع اسکالر را تقریباً می‌توان در همه‌جا درون کدهای T-SQL به کار برد.
- توابع اسکالر یک یا چند پارامتر می‌پذیرند، اما فقط یک مقدار باز می‌گردانند، در نتیجه باید شامل یک دستور RETURN باشند.
- توابع اسکالر می‌توانند از منطق‌هایی مانند قطعات کد IF یا حلقه‌های WHILE استفاده کنند.
- توابع اسکالر نمی‌توانند داده‌ها را update کنند. می‌توانند به داده‌ها دسترسی داشته باشند، اما این کار اصلاً توصیه نمی‌شود.
- توابع اسکالر می‌توانند توابع دیگر را فراخوانی کنند.

به این صورت، چگونگی استفاده از توابع اسکالر برای کپسوله‌سازی فرمول‌ها یا منطق‌های کسب‌وکار پیچیده و استفاده مجدد از آن‌ها در Queryها را آموختید.

۲-۱-۳-۱-۲-۳ توابع تاریخ یا Date

در این قسمت به کار با توابع تاریخ یا Date خواهیم پرداخت. در ادامه لیستی از توابع تاریخ در SQL Server را مشاهده می‌کنید که به شما اجازه می‌دهند داده‌های تاریخ و زمان را به صورت مؤثر و کاربردی مدیریت کنید.

توابع بازگرداننده تاریخ و زمان فعلی

توضیح	تابع
تاریخ و زمان فعلی سیستم را بدون بخش محدوده زمانی باز می‌گرداند.	CURRENT_TIMESTAMP

بخش تاریخ را به صورت یک عدد integer بازمی گرداند.	GETUTCDATE
تاریخ و زمان فعلی سیستم عاملی که SQL Server در حال اجرا شدن روی آن است را بازمی گرداند.	GETDATE
تاریخ و زمان فعلی سیستم را همراه با دقت ثانیه‌های اعشار بیشتر نسبت به تابع GETDATE() بازمی گرداند.	SYSDATETIME
تاریخ و زمان سیستم فعلی در زمان UTC را بازمی گرداند.	SYSUTCDATETIME
تاریخ و زمان سیستم فعلی همراه با محدوده زمانی را بازمی گرداند.	SYSDATETIMEOFFSET

توابع بازگرداننده بخش‌های تاریخ و زمان

توضیح	تابع
بخش تاریخ را به عنوان یک کاراکتر رشته بازمی گرداند.	DATENAME
بخش تاریخ را به عنوان یک عدد integer بازمی گرداند.	DATEPART
روز یک تاریخ خاص را به عنوان یک عدد integer بازمی گرداند.	DAY
ماه یک تاریخ خاص را به عنوان یک عدد integer بازمی گرداند.	MONTH
سال یک تاریخ را به عنوان یک عدد integer بازمی گرداند.	YEAR

تابع بازگرداننده اختلاف بین دو تاریخ

توضیح	تابع
یک مقدار به بخش تاریخ اضافه می‌کند و مقدار تاریخ جدید را بازمی گرداند.	DATEADD
آخرین روز ماه موجود در تاریخ مشخص شده را همراه با یک offset اختیاری بازمی گرداند.	EOMONTH
Offset منطقه زمانی یک مقدار DATETIMEOFFSET را تغییر می‌دهد و مقدار UTC را حفظ می‌کند.	SWITCHOFFSET
یک مقدار DATETIME2 را به یک مقدار DATETIMEOFFSET تبدیل می‌کند.	TODATETIMEOFFSET

توابع ایجاد تاریخ و زمان از بخش‌های مخصوص به خودشان

توضیح	تابع
یک مقدار DATE از سال، ماه و روز بازمی گرداند.	DATEFROMPARTS

یک مقدار DATETIME2 از آرگومان‌های تاریخ و زمان بازمی‌گرداند.	DATETIME2FROMPARTS
یک مقدار DATETIMEOFFSET از آرگومان‌های تاریخ و زمان بازمی‌گرداند.	DATETIMEOFFSETFROMPARTS
یک مقدار TIME از بخش‌های زمان همراه با دقت بیشتر بازمی‌گرداند.	TIMEFROMPARTS

تابع ارزیابی مقادیر تاریخ و زمان

توضیح	تابع
بررسی می‌کند که آیا یک مقدار یک تاریخ، زمان یا Datetime معتبر هست یا نه.	ISDATE

تابع CURRENT_TIMESTAMP در SQL Server

تابع CURRENT_TIMESTAMP برچسب زمان سیستم عامل سروری که SQL Server Database روی آن اجرا می‌شود را بازمی‌گرداند. برچسب زمان بازگشتی یک مقدار DATETIME بدون offset محدوده زمانی است.

تابع CURRENT_TIMESTAMP هیچ آرگومانی نمی‌گیرد:

```
CURRENT_TIMESTAMP
```

CURRENT_TIMESTAMP یک معادل ANSI SQL برای GETDATE() است.

می‌توانید از تابع CURRENT_TIMESTAMP در هر جایی که یک عبارت DATETIME قابل قبول باشد، استفاده کنید.

اجازه دهید چند مثال برای استفاده از تابع CURRENT_TIMESTAMP حل کنیم.

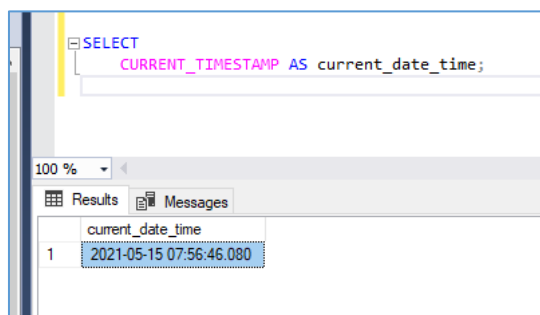
الف) مثال ساده CURRENT_TIMESTAMP

مثال زیر از تابع CURRENT_TIMESTAMP برای بازگرداندن تاریخ و زمان فعلی استفاده می‌کند:

```
SELECT
```

```
    CURRENT_TIMESTAMP AS current_date_time;
```

خروجی به این ترتیب است:



ب) مثال استفاده از تابع CURRENT_TIMESTAMP به عنوان یک مقدار پیش فرض برای ستون‌های جدول

ابتدا، یک جدول جدید به نام current_timestamp_demos ایجاد می‌کنیم که ستون created_at در آن یک مقدار پیش فرض را به عنوان برچسب زمان می‌پذیرد که در آن یک تابع اضافه شده است:

```
CREATE TABLE current_timestamp_demos
(
  id          INT IDENTITY,
  msg        VARCHAR(255) NOT NULL,
  created_at DATETIME NOT NULL
             DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY(id)
);
```

سپس، دو ردیف به جدول اضافه می‌کنیم:

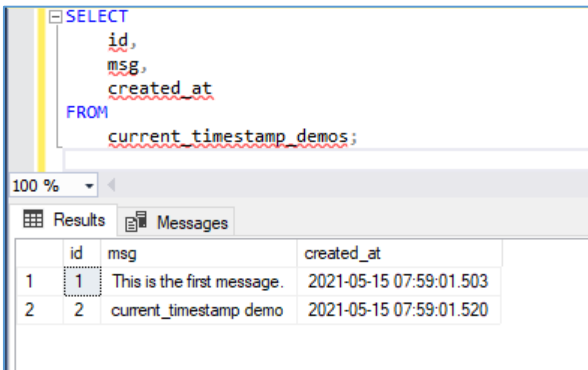
```
INSERT INTO current_timestamp_demos(msg)
VALUES('This is the first message.');
```

```
INSERT INTO current_timestamp_demos(msg)
VALUES('current_timestamp demo');
```

در آخر، داده‌ها را از جدول current_timestamp_demos به دست می‌آوریم:

```
SELECT
  id,
  msg,
  created_at
FROM
  current_timestamp_demos;
```

خروجی به این شکل است:



id	msg	created_at
1	This is the first message.	2021-05-15 07:59:01.503
2	current_timestamp demo	2021-05-15 07:59:01.520

همان‌طور که به وضوح در خروجی مشاهده می‌کنید، مقادیر موجود در ستون created_at مقدار برچسب زمان بازگشتی توسط تابع CURRENT_TIMESTAMP را می‌گیرند.

در نتیجه، چگونگی استفاده از تابع CURRENT_TIMESTAMP برای بازگرداندن برچسب زمان سیستم پایگاه داده به عنوان یک مقدار DATETIME را آموختید.

تابع GETUTCDATE در SQL Server

تابع GETUTCDATE() زمان UTC فعلی را بازمی‌گرداند. تابع GETUTCDATE() این مقدار را از سیستم عاملی که در آن اجرا می‌شود، محاسبه می‌کند.

تابع GETUTCDATE() دارای Syntax زیر است:

```
GETUTCDATE()
```

مثال GETUTCDATE() در SQL Server

این مثال از توابع GETDATE()، GETUTCDATE() و DATEDIFF() برای بازگرداندن زمان محلی، زمان UTC و محدوده زمانی سرور استفاده می‌کند:

```
DECLARE
    @local_time DATETIME,
    @utc_time DATETIME;

SET @local_time = GETDATE();
SET @utc_time = GETUTCDATE();

SELECT
    CONVERT(VARCHAR(40), @local_time)
        AS 'Server local time';
SELECT
    CONVERT(VARCHAR(40), @utc_time)
        AS 'Server UTC time'
SELECT
    CONVERT(VARCHAR(40), DATEDIFF(hour, @utc_time, @local_time))
        AS 'Server time zone';
GO
```

خروجی دستورات بالا را در شکل زیر مشاهده می‌کنید، در این مثال، چگونگی استفاده از تابع GETUTCDATE() برای به دست آوردن زمان UTC فعلی را آموختید.

```

CONVERT(VARCHAR(40), @local_time)
AS 'Server local time';
SELECT
CONVERT(VARCHAR(40), @utc_time)
AS 'Server UTC time'
SELECT
CONVERT(VARCHAR(40), DATEDIFF(hour, @utc_time, @local_time))

```

Server local time	
1	May 15 2021 8:03AM

Server UTC time	
1	May 15 2021 3:33AM

Server time zone	
1	5

تابع GETDATE در SQL Server

تابع GETDATE() برچسب زمان فعلی سیستم را به عنوان یک مقدار DATETIME بدون offset منطقه زمانی پایگاه داده باز می گرداند. مقدار DATETIME از سیستم عامل سرور به دست می آید که نمونه SQL Server در آن اجرا می شود. کد زیر syntax تابع GETDATE() را نشان می دهد:

GETDATE()

توجه کنید که تابع GETDATE() یک تابع غیرمتمرکز است، از این رو، نمی توانید برای ستون هایی که این تابع را رفرنس کرده اند یک index در view ها ایجاد کنید.

اجازه دهید چند مثال برای استفاده از تابع GETDATE() حل کنیم.

الف) مثال استفاده از تابع GETDATE() برای به دست آوردن تاریخ و زمان فعلی

این مثال از تابع GETDATE() برای بازگرداندن تاریخ و زمان فعلی سیستم عاملی که SQL Server روی آن اجرا می شود، استفاده می کند:

SELECT

```
GETDATE() current_date_time;
```

خروجی به این شکل است:

```

SELECT
GETDATE() current_date_time;

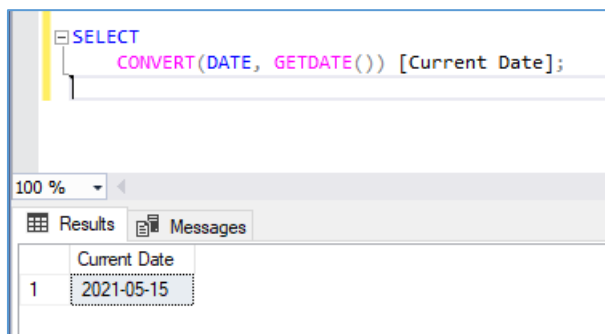
```

current_date_time	
1	2021-05-15 08:06:18.613

ب) مثال استفاده از تابع GETDATE() برای به دست آوردن تاریخ فعلی سیستم
 برای به دست آوردن تاریخ فعلی، می توانید از تابع CONVERT() برای تبدیل مقدار DATETIME به یک مقدار DATE استفاده کنید که در کد زیر مشاهده می کنید:

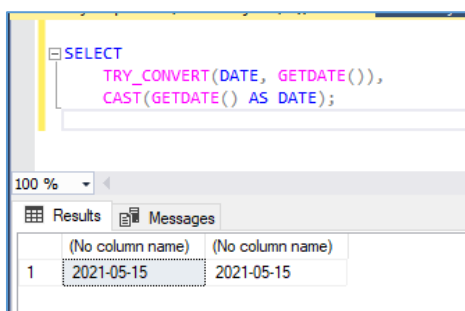
```
SELECT
    CONVERT(DATE, GETDATE()) [Current Date];
```

کد زیر نیز خروجی را نشان می دهد:



به طور مشابه، می توانید از توابع TRY_CONVERT() و CAST() برای تبدیل نتیجه تابع GETDATE() به یک تاریخ استفاده کنید:

```
SELECT
    TRY_CONVERT(DATE, GETDATE()),
    CAST(GETDATE() AS DATE);
```



پ) مثال استفاده از تابع GETDATE() برای به دست آوردن زمان فعلی سیستم
 برای به دست آوردن زمان فعلی، می توانید از تابع CONVERT(), TRY_CONVERT() یا CAST() برای تبدیل نتیجه تابع GETDATE() به یک زمان استفاده کنید:

```
SELECT
    CONVERT(TIME, GETDATE()),
    TRY_CONVERT(TIME, GETDATE()),
    CAST(GETDATE() AS TIME);
```

```
SELECT
  CONVERT(TIME, GETDATE()),
  TRY_CONVERT(TIME, GETDATE()),
  CAST(GETDATE() AS TIME);
```

	(No column name)	(No column name)	(No column name)
1	08:09:11.5000000	08:09:11.5000000	08:09:11.5000000

به این شکل، چگونگی استفاده از تابع GETDATE() برای بازگرداندن تاریخ و زمان فعلی سیستم را نیز آموختید.

تابع SYSDATETIME در SQL Server

تابع SYSDATETIME() یک مقدار DATETIME2 را بازمی‌گرداند که نشان‌دهنده تاریخ و زمان فعلی سروری است که نمونه در آن اجرا می‌شود.

تابع SYSDATETIME() هیچ پارامتری نمی‌گیرد:

SYSDATETIME()

خروجی به این شکل است:

```
Select SYSDATETIME()
```

	(No column name)
1	2021-05-15 08:41:43.7522648

توجه کنید که تابع SYSDATETIME() دارای دقت ثانیه‌های اعشاری بیشتری نسبت به تابع GETDATE() است.

تابع SYSDATETIME() یک تابع غیرمتمرکز است، از این رو، viewها و ستون‌هایی که دارای عبارات رفرنسی به سمت این تابع هستند را نمی‌توان index گذاری کرد.

بیاید چند مثال برای استفاده از تابع SYSDATETIME() حل کنیم.

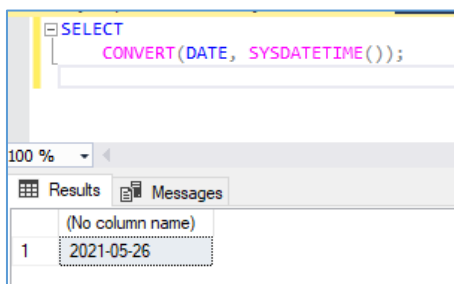
الف) بازگرداندن تاریخ فعلی سیستم

این مثال از تابع CONVERT() برای تبدیل نتیجه تابع SYSDATETIME() به تاریخ فعلی استفاده می‌کند:

SELECT

```
CONVERT(DATE, SYSDATETIME());
```

نتیجه به این صورت است:

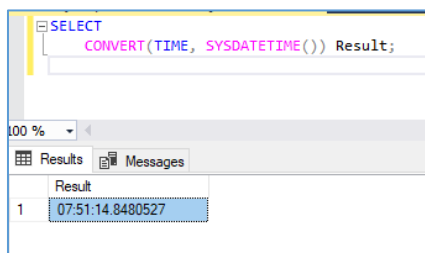


ب) بازگرداندن زمان فعلی سیستم

مثال زیر نتیجه تابع SYSDATETIME() را به زمان فعلی تبدیل می‌کند:

```
SELECT  
CONVERT(TIME, SYSDATETIME()) Result;
```

خروجی به این شکل است:



به این شکل، چگونگی استفاده از تابع SYSDATETIME() برای به‌دست‌آوردن تاریخ و زمان فعلی را نیز آموختید.

تابع SYSUTCDATETIME در SQL Server

تابع SYSUTCDATETIME() یک مقدار DATETIME2 بازمی‌گرداند که تاریخ و زمان فعلی سروری که نمونه SQL Server روی آن اجرا می‌شود را نشان می‌دهد. Datetime به فرمت زمان هماهنگی جهانی (UTC) می‌باشد.

Syntax تابع SYSUTCDATETIME() به‌صورت زیر است:

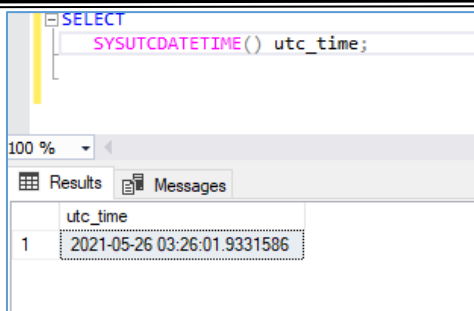
```
SYSUTCDATETIME()
```

برای مثال، برای به‌دست‌آوردن تاریخ و زمان فعلی در UTC، می‌توانید از کد زیر استفاده کنید:

```
SELECT  
SYSUTCDATETIME() utc_time;
```

خروجی به این صورت است:


```
SELECT
  SYSUTCDATETIME() utc_time;
```



	utc_time
1	2021-05-26 03:26:01.9331586

توجه کنید که تابع `SYSTCDATETIME()` دارای دقت ثانیه‌های اعشاری بیشتری نسبت به تابع `GETUTCDATE()` است.

مثال‌های تابع `SYSTCDATETIME()` در SQL Server

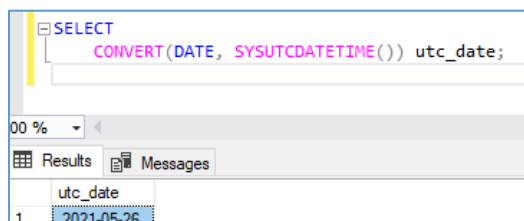
الف) مثال بازگرداندن تاریخ فعلی به فرم زمان UTC

مثال زیر از تابع `CONVERT()` برای تبدیل نتیجه تابع `SYSTCDATETIME()` به تاریخ فعلی به فرم زمان UTC استفاده می‌کند:

```
SELECT
  CONVERT(DATE, SYSUTCDATETIME()) utc_date;
```

خروجی به این شکل است:

```
SELECT
  CONVERT(DATE, SYSUTCDATETIME()) utc_date;
```



	utc_date
1	2021-05-26

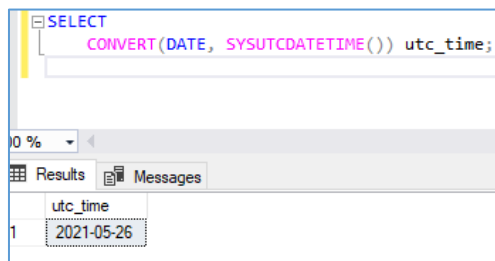
ب) مثال بازگرداندن زمان فعلی به فرم زمان UTC

این مثال نتیجه تابع `SYSTCDATETIME()` را به تاریخ فعلی به فرم زمان UTC تبدیل می‌کند:

```
SELECT
  CONVERT(DATE, SYSUTCDATETIME()) utc_time;
```

خروجی به این شکل است:

```
SELECT
  CONVERT(DATE, SYSUTCDATETIME()) utc_time;
```



	utc_time
1	2021-05-26

به این صورت چگونگی استفاده از تابع SYSTCDATETIME() برای به دست آوردن تاریخ و زمان فعلی به فرم زمان UTC را آموختید.

تابع SYSDATETIMEOFFSET در SQL Server

تابع SYSDATETIMEOFFSET() یک مقدار DATETIMEOFFSET (V) را بازمی گرداند که تاریخ و زمان فعلی را نشان می دهد و همچنین شامل منطقه زمانی رایانه ای که نمونه SQL Server روی آن اجرا می شود نیز هست.

کد زیر syntax تابع SYSDATETIMEOFFSET() را نشان می دهد:

SYSDATETIMEOFFSET()

برای مثال، برای به دست آوردن تاریخ و زمان فعلی همراه با منطقه زمانی سروری که به آن متصل هستید، می توانید از کد زیر استفاده کنید:

```
SELECT
    SYSDATETIMEOFFSET() [datetimeoffset with timezone];
```

خروجی به این شکل است:

datetimeoffset with timezone
2021-05-26 08:00:40.5387566 +04:30

مثال های تابع SYSDATETIMEOFFSET() در SQL Server

بیاید چند مثال برای استفاده از تابع SYSDATETIMEOFFSET() حل کنیم.

الف) مثال استخراج offset منطقه زمانی

مثال زیر از تابع DATEPART() برای بازگرداندن offset منطقه زمانی استفاده کرده است. این مثال یک integer بازمی گرداند که نشان دهنده offset منطقه زمانی به دقیقه است.

```
SELECT
    SYSDATETIMEOFFSET() AS [System DateTime Offset],
    DATEPART(TZoffset, SYSDATETIMEOFFSET()) AS [Timezone Offset];
```

خروجی به این شکل است:

	System DateTime Offset	Timezone Offset
1	2021-05-26 08:02:08.1413650 +04:30	270

(ب) مثال فرمت بندی offset منطقه زمانی

این مثال offset منطقه زمانی بازگشتی را با استفاده از تابع FORMAT() و آرگومان های zz و zzz به شکل رشته باز می گرداند:

SELECT

```
SYSDATETIMEOFFSET() AS 'System Date Time Offset',
FORMAT(SYSDATETIMEOFFSET(), 'zz') AS 'zz',
FORMAT(SYSDATETIMEOFFSET(), 'zzz') AS 'zzz';
```

تصویر زیر، خروجی را نشان می دهد:

	System Date Time Offset	zz	zzz
1	2021-05-26 08:02:58.8460054 +04:30	+04	+04:30

به این شکل، چگونگی استفاده از تابع SYSDATETIMEOFFSET() برای به دست آوردن تاریخ و زمان فعلی، همراه با منطقه زمانی را آموختید.

تابع DATENAME در SQL Server

تابع DATENAME() یک رشته (نوع NVARCHAR) باز می گرداند که نشان دهنده یک بخش تاریخ خاص است (یعنی سال، ماه و روز یک تاریخ خاص).

کد زیر syntax تابع DATENAME() را نشان می دهد:

DATENAME(date_part, input_date)

تابع DATENAME() دو آرگومان می پذیرد:

- Date_part بخشی از تاریخ است که می خواهید بازگشت داده شود. جدول زیر تمام مقادیر بخش تاریخ معتبر را لیست کرده است.
- Input_date یک تاریخ لفظی یا یک عبارت است که می توانیم آن را به یک مقدار DATETIME2، DATETIME، SMALLDATETIME، DATE، TIME یا DATETIMEOFFSET تبدیل کنیم.

اختصارات	Date_Part
yy, yyyy	سال
qq, q	ربع سال

mm, m	ماه
dy, y	روز از سال
dd, d	روز
wk, ww	هفته
dw	روز از هفته
hh	ساعت
mi, n	دقیقه
ss, s	ثانیه
ms	میلی ثانیه
mcs	میکروثانیه
ns	نانو ثانیه
tz	Tzoffset
isowk, isoww	ISO_WEEK

DATEPART() در برابر DATENAME()

توجه کنید که DATENAME() از لحاظ نوع بازگشتی، شبیه به DATEPART() است. تابع DATENAME() بخش تاریخ را به عنوان یک کاراکتر رشته بازمی گرداند، در حالی که DATEPART() بخش تاریخ را به عنوان یک integer بازمی گرداند. مثال زیر را ببینید:

```
SELECT
    DATEPART(year, '2018-05-10') [datepart],
    DATENAME(year, '2018-05-10') [datename];
```

خروجی به مانند شکل زیر خواهد بود.

	datepart	datename
1	2018	2018

به هر حال، نوع‌های داده‌ای آنها همان‌طور که در مثال زیر مشاهده می‌کنید، متفاوت هستند:

```
SELECT
    DATEPART(year, '2018-05-10') + '1' [datepart],
    DATENAME(year, '2018-05-10') + '1' [datename];
```

کد زیر نتیجه را نشان می‌دهد:

```

SELECT
    DATEPART(year, '2018-05-10') + '1' [datepart],
    DATENAME(year, '2018-05-10') + '1' [datetime] ;

```

	datepart	datetime
1	2019	20181

چون تابع DATEPART() یک integer باز می‌گرداند، پس مقدار عبارت برابر با ۲۰۱۹ (۲۰۱۸ + ۱) می‌شود. به‌هرحال، تابع DATENAME() یک رشته کاراکتر باز می‌گرداند، از این رو، علامت + به‌عنوان یک عملگر تلفیقی عمل می‌کند و نتیجه آن '۲۰۱۸۱' (یعنی ۲۰۱۸ + ۱) است.

مثال تابع DATENAME() در SQL Server

این مثال از تابع DATENAME() برای بازگرداندن بخش‌های تاریخ مختلف از '۰۸:۱۰ + ۱۲۳۴۵۶۷.۱۰:۲۰:۳۰ -۰۲' استفاده می‌کند:

```
DECLARE @dt DATETIME2= '2020-10-02 10:20:30.1234567 +08:10';
```

```

SELECT 'year,yyy,yy' date_part,
    DATENAME(year, @dt) result
UNION
SELECT 'quarter, qq, q',
    DATENAME(quarter, @dt)
UNION
SELECT 'month, mm, m',
    DATENAME(month, @dt)
UNION
SELECT 'dayofyear, dy, y',
    DATENAME(dayofyear, @dt)
UNION
SELECT 'day, dd, d',
    DATENAME(day, @dt)
UNION
SELECT 'week, wk, ww',
    DATENAME(week, @dt)
UNION
SELECT 'weekday, dw, w',
    DATENAME(week, @dt)
UNION
SELECT 'hour, hh' date_part,
    DATENAME(hour, @dt)
UNION
SELECT 'minute, mi,n',
    DATENAME(minute, @dt)
UNION
SELECT 'second, ss, s',
    DATENAME(second, @dt)
UNION
SELECT 'millisecond, ms',
    DATENAME(millisecond, @dt)
UNION
SELECT 'microsecond, mcs',
    DATENAME(microsecond, @dt)

```

```

UNION
SELECT 'nanosecond, ns',
    DATENAME(nanosecond, @dt)
UNION
SELECT 'TZoffset, tz',
    DATENAME(tz, @dt)
UNION
SELECT 'ISO_WEEK, ISOWK, ISOWW',
    DATENAME(ISO_WEEK, @dt);

```

خروجی به این ترتیب است:

date_part	result
1 day, dd, d	2
2 dayofyear, dy, y	276
3 hour, hh	10
4 ISO_WEEK, ISOWK, ISOWW	40
5 microsecond, mcs	123456
6 millisecond, ms	123
7 minute, mi, n	20
8 month, mm, m	October
9 nanosecond, ns	123456700
10 quarter, qq, q	4
11 second, ss, s	30
12 TZoffset, tz	+00:00
13 week, wk, ww	40
14 weekday, dw, w	40
15 year, yyyy, yy	2020

در نتیجه، چگونگی استفاده از تابع DATENAME() برای استخراج یک بخش تاریخ به عنوان یک رشته کاراکتر از یک تاریخ را آموختید.

تابع DATEPART در SQL Server

تابع DATEPART() یک integer بازمی‌گرداند که بخشی از یک تاریخ مانند روز، ماه و سال است.

کد زیر syntax تابع DATEPART() را نشان می‌دهد:

```
DATEPART ( date_part , input_date )
```

DATEPART() دو آرگومان می‌گیرد:

- Date_part بخشی از یک تاریخ است که باید استخراج شود. (بخش‌های معتبر تاریخ را در جدول زیر مشاهده می‌کنید).

○ Input_date تاریخی است که از آن بخش تاریخ استخراج می‌شود.

اختصارات	Date_part
yy, yyyy	سال
qq, q	ربع سال
mm, m	ماه
dy, y	روز از سال
dd, d	روز
wk, ww	هفته
dw	روز از هفته
hh	ساعت
mi, n	دقیقه
ss, s	ثانیه
ms	میلی ثانیه
mcs	میکروثانیه
ns	نانو ثانیه
tz	Tzoffset
isowk, isoww	ISO_WEEK

مثال‌های DATEPART() در SQL Server

اجازه دهید چند مثال برای استفاده از تابع DATEPART() حل کنیم.

الف) استفاده از تابع DATEPART() همراه با متغیرها

این مثال از تابع DATEPART() برای استخراج بخش‌های مختلف یک مقدار تاریخ ذخیره شده در یک متغیر استفاده

می‌کند:

```
DECLARE @d DATETIME = '2019-01-01 14:30:14';
SELECT
    DATEPART(year, @d) year,
    DATEPART(quarter, @d) quarter,
    DATEPART(month, @d) month,
    DATEPART(day, @d) day,
    DATEPART(hour, @d) hour,
    DATEPART(minute, @d) minute,
    DATEPART(second, @d) second;
```

خروجی به‌این ترتیب است:

```

DECLARE @d DATETIME = '2019-01-01 14:30:14';
SELECT
    DATEPART(year, @d) year,
    DATEPART(quarter, @d) quarter,
    DATEPART(month, @d) month,
    DATEPART(day, @d) day,
    DATEPART(hour, @d) hour,
    DATEPART(minute, @d) minute,
    DATEPART(second, @d) second;

```

year	quarter	month	day	hour	minute	second
2019	1	1	1	14	30	14

ب) مثال استفاده از تابع DATEPART() همراه با ستون‌های جدول از جداول sales.orders و sales.order_items در پایگاه داده Bikestores برای این مثال استفاده می‌کنیم: مثال زیر از تابع DATEPART() برای به دست آوردن نرخ رشد فروش بر اساس سال، ربع سال، ماه و روز استفاده می‌کند.

```

SELECT DATEPART(year, shipped_date) [year],
    DATEPART(quarter, shipped_date) [quarter],
    DATEPART(month, shipped_date) [month],
    DATEPART(day, shipped_date) [day],
    SUM(quantity * list_price) gross_sales
FROM sales.orders o
    INNER JOIN sales.order_items i ON i.order_id = o.order_id
WHERE shipped_date IS NOT NULL
GROUP BY DATEPART(year, shipped_date),
    DATEPART(quarter, shipped_date),
    DATEPART(month, shipped_date),
    DATEPART(day, shipped_date)
ORDER BY [year] DESC,
    [quarter],
    [month],
    [day];

```

بخشی از خروجی به شکل زیر است:


```

SELECT DATEPART(year, shipped_date) [year],
       DATEPART(quarter, shipped_date) [quarter],
       DATEPART(month, shipped_date) [month],
       DATEPART(day, shipped_date) [day],
       SUM(quantity * list_price) gross_sales
FROM sales_orders o
     INNER JOIN sales_order_items i ON i.order_id = o.order_id
WHERE shipped_date IS NOT NULL
GROUP BY DATEPART(year, shipped_date),

```

	year	quarter	month	day	gross_sales
1	2018	1	1	1	3259.96
2	2018	1	1	2	11963.92
3	2018	1	1	3	20810.85
4	2018	1	1	5	8819.93
5	2018	1	1	6	5126.94
6	2018	1	1	7	17117.84
7	2018	1	1	8	11869.93
8	2018	1	1	9	2909.94
9	2018	1	1	10	21789.92
10	2018	1	1	12	20839.92
11	2018	1	1	13	20919.92
12	2018	1	1	14	26655.85
13	2018	1	1	15	40799.80
14	2018	1	1	16	21097.95
15	2018	1	1	17	31089.89

در این مثال، ما از تابع DATEPART() برای استخراج سال، ربع سال، ماه و روز از مقادیر موجود در ستون shipped_date استفاده کردیم. در دستور GROUP BY، ما نرخ‌های رشد فروش (quantity * list_price) را با استفاده از این بخش‌های تاریخ جمع‌آوری کردیم.

توجه کنید که می‌توانید از تابع DATEPART() در دستورهای GROUP BY، HAVING، WHERE، SELECT و ORDER BY استفاده کنید.

تابع DAY در SQL Server

تابع DAY() یک مقدار integer باز می‌گرداند که روز یک ماه (از ۱ تا ۳۱) از یک تاریخ مشخص شده را نشان می‌دهد.

کد زیر syntax تابع DAY() را نشان می‌دهد:

DAY(input_date)

تابع DAY() یک آرگومان می‌گیرد که این آرگومان می‌تواند یک تاریخ یا یک عبارت باشد که به صورت مقدار DATETIME2، DATETIME، SMALLDATETIME، DATE، TIME یا DATETIMEOFFSET استفاده شود.

تابع DAY() مقدار مشابه با تابع DATEPART() که در زیر مشاهده می‌کنید باز می‌گرداند:

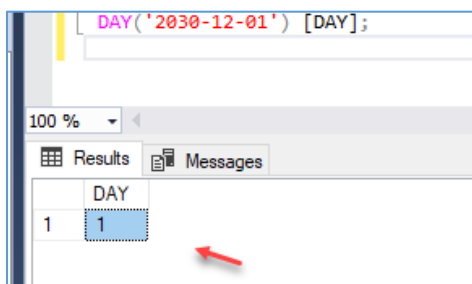
DATEPART(day, input_date)

الف) مثال استفاده از تابع DAY() با یک مقدار تاریخ

این مثال از تابع DAY() برای استخراج روز از تاریخ '۲۰۳۰-۱۲-۰۱' استفاده می‌کند:

```
SELECT
DAY('2030-12-01') [DAY];
```

خروجی به این شکل است:

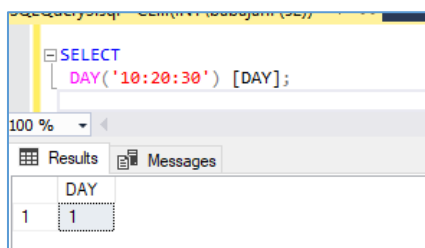


ب) مثال استفاده از تابع DAY() با یک مقدار تاریخ که فقط دارای بخش زمان است

تابع DAY() در صورتی که تاریخ ورودی شامل فقط بخش زمان باشد، مقدار ۱ را بازمی‌گرداند:

```
SELECT
DAY('10:20:30') [DAY];
```

کد زیر، خروجی را نشان می‌دهد:



پ) مثال استفاده از تابع DAY() با ستون‌های جدول

از جداول sales.orders و sales.order_items برای مثال استفاده می‌کنیم:

در این مثال از تابع DAY() برای استخراج داده روز از مقادیر موجود در ستون shipped_date استفاده شده است. این مثال با استفاده از تابع SUM() و دستور GROUP BY، نرخ‌های رشد فروش را بر اساس روز در فوریه ۲۰۱۷ بازمی‌گرداند:

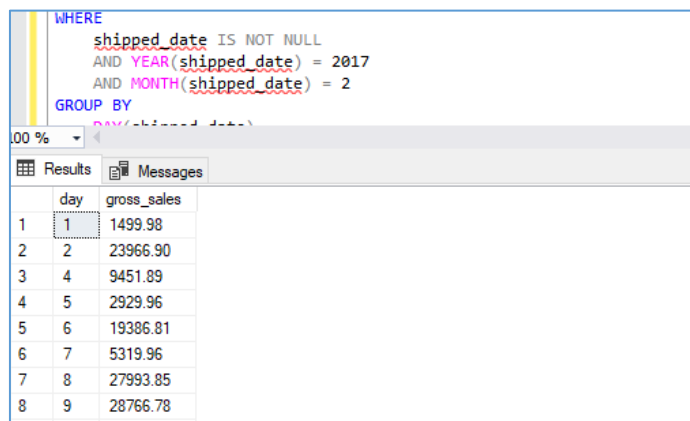
```
SELECT
DAY(shipped_date) [day],
SUM(list_price * quantity) gross_sales
```

```

FROM
    sales.orders o
    INNER JOIN sales.order_items i ON i.order_id = o.order_id
WHERE
    shipped_date IS NOT NULL
    AND YEAR(shipped_date) = 2017
    AND MONTH(shipped_date) = 2
GROUP BY
    DAY(shipped_date)
ORDER BY
    [day];

```

تصویر زیر، خروجی را نشان می‌دهد:



	day	gross_sales
1	1	1499.98
2	2	23966.90
3	4	9451.89
4	5	2929.96
5	6	19386.81
6	7	5319.96
7	8	27993.85
8	9	28766.78

تابع MONTH در SQL Server

تابع MONTH() یک مقدار integer باز می‌گرداند که نشان‌دهنده ماه از یک تاریخ مشخص شده است.

کد زیر syntax تابع MONTH() را نشان می‌دهد:

```
MONTH(input_date)
```

تابع MONTH() یک آرگومان می‌گیرد که می‌تواند یک مقدار تاریخ یا یک عبارت باشد که دارای مقدار DATETIME2، DATETIME، SMALLDATETIME، DATE، TIME یا DATETIMEOFFSET است.

تابع MONTH() مقدار مشابهی با تابع DATEPART() باز می‌گرداند:

```
DATEPART(month,input_date)
```

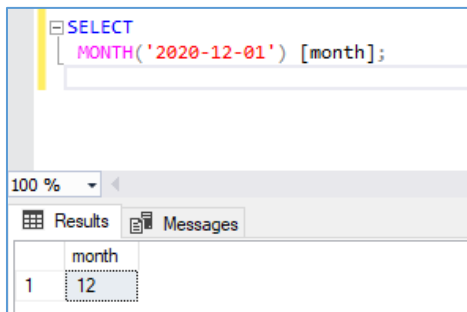
مثال‌های تابع MONTH() در SQL Server

الف) استفاده از تابع MONTH() با یک مقدار تاریخ

این مثال از تابع MONTH() برای استخراج ماه از تاریخ '۲۰۲۰-۱۲-۰۱' استفاده می‌کند:

```
SELECT
MONTH('2020-12-01') [month];
```

خروجی به این شکل است:

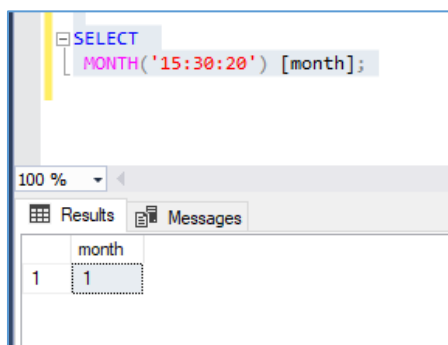


ب) استفاده از تابع MONTH() با یک مقدار تاریخ که فقط دارای داده زمان است

تابع MONTH() در صورتی که مقدار تاریخ شامل فقط بخش زمان باشد، مقدار ۱ را بازمی‌گرداند:

```
SELECT
MONTH('15:30:20') [month];
```

خروجی به این ترتیب است:



پ) استفاده از تابع MONTH() با ستون‌های جدول

طبق معمول از جداول Sales.orders و Sales.order_items برای مثال استفاده می‌کنیم.

این مثال از تابع MONTH() برای استخراج داده ماه از مقادیر موجود در ستون shipped_date استفاده می‌کند. نتیجه

این مثال، نرخ رشد فروش بر اساس ماه در سال ۲۰۱۸ است که با استفاده از تابع SUM() و دستور GROUP BY به

دست می‌آیند:

```
SELECT MONTH(shipped_date) [month],
SUM(list_price * quantity) gross_sales
FROM sales.orders o
INNER JOIN sales.order_items i ON i.order_id = o.order_id
WHERE shipped_date IS NOT NULL
AND YEAR(shipped_date) = 2017
GROUP BY MONTH(shipped_date)
ORDER BY [month];
```

تصویر زیر، خروجی را نشان می‌دهد:

	month	gross_sales
1	1	291487.89
2	2	294607.77
3	3	368210.07
4	4	266578.50
5	5	282580.74
6	6	420611.08
7	7	221318.92
8	8	326123.26
9	9	317060.85
10	10	311766.19
11	11	332052.61
12	12	293492.88

تابع YEAR در SQL Server

تابع YEAR() یک مقدار integer باز می‌گرداند که نشان‌دهنده سال از یک تاریخ مشخص شده است. کد زیر syntax تابع YEAR() را نشان می‌دهد:

YEAR(input_date)

تابع YEAR() یک آرگومان می‌گیرد که می‌تواند یک مقدار تاریخ یا یک عبارت باشد که دارای مقدار DATETIME2، DATETIME، SMALLDATETIME، DATE، TIME یا DATETIMEOFFSET است.

تابع YEAR() مقدار مشابهی با تابع DATEPART() باز می‌گرداند:

DATEPART(year,input_date)

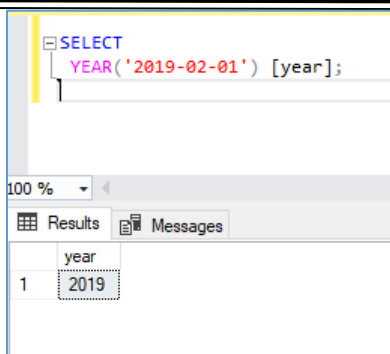
مثال‌های تابع YEAR() در SQL Server

الف) استفاده از تابع YEAR() با یک مقدار تاریخ

این مثال از تابع YEAR() برای استخراج سال از تاریخ '۲۰۱۹-۰۲-۰۱' استفاده می‌کند:

```
SELECT
  YEAR('2019-02-01') [year];
```

خروجی به این شکل است:

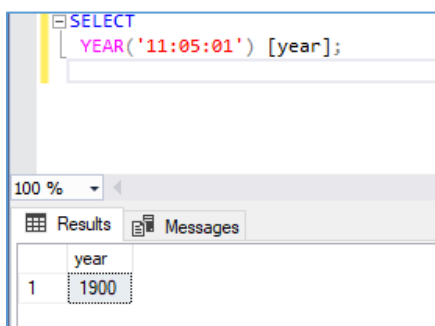


ب) استفاده از تابع YEAR() با یک مقدار تاریخ که فقط دارای بخش زمان است

اگر مقدار داده ورودی فقط دارای داده زمان باشد، تابع YEAR() مقدار ۱۹۰۰ را بازمی‌گرداند:

```
SELECT
YEAR('11:05:01') [year];
```

خروجی به این ترتیب است:



پ) استفاده از تابع YEAR() با ستون‌های جدول

باز هم از جداول Sales.orders و Sales.order_items برای مثال استفاده می‌کنیم.

این مثال از تابع YEAR() برای استخراج داده سال از مقادیر موجود در ستون shipped_date استفاده می‌کند. نتیجه این مثال، نرخ رشد فروش بر اساس سال است که با استفاده از تابع SUM() و دستور GROUP BY به دست می‌آید:

```
SELECT YEAR(shipped_date) [year],
SUM(list_price * quantity) gross_sales
FROM sales.orders o
INNER JOIN sales.order_items i ON i.order_id = o.order_id
WHERE shipped_date IS NOT NULL
GROUP BY YEAR(shipped_date)
order by [year];
```

کد زیر، خروجی را نشان می‌دهد:

```

SELECT YEAR(shipped_date) [year],
       SUM(list_price * quantity) gross_sales
FROM sales_orders o
INNER JOIN sales_order_items i ON i.order_id = o.order_id
WHERE shipped_date IS NOT NULL
GROUP BY YEAR(shipped_date)
order by [year];

```

year	gross_sales	
1	2016	2649649.97
2	2017	3725890.76
3	2018	1062469.33

تابع DATEDIFF در SQL Server

برای محاسبه تفاوت بین دو تاریخ از لحاظ سال، ماه، هفته و غیره از تابع DATEDIFF() استفاده می‌کنیم:

DATEDIFF(date_part , start_date , end_date)

تابع DATEDIFF() سه آرگومان می‌پذیرد: date_part, start_date و end_date.

○ Date_part بخشی از تاریخ (مثلاً یک سال، یک ربع سال، یک ماه، یک هفته و غیره) است که می‌خواهید اختلاف بر اساس آن بین start_date و end_date محاسبه شود. می‌توانید بخش‌های تاریخ معتبر و قابل قبول را در جدول زیر مشاهده کنید.

○ Start_date و end_date تاریخ‌هایی هستند که باید با هم مقایسه شوند. آن‌ها باید دارای مقادیری از نوع DATE, DATETIME, DATETIMEOFFSET, DATETIME2, SMALLDATETIME یا TIME باشند.

جدول زیر لیستی از مقادیر معتبر و قابل قبول برای date_part را نشان می‌دهد:

Date_part	اختصارات
سال	yy, yyyy
ربع سال	qq, q
ماه	mm, m
روز از سال	dy, y
روز	dd, d
هفته	wk, ww
ساعت	hh

mi, n	دقیقه
ss, s	ثانیه
ms	میلی ثانیه
mcs	میکروثانیه
ns	نانو ثانیه

تابع DATEDIFF() یک مقدار integer را بازمی‌گرداند که این مقدار نشان‌دهنده تفاوت بین start_date و end_date است که واحد مشخص‌کننده نوع این تفاوت توسط Date_part مشخص می‌شود.

اگر نتیجه به‌دست‌آمده خارج از بازه اعداد integer باشد (یعنی خارج از بازه ۲,۱۴۷,۴۸۳,۶۴۸- الی ۲,۱۴۷,۴۸۳,۶۴۷+), آنگاه تابع DATEDIFF() ارور می‌دهد. در چنین وضعیتی، باید در عوض از تابع DATEDIFF_BIG() استفاده کنید.

مثال‌های تابع DATEDIFF() در SQL Server

الف) استفاده از تابع DATEDIFF() برای مقایسه تفاوت‌های بین دو مقدار تاریخ

این مثال از تابع DATEDIFF() برای مقایسه تفاوت بین دو تاریخ که دارای بخش‌های مختلفی از تاریخ هستند، استفاده می‌کند:

DECLARE

```
@start_dt DATETIME2= '2019-12-31 23:59:59.9999999',
@end_dt DATETIME2= '2020-01-01 00:00:00.0000000';
```

SELECT

```
DATEDIFF(year, @start_dt, @end_dt) diff_in_year,
DATEDIFF(quarter, @start_dt, @end_dt) diff_in_quarter,
DATEDIFF(month, @start_dt, @end_dt) diff_in_month,
DATEDIFF(dayofyear, @start_dt, @end_dt) diff_in_dayofyear,
DATEDIFF(day, @start_dt, @end_dt) diff_in_day,
DATEDIFF(week, @start_dt, @end_dt) diff_in_week,
DATEDIFF(hour, @start_dt, @end_dt) diff_in_hour,
DATEDIFF(minute, @start_dt, @end_dt) diff_in_minute,
DATEDIFF(second, @start_dt, @end_dt) diff_in_second,
DATEDIFF(millisecond, @start_dt, @end_dt) diff_in_millisecond;
```

خروجی به این شکل است:

	diff_in_year	diff_in_quarter	diff_in_month	diff_in_dayofyear	diff_in_day	diff_in_week	diff_in_hour	diff_in_minute	diff_in_second	diff_in_millisecond
1	1	1	1	1	1	0	1	1	1	1

ب) مثال استفاده از تابع DATEDIFF() همراه با ستون جدول

مثال زیر از تابع DATEDIFF() برای مقایسه تاریخ دریافت درخواستی با تاریخ ارسال (بهروز) استفاده کرده و نتیجه‌ای که باز می‌گرداند بیان می‌کند که آیا سفارش به‌موقع تحویل داده شده یا خیر.

```
SELECT
    order_id,
    required_date,
    shipped_date,
    CASE
        WHEN DATEDIFF(day, required_date, shipped_date) < 0
        THEN 'Late'
        ELSE 'OnTime'
    END shipment
FROM
    sales.orders
WHERE
    shipped_date IS NOT NULL
ORDER BY
    required_date;
```

خروجی به‌این ترتیب است:

	order_id	required_date	shipped_date	shipment
1	1	2016-01-03	2016-01-03	OnTime
2	2	2016-01-04	2016-01-03	Late
3	4	2016-01-04	2016-01-05	OnTime
4	3	2016-01-05	2016-01-03	Late
5	8	2016-01-05	2016-01-05	OnTime
6	5	2016-01-06	2016-01-06	OnTime
7	10	2016-01-06	2016-01-06	OnTime
8	6	2016-01-07	2016-01-05	Late
9	7	2016-01-07	2016-01-05	Late
10	9	2016-01-08	2016-01-08	OnTime
11	11	2016-01-08	2016-01-07	Late
12	12	2016-01-08	2016-01-09	OnTime
13	15	2016-01-10	2016-01-12	OnTime
14	13	2016-01-11	2016-01-11	OnTime
15	14	2016-01-11	2016-01-12	OnTime
16	17	2016-01-14	2016-01-14	OnTime
17	16	2016-01-15	2016-01-15	OnTime
18	20	2016-01-16	2016-01-17	OnTime

تابع DATEADD در SQL Server

تابع DATEADD() یک عدد به یک بخش خاص از تاریخ یک مقدار تاریخ ورودی اضافه می‌کند و مقدار اصلاح شده را باز می‌گرداند.

کد زیر syntax تابع DATEADD() را نشان می‌دهد:

DATEADD (date_part , value , input_date)

تابع DATEADD() سه آرگومان می‌پذیرد:

○ Date_part بخشی از تاریخ است که تابع DATEADD() به آن value اضافه می‌کند. اگر value یک مقدار اعشاری یا float باشد، آنگاه تابع DATEADD() بخش اعشاری را نادیده می‌گیرد. در چنین وضعیتی، تابع عدد را گرد نمی‌کند.

○ Input_date یک تاریخ ورودی یا یک عبارت است که می‌تواند یک مقدار با نوع SMALLDATETIME، DATETIME2، DATETIMEOFFSET، DATETIME، DATE یا TIME باشد.

جدول زیر مقادیر معتبر و قابل قبول date_part را نمایش می‌دهد:

اختصارات	Date_part
yy, yyyy	سال
qq, q	ربع سال
mm, m	ماه
dy, y	روز از سال
dd, d	روز
wk, ww	هفته
hh	ساعت
mi, n	دقیقه
ss, s	ثانیه
ms	میلی ثانیه
mcs	میکرو ثانیه
ns	نانو ثانیه

تابع DATEADD() بعد از اضافه کردن value به date_part، یک مقدار تاریخ جدید را بازمی‌گرداند.

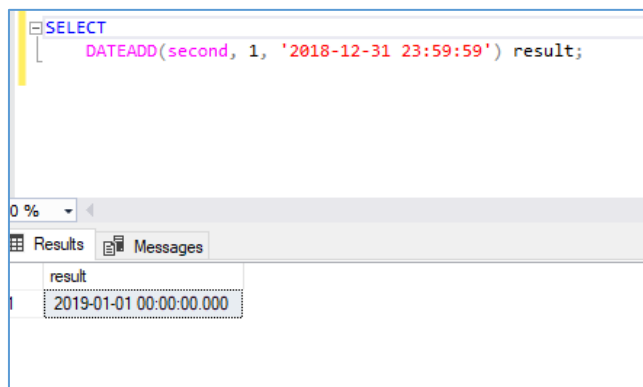
مثال‌های تابع DATEADD() در SQL Server

اضافه کردن ۱ ثانیه به ۲۳:۵۹:۵۹ -۳۱-۱۲-۲۰۱۸

SELECT

```
DATEADD(second, 1, '2018-12-31 23:59:59') result;
```

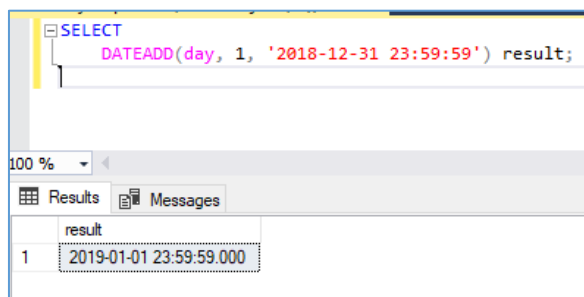
خروجی به این شکل است:



اضافه کردن ۱ روز به ۲۰۱۸-۱۲-۳۱ ۰۰:۰۰:۰۰ s

```
SELECT
DATEADD(day, 1, '2018-12-31 23:59:59') result;
```

خروجی:



این مثال از تابع DATEADD() برای محاسبه تاریخ ارسال تخمینی بر اساس تاریخ سفارش، استفاده می‌کند:

```
SELECT
    order_id,
    customer_id,
    order_date,
    DATEADD(day, 2, order_date) estimated_shipped_date
FROM
    sales.orders
WHERE
    shipped_date IS NULL
ORDER BY
    estimated_shipped_date DESC;
```

تصویر زیر، بخشی از خروجی را نمایش می‌دهد:

```

sales_orders
WHERE
  shipped_date IS NULL
ORDER BY
  estimated_shipped_date DESC;

```

	order_id	customer_id	order_date	estimated_shipped_date
1	1615	136	2018-12-28	2018-12-30
2	1614	135	2018-11-28	2018-11-30
3	1613	1	2018-11-18	2018-11-20
4	1612	3	2018-10-21	2018-10-23
5	1611	6	2018-09-06	2018-09-08
6	1610	15	2018-08-25	2018-08-27
7	1609	10	2018-08-23	2018-08-25
8	1608	53	2018-07-12	2018-07-14
9	1607	33	2018-07-11	2018-07-13
10	1606	119	2018-07-10	2018-07-12
11	1605	123	2018-07-01	2018-07-03
12	1604	7	2018-06-17	2018-06-19
13	1602	55	2018-04-30	2018-05-02

مثال‌های کار با ماه

اگر یک رقم ماه به یک تاریخ اضافه کنید و مقدار روز تاریخ به دست آمده وجود نداشته باشد، آنگاه تابع DATEADD() آخرین روز ماه بازگشتی را بازمی‌گرداند.

به مثال زیر توجه کنید:

```

SELECT
  DATEADD(month, 4, '2019-05-31') AS result;

```

در این مثال، مقدار ماه از تاریخ بازگشتی، ماه سپتامبر است. به هر حال، روز ۳۱ در ماه سپتامبر وجود ندارد، از این رو، تابع DATEADD() آخرین روز ماه سپتامبر (یعنی ۳۰) را به عنوان مقدار روز برای تاریخ بازگشتی نمایش می‌دهد:

```

SELECT
  DATEADD(month, 4, '2019-05-31') AS result;

```

	result
1	2019-09-30 00:00:00.000

توجه کنید که query زیر نیز همان نتیجه بالا را بازمی‌گرداند:

```

SELECT
  DATEADD(month,4, '2019-05-30') AS result;

```

خروجی:

The screenshot shows a SQL query window with the following text:

```
SELECT
DATEADD(month,4,'2019-05-30') AS result;
```

Below the query, the 'Results' tab is active, displaying a table with one row:

	result
1	2019-09-30 00:00:00.000

تابع EOMONTH در SQL Server

تابع EOMONTH() آخرین روز ماه یک تاریخ خاص را همراه با یک offset اختیاری، بازمی‌گرداند.

کد زیر syntax تابع EOMONTH() را نشان می‌دهد:

```
EOMONTH(start_date [, offset]);
```

تابع EOMONTH() دو آرگومان می‌پذیرد:

- Start_date یک عبارت تاریخ است که مقدار داخل آن نیز یک تاریخ است. تابع EOMONTH() آخرین روز ماه را برای این تاریخ بازمی‌گرداند.
- Offset یک integer است که عدد ماه‌هایی که باید به Start_date اضافه شوند را مشخص می‌کند.

اگر نتیجه اضافه شدن offset به start_date یک مقدار غیرمعتبر یا غیرقابل قبول باشد، آنگاه تابع EOMONTH() ارور می‌دهد.

مثال‌های EOMONTH() در SQL Server

الف) استفاده از تابع EOMONTH() برای یک تاریخ

این مثال از تابع EOMONTH() برای بازگرداندن آخرین روز ماه برای تاریخ ۲۰۱۹-۰۲-۱۵ استفاده می‌کند:

```
SELECT
DATEADD(month,4,'2019-05-30') AS result;
```

خروجی به این شکل است:

The screenshot shows a SQL query: `SELECT DATEADD(month,4,'2019-05-30') AS result;`. The results pane shows a single row with the value `2019-09-30 00:00:00.000`.

	result
1	2019-09-30 00:00:00.000

اگر تاریخ یک سال کبیسه را به تابع EOMONTH() بدهید، باز هم مقدار صحیح را بازمی‌گرداند:

```
SELECT
  EOMONTH('2020-02-09') end_of_month_feb2020;
```

خروجی به این صورت است:

The screenshot shows a SQL query: `SELECT EOMONTH('2020-02-09') end_of_month_feb2020;`. The results pane shows a single row with the value `2020-02-29`.

	end_of_month_feb2020
1	2020-02-29

همان‌طور که به‌وضوح در خروجی مشاهده می‌کنید، آخرین روز ماه فوریه ۲۰۱۹ روز ۲۹ام است، نه ۲۸ام.

ب) استفاده از تابع EOMONTH() برای به‌دست‌آوردن تعداد روزها در یک ماه خاص برای به‌دست‌آوردن تعداد روزهای یک ماه خاص، مراحل زیر را انجام دهید: ابتدا، از تابع EOMONTH() برای به‌دست‌آوردن آخرین روز ماه استفاده کنید. سپس، آخرین روز ماه را به تابع DAY() بفرستید.

این مثال تعداد روزهای ماه فوریه سال ۲۰۱۸ را بازمی‌گرداند:

```
SELECT
  DAY(EOMONTH('2020-02-09')) days;
```

خروجی به این شکل است:

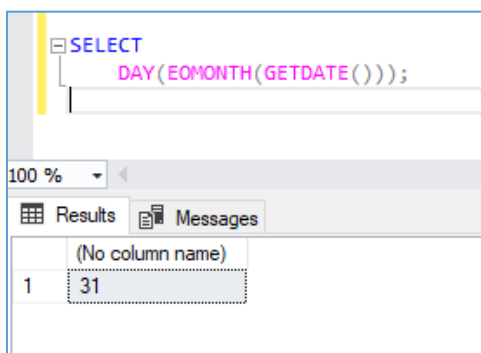
The screenshot shows a SQL query: `SELECT DAY(EOMONTH('2020-02-09')) days;`. The results pane shows a single row with the value `29`.

	days
1	29

برای به‌دست‌آوردن تعداد روزها در ماه جاری، می‌توانید از کد زیر استفاده کنید:

```
SELECT
```

```
DAY(EOMONTH(GETDATE()));
```

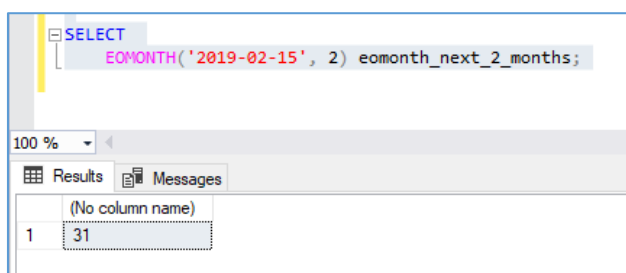


پ) مثال استفاده از تابع EOMONTH() با یک offset

مثال زیر از تابع EOMONTH() همراه با یک offset دو ماهه استفاده می‌کند:

```
SELECT
    EOMONTH('2019-02-15', 2) eomonth_next_2_months;
```

خروجی به شکل زیر است:



تابع SWITCHOFFSET در SQL Server

تابع SWITCHOFFSET() یک DATETIMEOFFSET که از offset منطقه زمانی ذخیره شده به یک offset منطقه زمانی جدید تغییر یافته است را بازمی‌گرداند.

کد زیر syntax تابع SWITCHOFFSET() را نشان می‌دهد:

```
SWITCHOFFSET(expression, time_zone)
```

تابع SWITCHOFFSET() دو آرگومان می‌پذیرد:

- Expression یک عبارت است که می‌تواند یک مقدار DATETIMEOFFSET باشد.
- Time_zone می‌توانید یک رشته کاراکتر به فرمت `{+|-}TZH:TZM` یا یک integer علامت‌دار از دقیقه‌ها باش. برای مثال، time_zone می‌تواند به شکل `+۰۸:۰۰`، `+۰۷:۰۰` یا `-۱۲۰` باشد.

تابع SWITCHOFFSET() یک تابع بسیار مفید و کارآمد برای به‌روزرسانی مقادیر در یک ستون DATETIMEOFFSET است.

مثال تابع SWITCHOFFSET() در SQL Server

ابتدا، یک جدول جدید ایجاد می‌کنیم که دارای ستون DATETIMEOFFSET باشد:

```
CREATE TABLE dbo.switchoffset_demo(
    dtz DATETIMEOFFSET
);
```

دوم، یک مقدار DATETIMEOFFSET جدید به جدول اضافه می‌کنیم:

```
INSERT INTO dbo.switchoffset_demo
VALUES('2019-03-06 9:20:00 +07:00');
```

سوم، برای به‌دست‌آوردن مقدار جدول dbo.switchoffset_demo یک query می‌نویسیم:

```
SELECT
    dtz
FROM
    dbo.switchoffset_demo;
```

خروجی به این شکل است:

dtz
2019-03-06 09:20:00.0000000 +07:00

چهارم، از تابع SWITCHOFFSET() برای تغییر منطقه زمانی به ۰۸:۰۰- استفاده می‌کنیم:

```
SELECT
    SWITCHOFFSET(dtz, '-08:00') result
FROM
    dbo.switchoffset_demo;
```

خروجی به‌این‌ترتیب می‌شود:

result
2019-03-05 18:20:00.0000000 -08:00

تابع TODATETIMEOFFSET در SQL Server

تابع TODATETIMEOFFSET() یک مقدار DATETIME2 را به یک مقدار DATETIMEOFFSET ترجمه می‌کند. کد زیر syntax تابع TODATETIMEOFFSET() را نشان می‌دهد:

TODATETIMEOFFSET(expression,time_zone)

تابع TODATETIMEOFFSET() دو آرگومان می‌پذیرد:

- Expression یک عبارت است که می‌تواند به مقدار DATETIME2 باشد.
- Time_zone می‌توانید یک integer یا string باشد. اگر از یک integer استفاده کنید، آنگاه time_zone یک offset منطقه زمانی به دقیقه می‌باشد. در غیر این صورت، یک ساعت یا دقیقه به فرم '+TZH:THM' یا '-TZH:THM' است که در آن TZH ساعت و THM دقیقه است. بازه ساعت از ۱۴- تا ۱۴+ است.

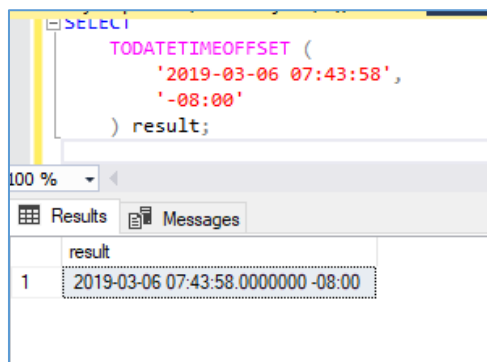
مثال‌های تابع TODATETIMEOFFSET() در SQL Server

الف) مثال استفاده از تابع TODATETIMEOFFSET() برای تغییر offset محدوده زمانی یک تاریخ و زمان این مثال، offset منطقه یک تاریخ و زمان را به منطقه ۰۸:۰۰- تغییر می‌دهد.

SELECT

```
TODATETIMEOFFSET (
    '2019-03-06 07:43:58',
    '-08:00'
) result;
```

خروجی به این ترتیب است:

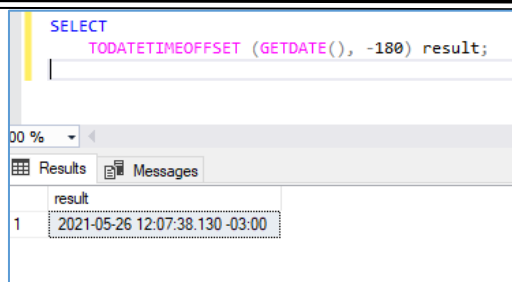


ب) مثال استفاده از تابع TODATETIMEOFFSET() برای تغییر offset محدوده زمانی به دقیقه این مثال، محدوده زمانی فعلی را به ۱۸۰- دقیقه تغییر می‌دهد:

SELECT

```
TODATETIMEOFFSET (GETDATE(), -180) result;
```

خروجی به این شکل است:

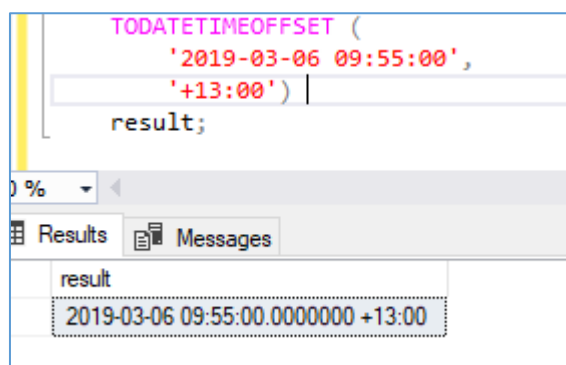


پ) مثال استفاده از تابع TODATETIMEOFFSET() برای اضافه کردن یک offset منطقه زمانی ساعت به تاریخ و زمان فعلی

مثال زیر یک offset منطقه زمانی ۱۲ ساعته را به تاریخ و زمان '۲۰۱۹-۰۳-۰۶ ۰۹:۵۵:۰۰' اضافه می کند:

```
SELECT
  TODATETIMEOFFSET (
    '2019-03-06 09:55:00',
    '+13:00')
  result;
```

خروجی به این ترتیب است:



تابع DATEFROMPARTS در SQL Server

تابع DATEFROMPARTS() یک مقدار DATE بازمی گرداند که متناظر با مقادیر سال، ماه و روز است. کد زیر، syntax تابع DATEFROMPARTS() را نمایش می دهد:

DATEFROMPARTS(year, month, day)

تابع DATEFROMPARTS() سه آرگومان می پذیرد:

- Year یک عبارت integer است که از نوع سال است.
- Month یک عبارت integer است که مقدار آن یک ماه از بازه ۱ تا ۱۲ است.
- Day یک عبارت integer است که یک روز را از بازه ۱ تا ۳۱ مشخص می کند.

تابع DATEFROMPARTS() یک مقدار DATE بازمی‌گرداند. اگر هرکدام از آرگومان‌ها NULL باشند، تابع مقدار NULL بازمی‌گرداند.

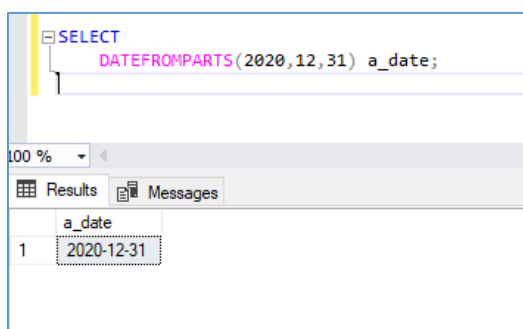
مثال‌های تابع DATEFROMPARTS() در SQL Server

الف) مثال استفاده از تابع DATEFROMPARTS() برای بازگرداندن یک تاریخ

این مثال از تابع DATEFROMPARTS() برای ساخت یک تاریخ متشکل از مقادیر سال، ماه و روز استفاده می‌کند:

```
SELECT
    DATEFROMPARTS(2020,12,31) a_date;
```

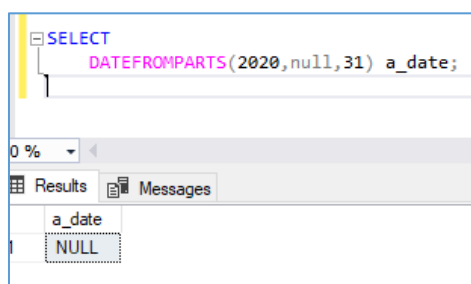
خروجی به این شکل است:



ب) مثال استفاده از تابع DATEFROMPARTS() همراه با مقدار NULL
مثال زیر مقدار NULL بازمی‌گرداند، چون آرگومان ماه برابر با NULL است:

```
SELECT
    DATEFROMPARTS(2020,null,31) a_date;
```

خروجی به این صورت است:



پ) مثال استفاده از تابع DATEFROMPARTS() همراه با آرگومان‌های خارج از بازه

این مثال آرگومان‌های غیرمعتبر و غیرقابل قبول را به تابع DATEFROMPARTS() ارسال می‌کند که در نتیجه سیستم ارور می‌دهد:

```
SELECT
    DATEFROMPARTS(2020,20,-1) a_date;
```

خط زیر، پیغام ارور را نشان می‌دهد:

```

SELECT
DATEFROMPARTS(2020,20,-1) a_date;

00 %
Results Messages
Msg 289, Level 16, State 1, Line 2
Cannot construct data type date, some of the arguments have values which are not valid.

Completion time: 2021-05-26T12:31:30.5318047+04:30

```

به این شکل، چگونگی ساخت یک مقدار DATE از بخش‌های سال، ماه و روز با استفاده از تابع DATEFROMPARTS() را آموختید.

تابع DATETIME2FROMPARTS در SQL Server

تابع DATETIME2FROMPARTS() یک مقدار تاریخ بازمی‌گرداند که از مقادیر سال، ماه، روز، ساعت، دقیقه، ثانیه، اعشار و دقت است. در واقع این تابع یک مقدار DATETIME2 ایجاد می‌کند. کد زیر، syntax تابع DATETIME2FROMPARTS() را نشان می‌دهد:

DATETIME2FROMPARTS (year, month, day, hour, minute, seconds, fractions, precision)

تابع DATETIME2FROMPARTS() هشت آرگومان می‌پذیرد:

- Year یک عبارت integer است که از نوع سال است.
- Month یک عبارت integer است که از نوع ماه و در بازه ۱ تا ۱۲ است.
- Day یک عبارت integer است که یک روز از بازه ۱ تا ۳۱ را مشخص می‌کند.
- Hour یک عبارت integer است که ساعت را مشخص می‌کند.
- Minute یک عبارت integer است که دقیقه را مشخص می‌کند.
- Seconds یک عبارت integer است که ثانیه‌ها را مشخص می‌کند.
- Fractions یک عبارت integer است که اعداد قسمت اعشار را نمایش می‌دهد.
- Precision یک عبارت integer است که دقت مقدار DATETIME2 را مشخص می‌کند.

تابع DATETIME2FROMPARTS() یک مقدار از نوع DATETIME2 بازمی‌گرداند. اگر هر کدام از آرگومان‌های بالا NULL باشند، تابع مقدار NULL را بازمی‌گرداند.

اگر یک آرگومان دارای یک مقدار غیرمعتبر باشد، تابع DATETIME2FROMPARTS() خطا می‌دهد.

مثال‌های تابع DATETIME2FROMPARTS() در SQL Server

الف) مثال استفاده از تابع DATETIME2FROMPARTS() برای بازگرداندن یک DATETIME2

مثال زیر از تابع DATETIME2FROMPARTS() برای ساخت یک DATETIME2 متشکل از سال، ماه، روز، ساعت، دقیق، ثانیه، اعشار و دقت استفاده می‌کند:

SELECT

```
DATETIME2FROMPARTS ( 2020, 12, 31, 11, 59, 59, 0, 0 ) result;
```

خروجی به این ترتیب است:

The screenshot shows a SQL query window with the following text:

```
SELECT
  DATETIME2FROMPARTS ( 2020, 12, 31, 11, 59, 59, 0, 0 ) result;
```

Below the query, the 'Results' pane shows a single row with the following value:

result
2020-12-31 11:59:59

ب) مثال استفاده از تابع DATETIME2FROMPARTS() همراه با آرگومان‌های NULL

این مثال مقدار NULL را بازمی‌گرداند، چون آرگومان month برابر با NULL است:

SELECT

```
DATETIME2FROMPARTS(2020, NULL, 31, 11, 59, 59, 0, 0) result;
```

خروجی به این شکل است:

The screenshot shows a SQL query window with the following text:

```
SELECT
  DATETIME2FROMPARTS(2020, NULL, 31, 11, 59, 59, 0, 0) result;
```

Below the query, the 'Results' pane shows a single row with the following value:

result
NULL

ب) مثال استفاده از تابع DATETIME2FROMPARTS() با مقادیر غیر معتبر در آرگومان‌ها

مثال زیر آرگومان‌های غیر معتبر را به تابع DATETIME2FROMPARTS() ارسال می‌کند که در نتیجه سیستم ارور می‌دهد:

SELECT

```
DATETIME2FROMPARTS(2020, 13, 31, 11, 59, 59, 0, 0) result;
```

پیغام ارور به این صورت است:

```

SELECT
  DATETIME2FROMPARTS(2020, 13, 31, 11, 59, 59, 0, 0) result;

```

Results Messages

Msg 289, Level 16, State 5, Line 1
Cannot construct data type datetime2, some of the arguments have values which are not valid.

Completion time: 2021-05-26T12:36:05.6669066+04:30

به این ترتیب، چگونگی استفاده از تابع DATETIME2FROMPARTS() برای ساخت یک مقدار DATETIME2 از اجزای تشکیل دهنده آن یعنی سال، ماه، روز، ساعت، دقیقه، ثانیه، اعشار و دقت را آموختید.

۳-۱۲-۳-۳ توابع رشته‌ای

در جدول زیر توابع رشته‌ای در SQL Server را مشاهده می‌کنید که روی یک رشته ورودی عمل می‌کنند و یک مقدار رشته یا عددی بازمی‌گردانند:

تایع	توضیح
ASCII	مقدار کد ASCII یک کاراکتر را بازمی‌گرداند.
CHAR	یک مقدار ASCII را به یک کاراکتر تبدیل می‌کند.
CHATINDEX	درون یک رشته که از یک موقعیت خاص شروع می‌شود به دنبال یک زیررشته می‌گردد و موقعیت زیررشته را بازمی‌گرداند.
CONCAT	دو یا تعداد بیشتری رشته را به یک رشته تبدیل می‌کند.
CONCAT_WS	چندین رشته را با یک علامت مجزا کننده، به یک رشته واحد تبدیل می‌کند.
DEFERENCE	مقادیر (SOUNDEX) از دو رشته را با هم مقایسه می‌کند.
FORMAT	یک مقدار فرمت شده با یک فرمت خاص و رفتار (اختیاری) را بازمی‌گرداند.
LEFT	تعداد مشخص شده‌ای کاراکتر را از یک رشته کاراکتر که از سمت چپ آغاز می‌شوند، استخراج می‌کند.
LEN	تعدادی کاراکتر را از یک رشته کاراکتر بازمی‌گرداند.
LOWER	کاراکترهای یک رشته را به حروف کوچک تبدیل می‌کند.
LTRIM	یک رشته جدید از یک رشته مشخص شده را پس از حذف جاهای خالی بازمی‌گرداند.
NCHAR	کاراکتر Unicode با کد integer مشخص شده را بازمی‌گرداند، همان‌طور که توسط استاندارد UNICODE تعریف شده است.
PATINDEX	موقعیت آغازین اولین وقوع الگو در یک رشته را بازمی‌گرداند.

یک رشته UNICODE همراه با فاصله دهنده را بازمی‌گرداند تا رشته ورودی را تبدیل به یک شناساگر دارای فاصله معتبر تبدیل کند.	QUOTENAME
تمام وقوع‌های یک زیررشته را درون یک رشته، با یک زیررشته دیگر جایگزین می‌کند.	REPLACE
یک رشته که به تعداد دفعات خاصی تکرار شده است را بازمی‌گرداند.	REPLICATE
ترتیب برعکس یک رشته کاراکتر را بازمی‌گرداند.	REVERSE
تعداد خاصی از کاراکترها را از یک رشته کاراکتر که از سمت راست شروع می‌شود، بازمی‌گرداند.	RIGHT
یک رشته جدید از یک رشته مشخص شده را پس از حذف تمام جاهای خالی، بازمی‌گرداند.	RTRIM
یک کد (یا SOUNDEX) چهار کاراکتری از یک رشته را بر اساس اینکه چگونه گفته شده است، بازمی‌گرداند.	SOUNDEX
یک رشته، از فضاها یا فاصله‌های تکرارشونده را بازمی‌گرداند.	SPACE
داده‌های کاراکتری که از داده‌های عددی تبدیل شده است را بازمی‌گرداند.	STR
ردیف‌های رشته‌های دارای یک مجزا کننده خاص را به یک رشته جدید متصل می‌کند.	STRING_AGG
کاراکترهای خاصی از یک رشته را رها می‌کند و یک رشته جدید از کاراکترهای رها شده بازمی‌گرداند.	STRING_ESCAPE
یک تابع با مقدار جدولی که یک رشته را بر اساس یک مجزا کننده مشخص شده، به ردیف‌های زیررشته مجزا می‌کند.	STRING_SPLIT
بخشی از یک رشته را حذف می‌کند و سپس یک زیررشته دیگر را با شروع از یک موقعیت خاص، به رشته اضافه می‌کند.	STUFF
یک زیررشته از درون یک رشته که از یک موقعیت خاص آغاز می‌شود و دارای طول خاصی است، استخراج می‌کند.	SUBSTRING
چندین تک کاراکتر را در یک عملیات جایگزین کرده و تک به تک ترجمه می‌کند.	TRANSLATE
یک رشته جدید از یک رشته مشخص شده را پس از حذف تمام جاهای خالی ابتدا و انتها و بین حروف، بازمی‌گرداند.	TRIM
مقدار integer از یک کاراکتر را بازمی‌گرداند (همان‌طور که توسط استاندارد Unicode تعریف شده است).	UNICODE
یک رشته را به حروف بزرگ تبدیل می‌کند.	UPPER

همان‌طور که در جدول بالا مشاهده کردید تعداد توابع زیاد هست و سعی می‌کنیم در این قسمت بعضی از آنها را بررسی کنیم.

تابع ASCII در SQL Server

تابع ASCII() یک عبارت کاراکتر می‌پذیرد و مقدار کد ASCII از اولین کاراکتر سمت چپ در یک عبارت کاراکتری را بازمی‌گرداند.

کد زیر، syntax تابع ASCII() را نشان می‌دهد:

ASCII (input_string)

Input_string می‌تواند یک کاراکتر حقیقی، یک عبارت رشته‌ای کاراکتری یا یک ستون باشد. اگر input_string دارای بیش از یک کاراکتر باشد، آنگاه تابع مقدار کد ASCII از اولین کاراکتر سمت چپ آن را بازمی‌گرداند.

همان‌طور که احتمالاً می‌دانید، ASCII مخفف کد استاندارد آمریکایی برای تبادل اطلاعات (American Standard Code for Information Interchange) است. ASCII به‌عنوان یک استاندارد رمزگذاری کاراکتر برای رایانه‌های مدرن به کار می‌رود.

مثال‌های تابع ASCII() در SQL Server

مثال زیر مقادیر کد ASCII از کاراکتر A تا Z را بازمی‌گرداند:

```
SELECT
  ASCII('AB') A,
  ASCII('Z') Z;
```

خروجی به این شکل است:

The screenshot shows a SQL Server query window with the following SQL code:

```
SELECT
  ASCII('AB') A,
  ASCII('Z') Z;
```

The results pane displays a table with two columns, A and Z. The first row shows the value 65 for column A and 90 for column Z.

	A	Z
1	65	90

قطعه کد زیر از تابع ASCII() برای تولید ۲۶ حرف الفبا استفاده می‌کند:

```
WITH cte AS(
    SELECT
        CHAR(ASCII('A')) [char],
        1 [count]
    UNION ALL
    SELECT
        CHAR(ASCII('A') + cte.count) [char],
        cte.count + 1 [count]
    FROM
        cte
)
SELECT
    TOP(26) cte.char
FROM
    cte;
```

خروجی به این ترتیب است:

	char
1	A
2	B
3	C
4	D
5	E
6	F
7	G
8	H
9	I
10	J
11	K
12	L
13	M
14	N
15	O
16	P

تابع CHAR در SQL Server

تابع CHAR() یک مقدار کد ASCII را به یک مقدار کاراکتر تبدیل می‌کند. کد زیر، syntax تابع CHAR() را نشان می‌دهد:

CHAR (int_exp)

در این syntax، کد inte_expr یک عبارت integer است که مقدار آن یک integer است که بازه آن بین ۰ تا ۲۵۵ است. اگر عبارت integer دارای مقداری باشد که خارج از این بازه است، آنگاه تابع CHAR() مقدار NULL را بازمی‌گرداند.

تابع CHAR() کاراکتری را بازمی‌گرداند که نوع داده آن CHAR(۱) است.

توجه کنید که برای تبدیل یک کاراکتر به یک مقدار ASCII، از تابع ASCII() استفاده می‌کنیم.

مثال تابع CHAR() در SQL Server

مثال زیر از تابع CHAR() برای به دست آوردن کاراکترهای اعداد ۶۵ و ۹۰ استفاده می کند:

SELECT

```
CHAR(65) char_65,
CHAR(90) char_90;
```

خروجی به این شکل است:

The screenshot shows a SQL query window with the following SQL code:

```
SELECT
  CHAR(65) char_65,
  CHAR(90) char_90;
```

The results pane shows a table with two columns: char_65 and char_90. The first row contains the values 'A' and 'Z'.

	char_65	char_90
1	A	Z

کد زیر مقدار NULL را بازمی گرداند، چون مقدار آرگومان خارج از بازه ۰ تا ۲۵۵ است.

SELECT

```
CHAR(1000) out_of_range;
```

کد بالا، خروجی زیر را نشان می دهد:

The screenshot shows a SQL query window with the following SQL code:

```
SELECT
  CHAR(1000) out_of_range;
```

The results pane shows a table with one column: out_of_range. The first row contains the value NULL.

	out_of_range
1	NULL

تابع CHARINDEX در SQL Server

تابع CHARINDEX() به جستجوی یک زیررشته درون یک رشته که از یک موقعیت خاص شروع می شود، می پردازد. این تابع موقعیت زیررشته یافته شده در رشته مورد جستجو را بازمی گرداند، یا اگر زیررشته اصلاً یافته نشود، مقدار صفر را بازمی گرداند.

موقعیت آغازین از ۱ شمرده می شود نه از ۰.

کد زیر، syntax تابع CHARINDEX() را نشان می دهد:

CHARINDEX(substring, string [, start_location])

در این syntax:

- Substring زیر رشته‌ای است که به دنبال آن هستیم. طول آن به ۸۰۰۰ کاراکتر محدود است.
- String می‌تواند یک رشته، عبارت یا ستون باشد. این رشته‌ای است که جستجو می‌شود.
- Start_location موقعیتی است که جستجو از آنجا آغاز می‌شود. Start_location می‌تواند یک integer یا یک big integer یا یک عبارت باشد که مقدار آن ممکن است هر نوع داده‌ای دیگری باشد.
- پارامتر start_location اختیاری است. اگر آن را مشخص نکنید یا مقدار آن صفر یا یک مقدار منفی باشد، آنگاه جستجو از ابتدای رشته آغاز می‌شود.

توجه کنید که تابع CAHRINDEX() بر اساس نوع قیاس، می‌تواند هم به صورت حساس به بزرگی یا کوچکی حروف و هم غیرحساس به این موضوع عمل کند.

مثال‌های تابع CAHRINDEX() در SQL Server

الف) استفاده از تابع CAHRINDEX() برای انجام یک جستجو

مثال زیر از تابع CAHRINDEX() برای انجام یک جستجوی ساده برای یافتن رشته 'SQL' درون رشته 'SQL Server' CHARINDEX استفاده می‌کند.

SELECT

```
CHARINDEX('SQL', 'SQL Server CHARINDEX') position;
```

خروجی به این شکل است:

The screenshot shows a SQL Server query window with the following SQL statement: `SELECT CHARINDEX('SQL', 'SQL Server CHARINDEX') position;`. Below the query, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, displaying a single row with the value '1' in the 'position' column.

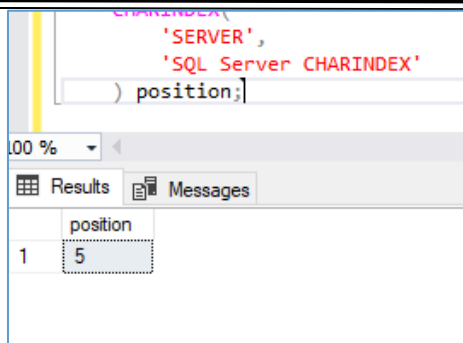
position
1

ب) استفاده از تابع CAHRINDEX() برای انجام یک جستجوی غیرحساس به بزرگی و کوچکی حروف کد زیر یک جستجوی غیرحساس به بزرگی و کوچکی حروف برای یافتن رشته 'SERVER' درون رشته 'SQL Server CHARINDEX' را نشان می‌دهد:

SELECT

```
CHARINDEX(
    'SERVER',
    'SQL Server CHARINDEX'
) position;
```

خروجی به این ترتیب است:

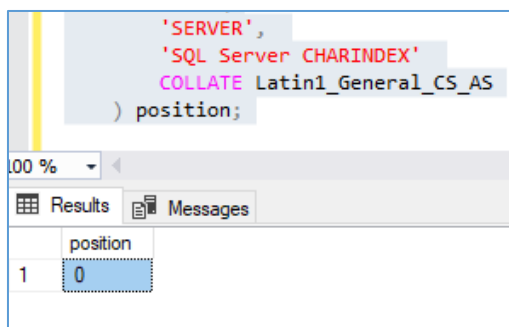


پ) استفاده از تابع CHARINDEX() برای انجام یک جستجوی حساس به بزرگی و کوچکی حروف

مثال زیر، یک جستجوی حساس به بزرگی و کوچکی حروف برای یافتن رشته 'SERVER' در رشته 'SQL Server CHARINDEX' را نشان می‌دهد.

```
SELECT
    CHARINDEX(
        'SERVER',
        'SQL Server CHARINDEX'
        COLLATE Latin1_General_CS_AS
    ) position;
```

خروجی به این شکل است:

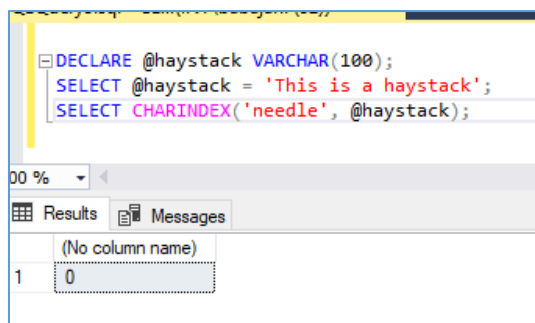


ت) استفاده از تابع CHARINDEX() برای جستجوی یک زیررشته ناموجود

مثال زیر یک جستجو برای یافتن زیررشته 'needle' در رشته 'This is a haystack' را نشان می‌دهد:

```
DECLARE @haystack VARCHAR(100);
SELECT @haystack = 'This is a haystack';
SELECT CHARINDEX('needle', @haystack);
```

خروجی به این شکل است:



ث) استفاده از تابع CHARINDEX() برای جستجو از یک موقعیت خاص

این مثال از پارامتر start_location برای آغاز جستجوی رشته 'is' از پنجمین و دهمین کاراکتر از رشته 'This is a my sister' استفاده می‌کند:

SELECT

```
CHARINDEX('is','This is a my sister',5) start_at_fifth,
CHARINDEX('is','This is a my sister',10) start_at_tenth;
```

خروجی به این ترتیب است:

The screenshot shows a SQL query window with the following SQL code:

```
SELECT
    CHARINDEX('is','This is a my sister',5) start_at_fifth,
    CHARINDEX('is','This is a my sister',10) start_at_tenth;
```

The results pane shows a table with two columns: start_at_fifth and start_at_tenth. The values are 6 and 15 respectively.

start_at_fifth	start_at_tenth
6	15

تابع CONCAT در SQL Server

برای تبدیل دو یا تعداد بیشتری رشته به یک رشته، از تابع CONCAT() با syntax زیر استفاده می‌کنیم:

```
CONCAT ( input_string1, input_string2 [, input_stringN ] );
```

تابع CONCAT() دو رشته ورودی با حداکثر ۲۵۵ کاراکتر را می‌گیرد و آن‌ها را به یک رشته تبدیل می‌کند. این تابع حداقل به دو رشته ورودی نیاز دارد. اگر یکی از رشته‌های ورودی را وارد نکنید، تابع CONCAT() ارور می‌دهد. اگر مقادیر رشته‌ای غیر کاراکتری وارد کنید، آنگاه تابع CONCAT() قبل از اینکه آن دو مقدار را با هم ادغام کند، به صورت ضمنی آن‌ها را به رشته تبدیل می‌کند.

تابع CONCAT() همچنین مقدار NULL را به یک رشته خالی با نوع VARCHAR(1) تبدیل می‌کند. توجه کنید که برای اضافه کردن یک مجزا کننده در طول فرایند ادغام، باید از تابع CONCAT_WS() استفاده کنید (که در قسمت بعدی همین جلسه آن را توضیح خواهیم داد).

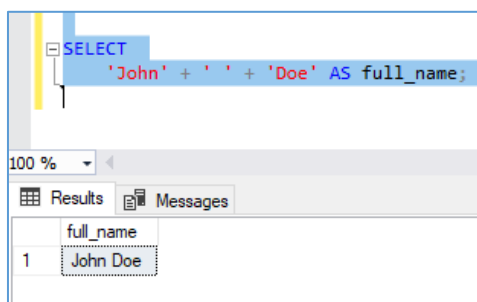
مثال‌های تابع CONCAT() در SQL Server

الف) استفاده از تابع CONCAT() برای چند رشته

مثال زیر از تابع CONCAT() برای ادغام سه رشته John، فاصله یا space و Doe استفاده می‌کند:

```
SELECT
    'John' + ' ' + 'Doe' AS full_name;
```

خروجی به این شکل است:



ب) استفاده از تابع CONCAT() برای ستونهای جدول

این مثال از جدول sales.customers از پایگاه داده نمونه Bikestores استفاده می کند:

جدول sales.customers

کد زیر از تابع CONCAT() برای ادغام مقادیر در ستونهای first_name و last_name از جدول Sales.customers استفاده می کند:

```
SELECT
    customer_id,
    first_name,
    last_name,
    CONCAT(first_name, ' ', last_name) full_name
FROM
    sales.customers
ORDER BY
    full_name;
```

تصویر زیر، خروجی را نشان می دهد:

	customer_id	first_name	last_name	full_name
1	1174	Aaron	Knapp	Aaron Knapp
2	338	Abbey	Pugh	Abbey Pugh
3	75	Abby	Gamble	Abby Gamble
4	1224	Abram	Copeland	Abram Copeland
5	673	Adam	Henderson	Adam Henderson
6	1085	Adam	Thornton	Adam Thornton
7	195	Addie	Hahn	Addie Hahn
8	1261	Adelaida	Hancock	Adelaida Hancock
9	22	Adelle	Larsen	Adelle Larsen
10	1023	Adena	Blake	Adena Blake
11	1412	Adrien	Hunter	Adrien Hunter
12	574	Adriene	Rivera	Adriene Rivera
13	1252	Adriene	Rollins	Adriene Rollins
14	527	Afton	Juarez	Afton Juarez
15	1353	Agatha	Daniels	Agatha Daniels
16	769	Agatha	Melton	Agatha Melton
17	771	Agnes	Sims	Agnes Sims
18	1181	Agustina	Lawrence	Agustina Lawrence
19	1322	Ai	Forbes	Ai Forbes
20	937	Aida	Koch	Aida Koch

پ) استفاده از تابع CONCAT() برای مقدار NULL

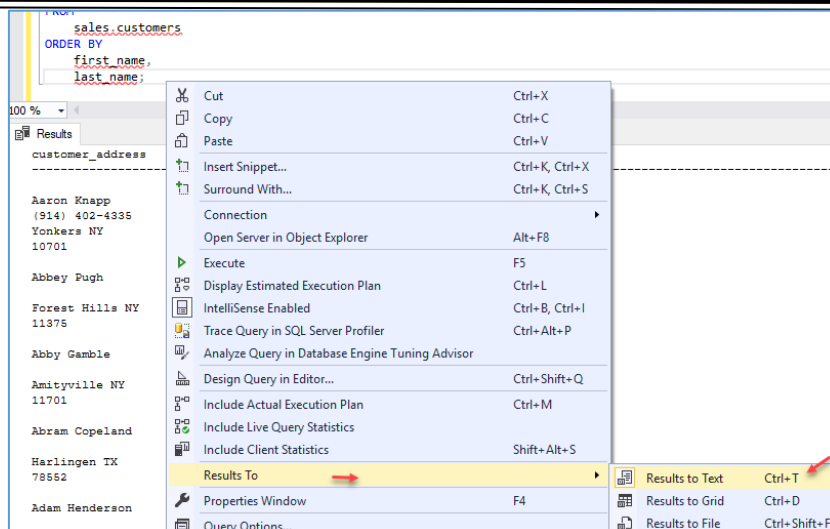
مثال زیر از تابع CONCAT() برای فرمت کردن آدرس‌ها برای مشتریان استفاده می‌کند:

```
SELECT
    CONCAT(
        CHAR(13),
        CONCAT(first_name, ' ', last_name),
        CHAR(13),
        phone,
        CHAR(13),
        CONCAT(city, ' ', state),
        CHAR(13),
        zip_code
    ) customer_address
FROM
    sales.customers
ORDER BY
    first_name,
    last_name;
```

بخشی از خروجی در فرمت متن به این شکل است:

	customer_address
1	Aaron Knapp (914) 402-4335 Yonkers NY 10701
2	Abbey Pugh Forest Hills NY 11375
3	Abby Gamble Amityville NY 11701
4	Abram Copeland Harlingen TX 78552
5	Adam Henderson Los Banos CA 93635
6	Adam Thomson Central Islip NY 11722
7	Addie Hahn Franklin Square NY 11010
8	Adelaida Hancock San Pablo CA 94806
9	Adelle Larsen East Northport NY 11731
10	Adena Blake Ballston Spa NY 12020
11	Adrien Hunter Rego Park NY 11374
12	Adriene Rivera Encino CA 91316
13	Adriene Rollins Plainview NY 11803

برای اینکه دقیق‌تر این دستور را درک کنید به‌مانند شکل زیر بر روی صفحه کد کلیک راست کنید و از قسمت Results To گزینه Results to Test را انتخاب کنید و بعد از آن با کلیک بر روی F5 خروجی را به‌صورت Test مشاهده خواهید کرد، البته می‌توانستید از کلیدهای ترکیبی Ctrl + T و Ctrl + D استفاده کنید.



همان‌طور که به‌وضوح در خروجی نشان‌داده شده است، اگر مشتری شماره‌تلفن نداشته باشد، آنگاه تابع CONCAT() از یک فضای خالی برای ادغام استفاده می‌کند.

توجه کنید که از تابع CHAR() برای به‌دست‌آوردن کاراکتر خط جدید در این مثال استفاده کردیم.

تابع CONCAT_WS در SQL Server

تابع CONCAT_WS() دو یا تعداد بیشتری رشته را با استفاده از یک مجزا کننده، به یک‌رشته تبدیل می‌کند. CONCAT_WS() یعنی ادغام با مجزا کنند (concatenate with separator).

کد زیر، syntax تابع CONCAT_WS() را نشان می‌دهد:

```
CONCAT_WS(separator,input_string1,input_string2,[...input_stringN]);
```

در این syntax:

- Separator یک عبارت کاراکتر محور است که مقدار آن می‌تواند از یکی از نوع‌های VARCHAR، NCHAR، CHAR یا NVARCHAR باشد.
- Input_string1 تا input_stringN عباراتی با هر نوع معتبر هستند. تابع CONCAT_WS() قبل از انجام عمل ادغام، به‌صورت ضمنی مقادیری که دارای نوع غیر کاراکتری هستند را به نوع کاراکتری تبدیل می‌کند.

تابع CONCAT_WS() رشته‌های ورودی را به یک‌رشته تبدیل می‌کند. این تابع، رشته‌های در حال ادغام را با separator مشخص شده در اولین آرگومان با هم ادغام می‌کند.

توجه کنید که CONCAT_WS() حداقل به دو رشته ورودی نیاز دارد. یعنی اینکه اگر صفر یا فقط یک‌رشته ورودی وارد کنید، تابع ارور می‌دهد.

تابع `CONCAT_WS()` با مقدار `NULL` همانند یک رشته خالی از نوع `VARCHAR` (۱) برخورد می‌کند. این تابع همچنین در بین مقادیر `NULL` از مجزا کننده استفاده نمی‌کند.

در نتیجه، تابع `CONCAT_WS()` می‌تواند بدون اضافه کردن چیزی، رشته‌های دارای مقادیر خالی را به هم متصل کند.

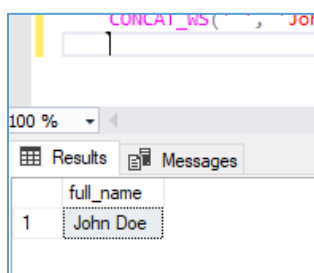
مثال‌های تابع `CONCAT_WS()` در `SQL Server`

الف) استفاده از تابع `CONCAT_WS()` برای اتصال چند رشته با یک مجزا کننده

مثال زیر از تابع `CONCAT_WS()` برای تبدیل دو رشته به یک رشته استفاده می‌کند. در این مثال، دو رشته ادغام شده با استفاده از یک فضای خالی یا `Space` از هم جدا شده‌اند:

```
SELECT
    CONCAT_WS(' ', 'John', 'Doe') full_name
```

خروجی به این شکل است:



ب) استفاده از تابع `CONCAT_WS()` برای ستون‌های جدول

کد زیر از تابع `CONCAT_WS()` برای اتصال مقادیر ستون‌های `last_name` و `first_name` از جدول `sales.customers` استفاده می‌کند. نام خانوادگی و نام با استفاده از `Space` و `ویرگول` از هم جدا شده‌اند:

```
SELECT
    first_name,
    last_name,
    CONCAT_WS(' ', last_name, first_name) full_name
FROM
    sales.customers
ORDER BY
    first_name,
    last_name;
```

تصویر زیر، بخشی از خروجی را نشان می‌دهد:

	first_name	last_name	full_name
1	Aaron	Knapp	Knapp, Aaron
2	Abbey	Pugh	Pugh, Abbey
3	Abby	Gamble	Gamble, Abby
4	Abram	Copeland	Copeland, Abram
5	Adam	Henderson	Henderson, Adam
6	Adam	Thomton	Thomton, Adam
7	Addie	Hahn	Hahn, Addie
8	Adelaida	Hancock	Hancock, Adelaida
9	Adelle	Larsen	Larsen, Adelle
10	Adena	Blake	Blake, Adena
11	Adrien	Hunter	Hunter, Adrien
12	Adriene	Rivera	Rivera, Adriene
13	Adriene	Rollins	Rollins, Adriene
14	Afton	Juarez	Juarez, Afton
15	Agatha	Daniels	Daniels, Agatha
16	Agatha	Melton	Melton, Agatha
17	Agnes	Sims	Sims, Agnes
18	Austin	Levenson	Levenson, Austin

پ) استفاده از تابع `CONCAT_WS()` برای مقدار `NULL`

کد زیر، چگونگی عملکرد تابع `CONCAT_WS()` برای رشته‌های ورودی که دارای مقادیر `NULL` هستند را نشان می‌دهد:

```
SELECT
    CONCAT_WS(',', 1, 2, NULL, NULL, 3);
```

خروجی به این شکل است:

The screenshot shows a SQL query window with the following text:

```
SELECT
    CONCAT_WS(',', 1, 2, NULL, NU
```

The results pane below shows a single row with the value '1,2,3'.

(No column name)
1

همان‌طور که به‌وضوح در خروجی مشاهده می‌کنید، تابع `CONCAT_WS()` مقدار `NULL` را نادیده می‌گیرد و هیچ مجزا کننده‌ای بین مقادیر `NULL` اضافه نمی‌کند.

مثال زیر داده‌های مشتری را ادغام می‌کند تا آدرس‌های مشتریان را فرمت کند. اگر یک مشتری دارای شماره‌تلفن نباشد، تابع آن را نادیده می‌گیرد:

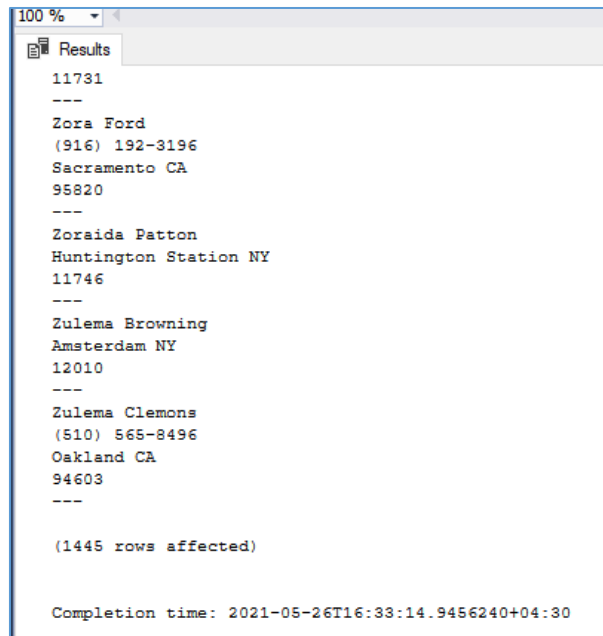
```
SELECT
    CONCAT_WS
    (
        CHAR(13),
        CONCAT(first_name, ' ', last_name),
        phone,
        CONCAT(city, ' ', state),
        zip_code,
        '-----'
    ) customer_address
FROM
```

```

sales.customers
ORDER BY
  first_name,
  last_name;

```

این تصویر، بخشی از خروجی را نشان می‌دهد:



ت) استفاده از تابع `CONCAT_WS()` برای ایجاد فایل CSV
این کد از یک ویرگول و عنوان مجزا کننده استفاده می‌کند و مقادیر موجود در ستون‌های `last_name`، `first_name` و `email` را با هم ادغام می‌کند تا یک فایل CSV ایجاد کند:

```

SELECT
  CONCAT_WS(',', first_name, last_name, email)
FROM
  sales.customers
ORDER BY
  first_name,
  last_name;

```

بخشی از خروجی به این شکل است:

	(No column name)
1	Aaron,Knapp,aaron.knapp@yahoo.com
2	Abbey,Pugh,abbey.pugh@gmail.com
3	Abby,Gamble,abby.gamble@aol.com
4	Abram,Copeland,abram.copeland@gmail.com
5	Adam,Henderson,adam.henderson@hotmail.com
6	Adam,Thomton,adam.thomton@hotmail.com
7	Addie,Hahn,addie.hahn@hotmail.com
8	Adelaida,Hancock,adelaida.hancock@aol.com
9	Adelle,Larsen,adelle.larsen@gmail.com
10	Adena,Blake,adena.blake@hotmail.com
11	Adrien,Hunter,adrien.hunter@yahoo.com

تابع DEFERENCE در SQL Server

تابع `SOUNDEX()` بر اساس اینکه وقتی یک رشته به زبان می‌آید چگونه تلفظ می‌شود، رشته را به کد چهار کاراکتری تبدیل می‌کند.

برای مثال، هر دو واژه‌های `Too` و `Two` تلفظ یکسانی دارند، بنابراین باید دارای مقادیر `SOUNDEX()` مشابه باشند:

SELECT

```
SOUNDEX('Too') Too,
SOUNDEX('Two') Two;
```

خروجی به این ترتیب است:

	Too	Two
1	T000	T000

تابع `DIFFERENCE()` یک مقدار `integer` بازمی‌گرداند که تفاوت بین مقادیر `SOUNDEX()` دو رشته را می‌سنجد. کد زیر، `DIFFERENCE()` تابع `DIFFERENCE()` را نشان می‌دهد:

```
DIFFERENCE(input_string1, input_string2)
```

نتیجه `DIFFERENCE()`، تفاوت بین دو مقدار `SOUNDEX()` را در مقیاس ۰ تا ۴ نشان می‌دهد. مقدار ۰ یعنی ضعیف یا بدون وجود تشابه بین مقادیر `SOUNDEX()` و مقدار ۴ یعنی مقادیر `SOUNDEX()` بسیار مشابه یا همسان.

مثال‌های تابع `DIFFERENCE()` در SQL Server

الف) استفاده از تابع `DIFFERENCE()` با مقادیر `SOUNDEX()` مشابه

این مثال از تابع `DIFFERENCE()` برای مقایسه مقادیر `SOUNDEX()` از دو رشته `Too` و `Two` استفاده می‌کند:

SELECT

```
SOUNDEX('Two') soundex_two,
SOUNDEX('Too') soundex_too,
DIFFERENCE('Two', 'Too') similarity;
```

خروجی به این شکل است:

	soundex_two	soundex_too	similarity
1	T000	T000	4

کد زیر یک مثال دیگر است که تفاوت موجود در مقادیر `SOUNDEX()` از رشته‌های `John` و `Johny` را مقایسه می‌کند:

SELECT

```
SOUNDEX('Johny') soundex_johny,
SOUNDEX('John') soundex_john,
DIFFERENCE('Johny', 'John') similarity;
```

خروجی به این ترتیب است:

	soundex_johny	soundex_john	similarity
1	J500	J500	4

ب) استفاده از تابع DEFFERENCE() برای مقادیر SOUNDEX() مختلف

مثال زیر، چگونگی عملکرد DEFFERENCE() در هنگام مقایسه دو رشته که دارای تلفظهای ضعیف یا غیرمشابه هستند را نشان می‌دهد:

SELECT

```
SOUNDEX('Coffee') soundex_coffee,
SOUNDEX('Laptop') soundex_laptop,
DIFFERENCE('Coffee', 'Laptop') similarity;
```

خروجی به این شکل است:

	soundex_coffee	soundex_laptop	similarity
1	C100	L131	1

تابع LEFT در SQL Server

تابع LEFT() تعداد مشخصی کاراکتر را از سمت چپ یک رشته ارائه شده استخراج می‌کند. برای مثال، کد LEFT('SQL Server', 3) خروجی SQL را نمایش می‌دهد.

Syntax تابع LEFT() به این صورت است:

LEFT (input_string , number_of_characters)

در این syntax:

- Input_string می‌تواند یک رشته، متغیر یا ستون باشد. نوع داده نتیجه input_string می‌تواند هر نوع داده‌ای به جز نوع‌های TEXT یا NTEXT باشد، این دو نوع به صورت ضمنی به VARCHAR و NVARCHAR تبدیل می‌شوند.
- Number_of_characters یک integer مثبت است که تعداد کاراکترهای input_string که بازگشت داده می‌شوند را مشخص می‌کند.

وقتی input_string یک نوع داده‌ای کاراکتر غیر Unicode باشد، آنگاه تابع LEFT() یک مقدار VARCHAR و اگر input_string یک نوع داده کاراکتر Unicode باشد، یک مقدار NVARCHAR بازمی‌گرداند.

مثال‌های تابع LEFT() در SQL Server

الف) استفاده از تابع LEFT() برای یک‌رشته کاراکتر

کد زیر از تابع LEFT() برای بازگرداندن سه کاراکتر اول از سمت چپ رشته کاراکتر SQL Server را نشان می‌دهد:

```
SELECT LEFT('SQL Server',3) Result_string;
```

خروجی به این شکل است:

	Result_string
1	SQL

ب) استفاده از تابع LEFT() برای یک ستون جدول

مثال زیر ۷ کاراکتر اول سمت چپ هر نام محصول در جدول production.products را بازمی‌گرداند:

```
SELECT
    product_name,
    LEFT(product_name, 7) first_7_characters
FROM
    production.products
ORDER BY
    product_name;
```

تصویر زیر بخشی از خروجی را نشان می‌دهد:

	product_name	first_7_characters
1	Electra Amsterdam Fashion 3i Ladies' - 2017/2018	Electra
2	Electra Amsterdam Fashion 7i Ladies' - 2017	Electra
3	Electra Amsterdam Original 3i - 2015/2017	Electra
4	Electra Amsterdam Original 3i Ladies' - 2017	Electra
5	Electra Amsterdam Royal 8i - 2017/2018	Electra
6	Electra Amsterdam Royal 8i Ladies' - 2018	Electra
7	Electra Cruiser 1 (24-Inch) - 2016	Electra
8	Electra Cruiser 1 (24-Inch) - 2016	Electra
9	Electra Cruiser 1 - 2016/2017/2018	Electra
10	Electra Cruiser 1 Ladies' - 2018	Electra
11	Electra Cruiser 1 Tall - 2016/2018	Electra
12	Electra Cruiser 7D (24-Inch) Ladies' - 2016/2018	Electra
13	Electra Cruiser 7D (24-Inch) Ladies' - 2016/2018	Electra
14	Electra Cruiser 7D - 2016/2017/2018	Electra
15	Electra Cruiser 7D Ladies' - 2016/2018	Electra
16	Electra Cruiser 7D Tall - 2016/2018	Electra
17	Electra Cruiser Lux 1 - 2016/2018	Electra

پ) استفاده از تابع LEFT() برای دستور GROUP BY

مثال زیر از تابع LEFT() برای بازگرداندن مجموعه‌ای از حروف اول نام محصول و تعداد هر محصول برای هر حرف اول، استفاده می‌کند:

```
SELECT
LEFT(product_name, 1) initial,
COUNT(product_name) product_count
FROM
production.products
GROUP BY
left(product_name, 1)
ORDER BY
initial;
```

خروجی به این شکل است:

	initial	product_count
1	E	118
2	H	13
3	P	3
4	R	1
5	S	51
6	T	135

تابع LEN در SQL Server

تابع LEN() تعداد کاراکترهای یک‌رشته ورودی (به جز فضاها یا بین حروف) را بازمی‌گرداند. کد زیر، syntax تابع LEN() را نشان می‌دهد:

LEN(input_string)

در این syntax، کد input_string می‌تواند یک‌رشته کاراکتر، عبارت رشته‌ای یا یک ستون از داده‌های رشته‌ای یا دودویی باشد.

تابع LEN() مقداری را بازمی‌گرداند که اگر input_string از نوع داده (VARCHAR(max)، NVARCHAR(max)) یا VARBINARY(max) باشد، نوع داده آن BIGINT است و در غیر این صورت، نوع آن INT است.

مثال‌های تابع LEN() در SQL Server

الف) استفاده از تابع LEN() برای یک‌رشته

مثال زیر از تابع LEN() برای بازگرداندن تعداد کاراکترهای رشته LEN SQL Server و همین رشته با فضاها یا خالی بین حروف استفاده می‌کند.

```
SELECT
LEN('SQL Server LEN') length,
LEN('SQL Server LEN ') length_with_trailing_blanks;
```

خروجی به این شکل است:

Results		Messages	
	length	length_with_trailing_blanks	
1	14	14	

همان‌طور که مشاهده می‌کنید، تابع LEN() از فضاهاى خالی بین حروف صرف‌نظر می‌کند.

ب) استفاده از تابع LEN() برای یک ستون

برای این مثال از جدول production.products از پایگاه‌داده نمونه استفاده می‌کنیم.

کد زیر از تابع LEN() برای بازگرداندن نام محصول و طول آن استفاده کرده و محصولات را بر اساس طول نام محصول مرتب می‌کند.

```
SELECT
    product_name,
    LEN(product_name) product_name_length
FROM
    production.products
ORDER BY
    LEN(product_name) DESC;
```

تصویر زیر، بخشی از خروجی را نشان می‌دهد:

	product_name	product_name_length
1	Electra Townie Balloon 8D EQ Ladies' - 2016/2017/...	53
2	Electra Townie Balloon 8D EQ Ladies' - 2016/2017/...	53
3	Electra Townie Original 7D EQ Ladies' - 2017/2018	49
4	Pure Cycles Westem 3-Speed - Women's - 2015/2016	49
5	Electra Sugar Skulls 1 (20-inch) - Girl's - 2017	48
6	Electra Townie Balloon 7i EQ Ladies' - 2017/2018	48
7	Electra Amsterdam Fashion 3i Ladies' - 2017/2018	48
8	Electra Cruiser 7D (24-Inch) Ladies' - 2016/2018	48
9	Electra Cruiser 7D (24-Inch) Ladies' - 2016/2018	48
10	Electra Townie Balloon 7i EQ Ladies' - 2017/2018	48
11	Electra Tiger Shark 3i (20-inch) - Boys' - 2018	47
12	Electra Sweet Ride 3i (20-inch) - Girls' - 2018	47
13	Electra Tiger Shark 1 (20-inch) - Boys' - 2018	46
14	Electra Sweet Ride 1 (20-inch) - Girl's - 2018	46
15	Electra Soft Serve 1 (16-inch) - Girl's - 2018	46
16	Electra Cyclosaurus 1 (16-inch) - Boy's - 2018	46
17	Sun Bicycles Boardwalk (24-inch Wheels) - 2017	46

تابع LOWER در SQL Server

تابع LOWER() یک‌رشته را به حروف کوچک تبدیل می‌کند. کد زیر، syntax تابع LOWER() را نشان می‌دهد:
LOWER(input_string)

در این syntax، کد input_string می‌تواند یک‌رشته کاراکتر، متغیر، عبارت رشته کاراکتر یا ستون جدول باشد.

نوع `input_string` باید به صورت ضمنی قابل تبدیل شدن به `VARCHAR` باشد. در غیر این صورت، باید از تابع `CAST()` برای تبدیل غیر ضمنی `input_string` استفاده کنید. تابع `LOWER()` حروف کوچک `input_string` را بازمی گرداند.

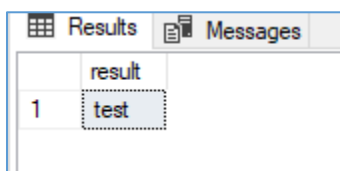
مثال‌های تابع `LOWER()` در SQL Server

الف) استفاده از تابع `LOWER()` برای رشته‌ها

این مثال از تابع `LOWER()` برای تبدیل رشته `'TEST'` به `'test'` استفاده می‌کند:

```
SELECT
    LOWER('TEST') result;
```

خروجی به این شکل است:



	result
1	test

ب) استفاده از تابع `LOWER()` برای ستون جدول

از جدول `customers` از پایگاه داده `Bikestores` در این مثال استفاده می‌کنیم:

کد زیر قبل از انجام عمل ادغام، از تابع `LOWER()` برای تبدیل نام و نام خانوادگی مشتریان به حروف کوچک استفاده می‌کند:

```
SELECT
    first_name,
    last_name,
    CONCAT_WS(
        ' ',
        LOWER(first_name),
        LOWER(last_name)
    ) full_name_lowercase
FROM
    sales.customers
ORDER BY
    first_name,
    last_name;
```

تصویر زیر بخشی از خروجی را نشان می‌دهد:

	first_name	last_name	full_name_lowercase
1	Aaron	Knapp	aaron knapp
2	Abbey	Pugh	abbey pugh
3	Abby	Gamble	abby gamble
4	Abram	Copeland	abram copeland
5	Adam	Henderson	adam henderson
6	Adam	Thornton	adam thorton
7	Addie	Hahn	addie hahn
8	Adelaida	Hancock	adelaida hancock
9	Adelle	Larsen	adelle larsen
10	Adena	Blake	adena blake
11	Adrien	Hunter	adrien hunter
12	Adriene	Rivera	adriene rivera
13	Adriene	Rollins	adriene rollins
14	Aaron	Lvarez	aaron lvarez

به این صورت، چگونگی استفاده از تابع LOWER() برای تبدیل یک رشته به حروف کوچک را نیز آموختید.

تابع LTRIM در SQL Server

تابع LTRIM() پس از حذف فضاهای خالی ابتدا و انتهای رشته، یک رشته بازمی‌گرداند. کد زیر، syntax تابع LTRIM() را نشان می‌دهد:

LTRIM(input_string)

در این syntax، رشته ورودی یک عبارت کاراکتری یا داده‌های باینری است. این عبارت می‌تواند رشته، متغیر یا ستون باشد.

مقدار Input_string باید یک مقدار با نوع داده‌ای (به جز نوع‌های داده‌ای NTEXT، TEXT و IMAGE) باشد که بتوانیم آن را به صورت ضمنی به VARCHAR تبدیل کنیم.

در غیر این صورت، باید از تابع CAST() برای تبدیل غیر ضمنی آن به یک رشته کاراکتر استفاده کنیم.

مثال‌های تابع LTRIM() در SQL Server

الف) استفاده از تابع LTRIM() برای رشته‌ها

این مثال از تابع LTRIM() برای حذف فضاهای خالی ابتدا و انتهای رشته 'SQL Server LTRIM function' استفاده می‌کند:

SELECT

```
LTRIM(' SQL Server LTRIM Function') result;
```

خروجی به این ترتیب است:

	result
1	SQL Server LTRIM Function

ب) استفاده از تابع `LTRIM()` برای حذف فضاهای خالی

مثال زیر پس از تبدیل رشته‌های به چند زیررشته، از تابع `LTRIM()` برای حذف فضاهای خالی ابتدا و انتهای رشته استفاده می‌کند:

```
SELECT
    LTRIM(value) part
FROM
    STRING_SPLIT('Doe, John', ',');
```

خروجی به این شکل است:

	part
1	Doe
2	John

توجه کنید که اگر از تابع `LTRIM()` استفاده نکنید، دومین ردیف یک فضای خالی ابتدا یا انتهای رشته خواهد داشت.

۴-۱۲-۱-۳ توابع سیستمی

توابع سیستمی عبارت‌اند از:

- **CAST** - یک مقدار با یک نوع را به یک نوع دیگر `cast` (یعنی تبدیل صریح) می‌کند.
- **CONVERT** - یک مقدار با یک نوع را به یک نوع دیگر `convert` (یعنی تبدیل ضمنی) می‌کند.
- **CHOOSE** - یکی از دو مقدار را بر اساس نتیجه اولین آرگومان بازمی‌گرداند.
- **ISNULL** - مقدار `NULL` را با یک مقدار مشخص شده جایگزین می‌کند.
- **ISNUMERIC** - بررسی می‌کند که آیا یک عبارت از نوع عددی معتبر هست یا نه.
- **IIF** - منطق `if-else` را به یک `query` اضافه می‌کند.
- **TRY-CAST** - یک مقدار با یک نوع را به یک نوع دیگر `cast` می‌کند و اگر `cast` موفقیت‌آمیز نبود، مقدار `NULL` بازمی‌گرداند.
- **TRY-CONVERT** - یک مقدار با یک نوع را به یک نوع دیگر `convert` می‌کند و مقداری را بازمی‌گرداند که به یک نوع خاص ترجمه شده است. اگر `cast` یا تبدیل موفقیت‌آمیز نباشد، مقدار `NULL` را بازمی‌گرداند.
- **TRY-PARSE** - یک رشته را به یک `date/time` یا یک عدد تبدیل می‌کند و اگر تبدیل موفقیت‌آمیز نباشد، مقدار `NULL` بازمی‌گرداند.
- تبدیل `Datetime` به `string` - چگونگی تبدیل یک مقدار `datetime` به یک رشته با یک فرمت خاص را نشان می‌دهد.
- تبدیل رشته به `datetime` - چگونگی تبدیل یک رشته به یک مقدار `datetime` را نشان می‌دهد.
- تبدیل `datetime` به `date` - یک `Datetime` را به `date` تبدیل می‌کند.

تابع CAST در SQL Server

به query زیر دقت کنید:

`SELECT 1 + '1' AS result;` را بازمی‌گرداند:

	result
1	2

در این کد، SQL Server به صورت ضمنی کاراکتر رشته '۱' را به عدد ۱ تبدیل می‌کند.

وقتی از دو مقدار با نوع‌های داده‌ای مختلف استفاده می‌کنیم، SQL Server سعی می‌کند قبل از اینکه محاسبه را پردازش کند، نوع داده‌ای کمتر را به نوع داده‌ای بیشتر و بالاتر تبدیل کند. به این نوع تبدیل در SQL Server، تبدیل ضمنی گفته می‌شود.

در مقابل تبدیل‌های ضمنی، تبدیل‌های صریح یا غیر ضمنی را داریم که در آن‌ها تابع `CAST()` را فراخوانی می‌کنیم تا به صورت صریح یک مقدار با یک نوع را به یک نوع دیگر تبدیل کند:

`SELECT 1 + CAST(1 AS INT) result;`

	result
1	2

Syntax تابع `CAST()` به این شکل است:

`CAST (expression AS target_type [(length)])`

در این syntax:

- Expression می‌تواند یک رشته یا یک عبارت معتبر با هر نوعی (که قابل تبدیل شدن است) باشد.
- Target_type نوع داده مورد نظر است که می‌خواهید عبارت به آن تبدیل شود. این نوع می‌تواند SQL-INT, BIT, VARIANT و غیره باشد. توجه کنید که این کد نمی‌تواند یک نوع داده‌ای alias باشد.
- Length یک integer اختیاری است که طول نوع داده‌ای مورد نظر را مشخص می‌کند. مقدار پیش فرض length برابر با ۳۰ است.

تابع `CAST()` عبارت تبدیل شده به نوع داده‌ای مورد نظر را بازمی‌گرداند.

مثال‌های تابع `CAST()` در SQL Server

الف) استفاده از تابع CAST() برای تبدیل یک عدد اعشاری به یک عدد صحیح

این مثال از تابع CAST() برای تبدیل عدد اعشاری ۵.۹۵ به یک عدد صحیح استفاده می‌کند:

```
SELECT CAST(5.95 AS INT) result;
```

خروجی به این ترتیب است:

	result
1	5

ب) استفاده از تابع CAST() برای تبدیل یک عدد اعشاری به یک عدد اعشاری دیگر با طول متفاوت

مثال زیر از تابع CAST() برای تبدیل عدد اعشاری ۵.۹۵ به یک عدد اعشاری دیگر با صفر عدد اعشار تبدیل می‌کند:

```
SELECT CAST(5.95 AS DEC(3,0)) result;
```

خروجی به این شکل است:

	result
1	6

وقتی یک مقدار با نوع‌های داده‌ای را به حالت‌های مختلف تبدیل می‌کنید، SQL Server بر اساس قوانین زیر، یک نتیجه کوتاه شده یا یک مقدار گرد شده بازمی‌گرداند:

از نوع داده	به نوع داده	رفتار
numeric	numeric	گرد کردن
numeric	int	کوتاه شدن
Numeric	Money	گرد شدن
Money	Int	گرد شدن
Money	Numeric	گرد شدن
Float	Int	کوتاه شدن
Float	Numeric	گرد شدن
Float	Datetime	گرد شدن
Datetime	Int	گرد شدن

پ) استفاده از تابع CAST() برای تبدیل یک رشته به یک مقدار datetime

این مثال از تابع CAST() برای تبدیل رشته '۲۰۱۹-۰۳-۱۴' به datetime استفاده می‌کند:

```
SELECT
    CAST('2019-03-14' AS DATETIME) result;
```

خروجی به این ترتیب است:

result
2019-03-14 00:00:00.000

ت) استفاده از تابع CAST() با عملگرهای ریاضی

برای این مثال از جداول sales.orders و sales.order_items از پایگاه داده Bikestores استفاده می‌کنیم:
کد زیر از تابع CAST() برای تبدیل فروش‌های ماهیانه در سال ۲۰۱۷ به مقادیر integer استفاده می‌کند.

```
SELECT
    MONTH(order_date) month,
    CAST(SUM(quantity * list_price * (1 - discount)) AS INT) amount
FROM sales.orders o
    INNER JOIN sales.order_items i ON o.order_id = i.order_id
WHERE
    YEAR(order_date) = 2017
GROUP BY
    MONTH(order_date)
ORDER BY
    month;
```

تصویر زیر، خروجی را نشان می‌دهد:

month	amount
1	285616
2	312923
3	308911
4	227290
5	268233
6	378865
7	229995
8	290553
9	293405
10	310328
11	281577
12	259505

به این صورت چگونگی استفاده از تابع CAST() برای تبدیل یک مقدار با یک نوع به یک نوع دیگر را آموختید.

تابع CONVERT در SQL Server

تابع `CONVERT()` به شما امکان می‌دهد تا یک مقدار با یک نوع را به یک نوع دیگر تبدیل کنید. کد زیر، syntax تابع `CONVERT()` را نشان می‌دهد:

`CONVERT (target_type [(length)] , expression [, style])`

در این syntax:

- `Target_type` نوع داده‌ای مورد نظر است که می‌خواهید عبارت به آن تبدیل شود. این نوع می‌تواند `SQL_VARIANT`، `INT` و غیره باشد. توجه کنید که این کد نمی‌تواند یک نوع داده‌ای `Alias` باشد.
- `Length` یک `integer` است که طول نوع داده‌ای مورد نظر را مشخص می‌کند. `length` اختیاری است و مقدار پیش‌فرض آن ۳۰ است.
- `Expression` یک عبارت معتبر با هر نوع داده‌ای قابل تبدیل شدن است.
- `Style` یک `integer` اختیاری است که مشخص می‌کند تابع `CONVERT()` چگونه عبارت را ترجمه می‌کند. اگر مقدار `style` برابر با `NULL` باشد، تابع `CONVERT()` مقدار `NULL` را بازمی‌گرداند.

تابع `CONVERT()` مقدار ترجمه شده `expression` به `target_type` را با یک `style` مشخص شده بازمی‌گرداند.

تابع `CONVERT()` مشابه تابع `CAST()` است. به‌هرحال، برای `SQL Server` خاص و متفاوت است. برخلاف تابع `CONVERT()`، تابع `CAST()` بخشی از توابع `ANSI-SQL` است که به‌طور وسیع در بسیاری از محصولات پایگاه داده‌ای دیگر در دسترس است.

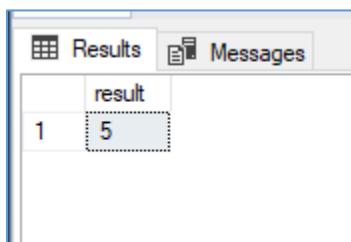
مثال‌های تابع `CONVERT()` در `SQL Server`

الف) استفاده از تابع `CONVERT()` برای تبدیل یک عدد اعشاری به یک `integer`

این مثال از تابع `CONVERT()` برای تبدیل عدد اعشاری ۹.۹۵ به یک `integer` استفاده می‌کند:

```
SELECT CONVERT(INT, 5.95) result;
```

خروجی به این شکل است:



	result
1	5

ب) استفاده از تابع `CONVERT()` برای تبدیل یک عدد اعشاری به یک عدد اعشاری دیگر با طول متفاوت

این مثال از تابع CONVERT() برای تبدیل عدد اعشاری ۹.۹۵ به یک عدد اعشاری دیگر با صفر عدد اعشار استفاده می‌کند:

```
SELECT CAST(9.95 AS DEC(2,0)) result;
```

خروجی به این صورت است:

Results		Messages
result		
1	10	

توجه کنید که رفتارهای گرد کردن و کوتاه کردن تابع CONVERT() دقیقاً مشابه رفتارهای تابع CAST() است. (پ) استفاده از تابع CONVERT() برای تبدیل یک مقدار datetime به یک مقدار

این مثال از تابع CONVERT() برای تبدیل رشته '۲۰۱۹-۰۳-۱۴' به یک مقدار Datetime استفاده می‌کند:

```
SELECT CONVERT(DATETIME, '2019-03-14') result;
```

خروجی به این صورت است:

Results		Messages
result		
1	2019-03-14 00:00:00.000	

ت) استفاده از تابع CONVERT() برای تبدیل یک مقدار datetime به یک رشته

این مثال از تابع CONVERT() برای تبدیل تاریخ و زمان فعلی به یک رشته با style مشخص استفاده می‌کند:

```
SELECT CONVERT(VARCHAR, GETDATE(),13) result;
```

خروجی به این ترتیب است:

Results		Messages
result		
1	26 May 2021 17:22:45:610	

در نتیجه، چگونگی استفاده از تابع CONVERT() برای تبدیل یک مقدار با یک نوع داده‌ای به یک نوع داده دیگر را نیز آموختید.

تابع CHOOSE در SQL Server

تابع CHOOSE() یک آیتم را از لیست آیتم‌های موجود در یک index مشخص شده بازمی‌گرداند.

کد زیر، syntax تابع CHOOSE() را نشان می‌دهد:

```
CHOOSE ( index, elem_1, elem_2 [, elem_n ] )
```

در این syntax:

- Index یک عبارت integer است که index عنصری که باید بازگردانده شود را مشخص می‌کند. توجه کنید که indexهای عناصر بر اساس ۱ هستند.

- Elem_1, elem_2, ... elem_n لیستی از مقادیر مجزا شده با ویرگول هستند که هر نوعی می‌توانند داشته باشند.

اگر index برابر با ۱ باشد، تابع CHOOSE() مقدار elem_1 را بازمی‌گرداند. اگر index برابر با ۲ باشد، تابع CHOOSE() مقدار elem_2 را بازمی‌گرداند و الی آخر.

اگر index یک مقدار integer نباشد، به یک integer تبدیل یا cast می‌شود. در صورتی که index خارج از بازه لیست باشد، تابع CHOOSE() مقدار NULL را بازمی‌گرداند.

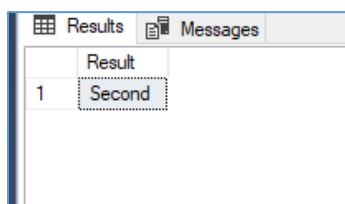
مثال‌های تابع CHOOSE() در SQL Server

الف) استفاده از تابع CHOOSE() برای مقادیر رشته

این مثال، دومین آیتم از لیست مقادیر را بازمی‌گرداند:

```
SELECT  
    CHOOSE(2, 'First', 'Second', 'Third') Result;
```

خروجی به این شکل است:



Result
1 Second

ب) استفاده از تابع CHOOSE() برای ستون جدول

مثال زیر از تابع CHOOSE() برای بازگرداندن وضعیت سفارش بر اساس مقدار موجود در ستون order_status از جدول sales.orders استفاده می‌کند:

```
SELECT  
    order_id,
```

```

order_date,
  CHOOSE(order_status,
    'Pending',
    'Processing',
    'Rejected',
    'Completed') AS order_status
FROM
  sales.orders
ORDER BY
  order_date DESC;

```

تصویر زیر بخشی از خروجی را نشان می‌دهد:

	order_id	order_date	order_status
1	1615	2018-12-28	Rejected
2	1614	2018-11-28	Rejected
3	1613	2018-11-18	Rejected
4	1612	2018-10-21	Rejected
5	1611	2018-09-06	Rejected
6	1610	2018-08-25	Rejected
7	1609	2018-08-23	Rejected
8	1608	2018-07-12	Rejected
9	1607	2018-07-11	Rejected
10	1606	2018-07-10	Rejected
11	1605	2018-07-01	Rejected

پ) استفاده از تابع CHOOSE() همراه با تابع MONTH

مثال زیر از تابع MONTH() برای بازگرداندن فصل‌هایی که در آن‌ها مشتریان محصول خریداری کرده‌اند، استفاده می‌کند. نتیجه تابع MONTH() در تابع CHOOSE() استفاده می‌شود تا فصل متناظر بازگردانده شود:

```

SELECT
  order_id,
  order_date,
  customer_id,
  CHOOSE(
    MONTH(order_date),
    'Winter',
    'Winter',
    'Spring',
    'Spring',
    'Spring',
    'Summer',
    'Summer',
    'Summer',
    'Autumn',
    'Autumn',
    'Autumn',
    'Winter') month
FROM
  sales.orders
ORDER BY
  customer_id;

```

تصویر زیر بخشی از خروجی را نشان می‌دهد:

	order_id	order_date	customer_id	month
1	599	2016-12-09	1	Winter
2	1555	2018-04-18	1	Spring
3	1613	2018-11-18	1	Autumn
4	1509	2018-04-09	2	Spring
5	692	2017-02-05	2	Winter
6	1084	2017-08-21	2	Summer
7	1496	2018-04-06	3	Spring
8	1612	2018-10-21	3	Autumn
9	1468	2018-03-27	3	Spring
10	1259	2017-11-21	4	Autumn
11	1556	2018-04-18	4	Spring

تابع ISNULL در SQL Server

تابع ISNULL() مقدار NULL را با یک مقدار مشخص شده جایگزین می‌کند. کد زیر، Syntax تابع ISNULL() را نشان می‌دهد:

ISNULL(expression, replacement)

تابع ISNULL() دو آرگومان می‌پذیرد:

- Expression یک عبارت با هر نوعی است که بررسی می‌شود که NULL هست یا خیر.
- Replacement مقداری است که اگر عبارت NULL باشد بازگردانده می‌شود. Replacement باید قابل تبدیل شدن به یک مقدار از نوع مقدار موجود در expression باشد.

تابع ISNULL() در صورتی که مقدار expression برابر با NULL باشد، مقدار replacement را بازمی‌گرداند. اگر نوع‌های دو آرگومان متفاوت باشند، این تابع قبل از بازگرداندن یک مقدار، به صورت ضمنی نوع replacement را به نوع expression تبدیل می‌کند.

در صورتی که مقدار expression برابر با NULL نباشد، تابع ISNULL() مقدار expression را بازمی‌گرداند.

مثال‌های تابع ISNULL() در SQL Server

الف) استفاده از تابع ISNULL() برای داده‌های عددی

مثال زیر از تابع ISNULL() برای بازگرداندن دومین آرگومان استفاده می‌کند، چون اولین آرگومان برابر با NULL است:

```
SELECT
    ISNULL(NULL,20) result;
```

خروجی به این شکل است:

	result
1	20

ب) استفاده از تابع ISNULL() برای رشته کاراکتر

مثال زیر از تابع ISNULL() برای بازگرداندن رشته 'Hello' استفاده می‌کند، چون اولین آرگومان است و NULL نیست:

```
SELECT
    ISNULL('Hello', 'Hi') Result;
```

خروجی به این ترتیب است:

	Result
1	Hello

پ) استفاده از تابع ISNULL() برای جایگزینی مقادیر NULL با مقادیر معنادار

ابتدا، یک جدول جدید به نام divisions ایجاد می‌کنیم که دسته‌بندی‌های ورزشکار را بر اساس سن ذخیره می‌کند:

```
CREATE TABLE divisions
(
    id          INT
    PRIMARY KEY IDENTITY,
    min_age    INT DEFAULT 0,
    max_age    INT
);
```

دوم، چند ردیف به جدول divisions اضافه می‌کنیم:

```
INSERT INTO divisions(min_age, max_age)
VALUES(5,null),
      (20,null),
      (null,30);
```

سوم، داده‌ها را از جدول divisions به دست می‌آوریم:

```
SELECT
    id,
    min_age,
    max_age
FROM
    divisions;
```

تصویر زیر، خروجی را نشان می‌دهد:

	id	min_age	max_age
1	1	5	NULL
2	2	20	NULL
3	3	NULL	30

اگر یک دسته‌بندی دارای محدودیت حداقل سن نباشد، پس ستون min_age دارای NULL خواهد بود. به طور مشابه، اگر یک دسته‌بندی نیز به حداکثر سن نداشته باشد، ستون max_age نیز دارای NULL خواهد بود.

در آخر، از تابع ISNULL() برای تبدیل NULL در ستون min_age به ۰ و تبدیل NULL در ستون max_age به ۹۹ استفاده می‌کنیم:

```
SELECT
    id,
    ISNULL(min_age,0) min_age,
    ISNULL(max_age,99) max_age
FROM
    divisions;
```

تصویر زیر، خروجی را نشان می‌دهد:

	id	min_age	max_age
1	1	5	99
2	2	20	99
3	3	0	30

تابع ISNUMERIC در SQL Server

تابع ISNUMERIC() یک عبارت می‌پذیرد و اگر عبارت از نوع عددی معتبری باشد، مقدار ۱ را بازمی‌گرداند، در غیر این صورت، مقدار ۰ را بازمی‌گرداند.

کد زیر، syntax تابع ISNUMERIC() را نشان می‌دهد:

ISNUMERIC (expression)

در این syntax، کد expression می‌تواند هر عبارت معتبری باشد.
توجه داشته باشید که یک نوع عددی معتبر، یکی از نوع‌های زیر است:

اعداد صحیح: TINYINT، SMALLINT، INT، BIGINT و BIT

دارای دقت ثابت: DECIMAL و NUMERIC

تقریبی: FLOAT و REAL

مقادیر پولی: MONEY و SMALLMONEY

تابع ISNUMERIC() در واقع بررسی می‌کند که آیا یک مقدار می‌تواند به یک نوع داده‌ای عددی تبدیل شود یا خیر، و سپس پاسخ صحیح را بازمی‌گرداند. به‌هرحال، این تابع به شما نمی‌گوید که کدام نوع داده‌ای، به‌خوبی می‌تواند جریان یا overflow را کنترل کند.

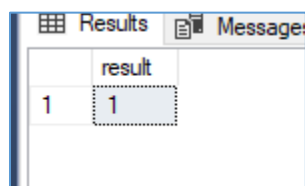
به همین دلیل از زمان انتشار SQL Server 2012 توابع TRY_CAST، TRY_PARSE() و TRY_CONVERT() معرفی شدند.

مثال‌های تابع ISNUMERIC() در SQL Server

این مثال، از تابع ISNUMERIC() استفاده می‌کند تا بررسی کند که آیا رشته '\$۱۰' می‌تواند به یک عدد تبدیل شود یا خیر:

```
SELECT
    ISNUMERIC('$10') result;
```

خروجی به این صورت است:



result
1

مثال زیر بررسی می‌کند که آیا رشته '-E-308۲.۲۳' یک عدد هست یا خیر:

```
SELECT
    ISNUMERIC('-2.23E-308') result;
```

خروجی عبارت است از:

	result
1	1

مثال زیر مقدار ۰ را بازمی‌گرداند که نشان‌دهنده این است که رشته '+ABC' یک عدد نیست:

```
SELECT
    ISNUMERIC('+ABC') result;
```

خروجی به این شکل است:

	result
1	0

تابع IIF در SQL Server

تابع IIF() سه آرگومان می‌پذیرد. سپس اولین آرگومان را ارزیابی می‌کند و اگر مقدار آن true باشد، آنگاه دومین آرگومان را بازمی‌گرداند، در غیر این صورت، سومین آرگومان را بازمی‌گرداند.

کد زیر، syntax تابع IIF() را نشان می‌دهد:

```
IIF(boolean_expression, true_value, false_value)
```

در این syntax:

- Boolean_expression یک عبارت است که باید ارزیابی شود. این عبارت باید یک عبارت boolean معتبر باشد، در غیر این صورت تابع ارور می‌دهد.
- True_value مقداری است که اگر مقدار boolean_expression برابر با true باشد، بازگردانده می‌شود.
- False_value مقداری است که اگر مقدار boolean_expression برابر با false باشد، بازگردانده می‌شود.

در واقع، تابع IIF() خلاصه شده یا short hand یک عبارت CASE است:

```
CASE
    WHEN boolean_expression
        THEN true_value
    ELSE
        false_value
END
```

مثال‌های تابع IIF() در SQL Server

الف) استفاده از تابع IIF() برای یک‌رشته

این مثال از تابع IIF() استفاده می‌کند تا بررسی کند که آیا $10 < 20$ هست یا نه و رشته True را بازمی‌گرداند:

```
SELECT
    IIF(10 < 20, 'True', 'False') Result ;
```

خروجی به این شکل است:

Result
1 True

ب) استفاده از تابع IIF() برای ستون جدول

مثال زیر تابع IIF() را به صورت تودرتو درون توابع IIF() به کار می‌برد و وضعیت سفارش متناظر را بر اساس تعداد سفارش بازمی‌گرداند:

```
SELECT
    IIF(order_status = 1, 'Pending',
        IIF(order_status=2, 'Processing',
            IIF(order_status=3, 'Rejected',
                IIF(order_status=4, 'Completed', 'N/A')
            )
        )
    ) order_status,
    COUNT(order_id) order_count
FROM
    sales.orders
WHERE
    YEAR(order_date) = 2018
GROUP BY
    order_status;
```

تصویر زیر، خروجی را نشان می‌دهد:

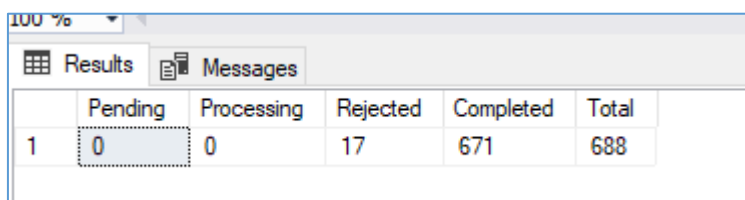
	order_status	order_count
1	Pending	62
2	Processing	63
3	Rejected	13
4	Completed	154

پ) استفاده از تابع IIF() برای توابع تجمعی

این مثال از تابع IIF() برای تابع SUM() استفاده می‌کند تا تعداد سفارش‌ها را بر اساس وضعیت سفارش در سال ۲۰۱۸ به دست آورد:

```
SELECT
    SUM(IIF(order_status = 1, 1, 0)) AS 'Pending',
    SUM(IIF(order_status = 2, 1, 0)) AS 'Processing',
    SUM(IIF(order_status = 3, 1, 0)) AS 'Rejected',
    SUM(IIF(order_status = 4, 1, 0)) AS 'Completed',
    COUNT(*) AS Total
FROM
    sales.orders
WHERE
    YEAR(order_date) = 2017;
```

در این مثال، اگر وضعیت متناظری پیدا شد، تابع IIF() مقدار ۱ یا ۰ را بازمی‌گرداند. تابع SUM() تعداد سفارش‌ها را برای هر وضعیت بازمی‌گرداند. خروجی به این شکل است:



	Pending	Processing	Rejected	Completed	Total
1	0	0	17	671	688

فصل چهارم

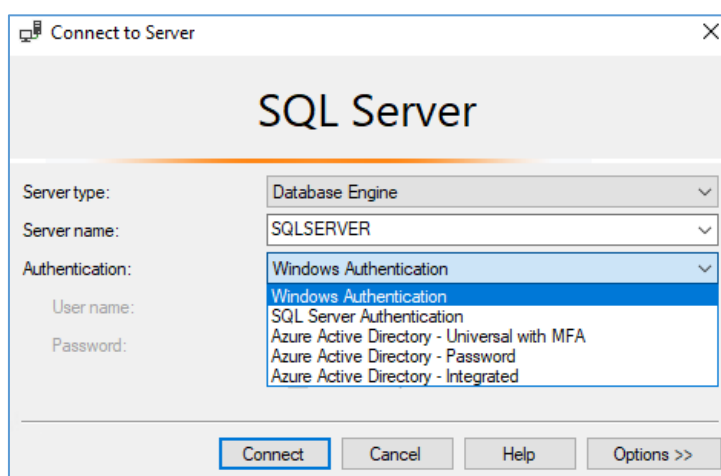
امنیت در SQL Server

یکی از مهم‌ترین بخش‌هایی که باید به آن توجه کرد امنیت در SQL است که اگر درست به آن توجه نکنیم می‌تواند باعث ایجاد مشکلات امنیتی و درز اطلاعات مهم سازمان شود..

زمانی که یک دیتابیس را ایجاد می‌کنید، امنیت زیادی نخواهد داشت و اگر روش ایجاد امنیت بر روی دیتابیس را ندانید، شاید تمام اطلاعات خود را از دست بدهید.

۱-۴ روش‌های احراز هویت

برای اینکه روش‌های احراز هویت در SQL Server 2019 را بررسی کنیم بهتر است صفحه‌ی اول نرم‌افزار SSMS را مشاهده کنید که در شکل ۱-۴ مشخص شده است.



شکل ۱-۴ احراز هویت

همان‌طور که در شکل ۱-۴ مشاهده می‌کنید احراز هویت SQL از چند قسمت تشکیل شده است که در زیر همگی آنها را بررسی خواهیم کرد:

Windows Authentication

یک روش احراز هویت است که در سرویس‌گیرنده‌های ویندوزی تعبیه شده است، این یک روش پیش‌فرض است که خود ویندوز بر روی آن تأکید می‌کند، در زمان نصب SQL این روش به صورت پیش‌فرض اجرا خواهد شد و کاربر موردنظر برای ورود به SQL تأیید خواهد شد.

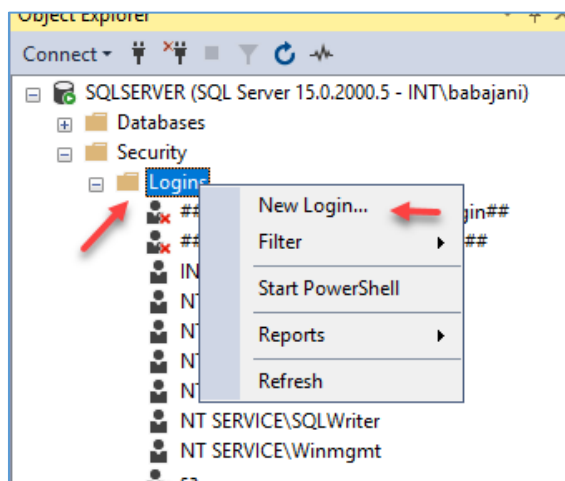
اصولاً سازمان‌ها برای مدیریت آسان‌تر کاربران یک سرویس Active directory در شبکه خود ایجاد می‌کنند و کاربران را به صورت متمرکز در آن تعریف می‌کنند و بعد می‌توان از طریق احراز هویت Windows Authentication به این کاربران دسترسی داد، اصولاً هر کاربر در ویندوز یک شماره‌ی خاص با نام SID دارد که با بقیه متمایز خواهد بود.

SQL Server Authentication

این روش از احراز هویت مختص نرم‌افزار SQL است و رمز عبور و نام کاربری در داخل دیتابیس نرم‌افزار تعریف می‌شود.

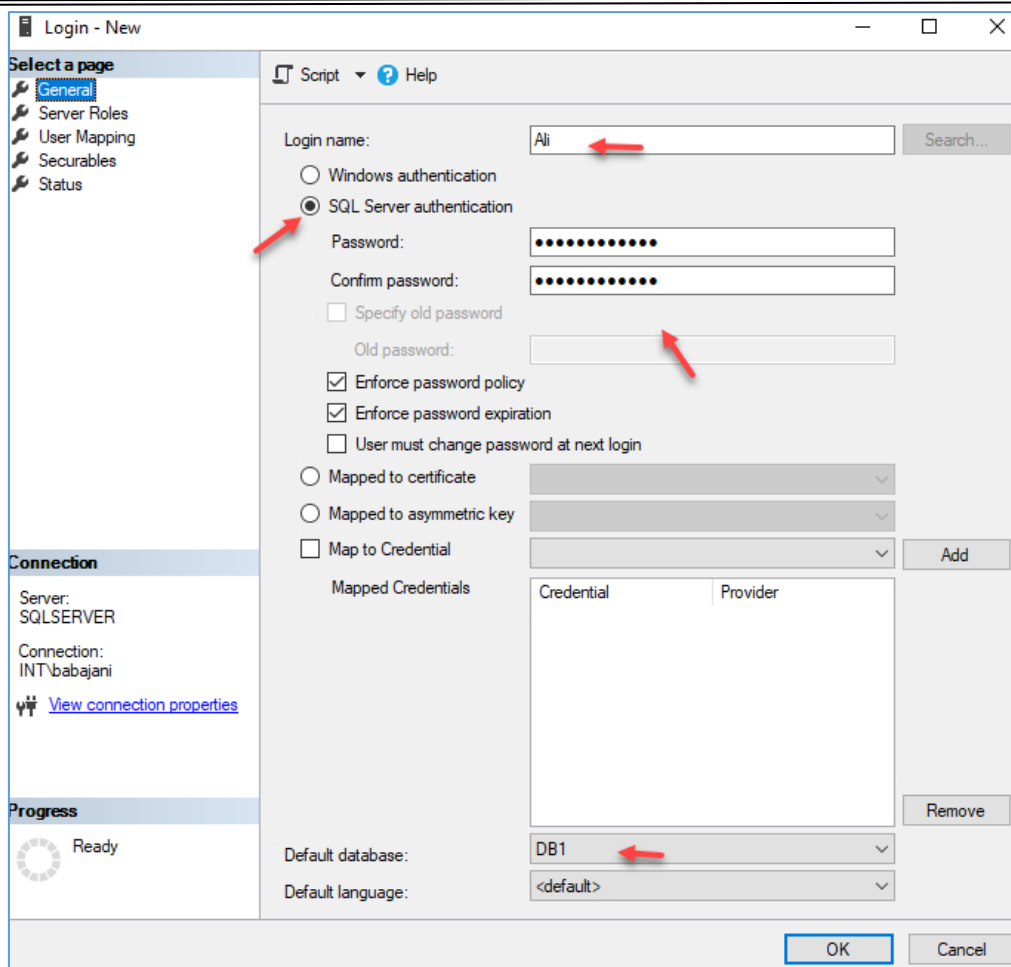
۲-۴ ایجاد کاربر و اعطای دسترسی به آن

همان‌طور که در اوایل کتاب گفتیم، زمانی که سرور SQL را عضو شبکه دامین می‌کنید، می‌توانید از کاربران Active Directory برای دسترسی به SQL استفاده کنید و مجوزهای لازم را برای آن در نظر بگیرید، اما اگر بخواهید کاربر محلی در SQL ایجاد کنید باید به‌مانند شکل ۲-۴ این کار را انجام دهید:



شکل ۲-۴ / ایجاد کاربر

در شکل ۳-۴ باید کاربر خود را ایجاد کنید، در قسمت Login Name باید نام کاربر خود را وارد کنید و در زیر آن SQL Server Authentication را انتخاب کنید، با انتخاب این گزینه شما یک کاربر در داخل SQL Server ایجاد می‌کنید.



شکل ۳-۴ / ایجاد کاربر

یک رمز عبور پیچیده وارد کنید مانند Test@12345، اگر می‌خواهید رمز عبور شما با خط مش سازمان شما یکی باشد باید تیک گزینه‌ی Enforce password policy را انتخاب کنید، خط مش سازمان برای رمز عبور می‌تواند موارد زیر باشد:

زمانی که این گزینه را انتخاب می‌کنید باید موارد زیر رعایت شود:

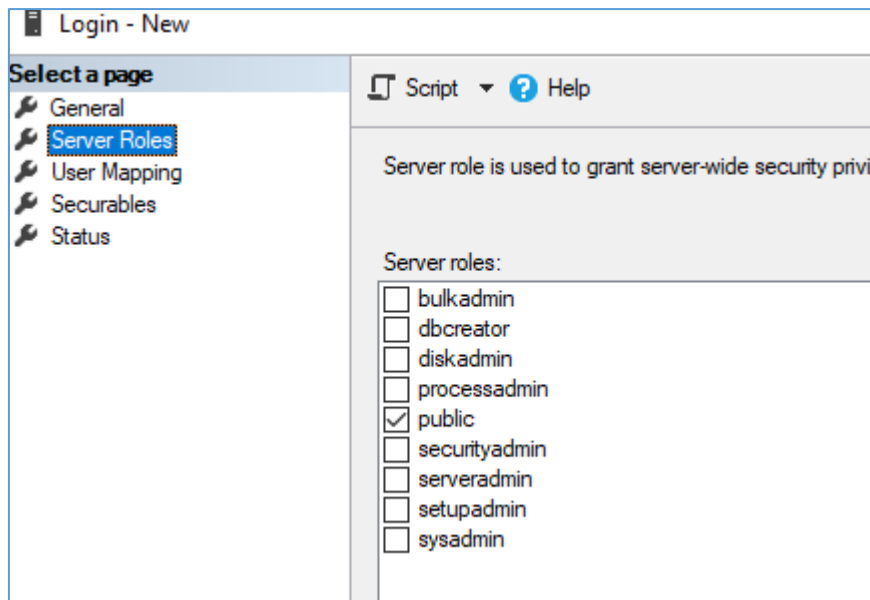
- ۱- رمز عبور نباید شامل نام کاربری باشد.
- ۲- رمز عبور حداقل هشت کاراکتر باشد.
- ۳- رمز عبور باید شامل حروف (a-z)، (A-Z) و اعداد (0-9) باشد.
- ۴- رمزهای غیرالفبایی مانند: تعجب (!)، علامت دلار (\$)، علامت عدد (#) یا درصد (%).

در ادامه کار اگر تیک گزینه‌ی Enforce Password Expiration را انتخاب کنید به این معنا است که یک تاریخ برای انقضای رمز عبور کاربران در نظر گرفته خواهد شد و قبل از اینکه تاریخ انقضای برسد به کاربران هشدارهای لازم داده خواهد شد تا رمز عبور خود را تغییر دهند.

گزینه‌ی User Must Change Passeowrd at next Login را انتخاب کنید کاربر بعد از ورود باید رمز عبور را خودش تغییر دهد.

در آخر شکل ۳-۴ می‌توانید کاربر موردنظر را به دیتابیس موردنظر خود تخصیص دهید، تا دسترسی اولیه به آن را داشته باشد.

در ادامه کار به‌مانند شکل ۴-۴ وارد تب Server Role شوید، در این قسمت گزینه‌های مختلف دسترسی وجود دارد که در جدول ۴-۱ به طور کامل آن را بررسی می‌کنیم.



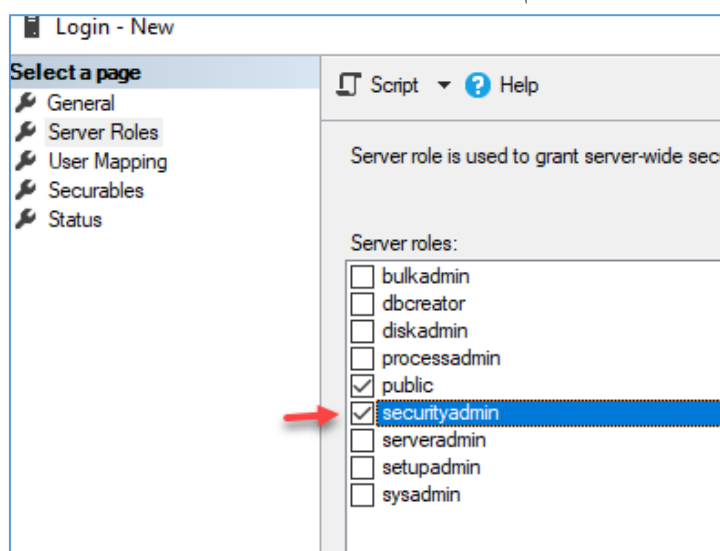
شکل ۴-۴ دسترسی به کاربر

جدول ۴-۱ بررسی Role دسترسی

توضیحات	نام Role
کاربران عضو sysadmin می‌توانند هر فعالیتی را در سرور انجام دهند. یعنی آزاد و رها هستند.	sysadmin
کاربران عضو serveradmin می‌توانند پیکربندی سرور موردنظر را تغییر دهند و همچنین می‌توانند سرور را خاموش کنند.	serveradmin
اعضای گروه securityadmin می‌توانند قسمت Login و جزئیات آنها را مدیریت کنند، کاربرانی که عضو این نقش هستند توانایی این را دارند که مجوزهای سطح سرور را اعطا کنند مانند GRANT و DENY و یا REVOKE، آنها همچنین می‌توانند در صورت دسترسی به دیتابیس مجوزهای آن را هم مدیریت کنند، علاوه بر این هم آنها می‌توانند رمزهای عبور کاربران را ریست کنند. به این نکته توجه کنید که این نقش بسیار قدرتمند است و باید مواظب باشید که به هر کاربری آن را اعطا نکنید، این نقش معادل نقش sysadmin در سرور است.	securityadmin
کاربرانی که عضو نقش processadmin باشند می‌توانند فرایندهای در حال اجرا را خاتمه دهند.	processadmin

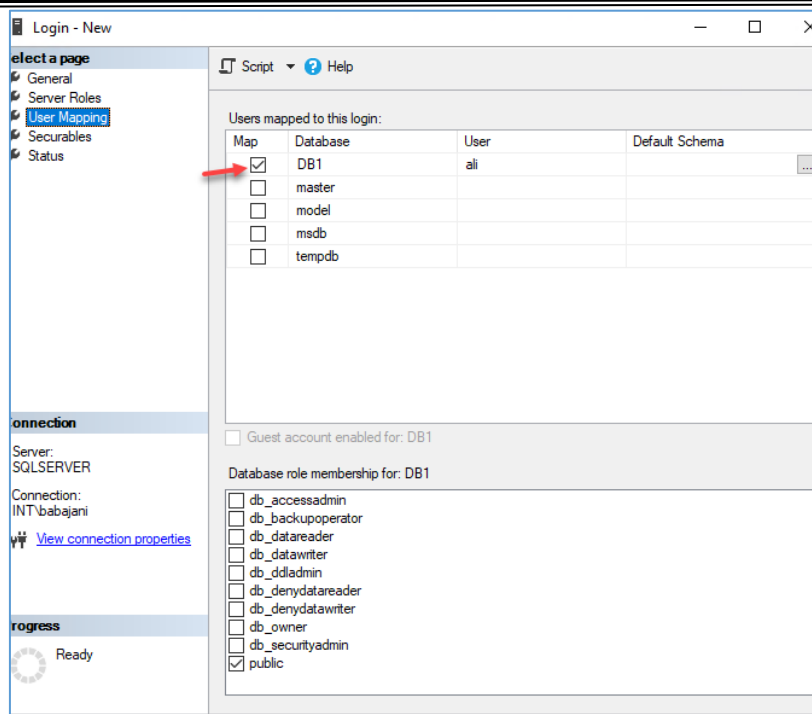
کاربران این نقش می‌توانند با استفاده از زبان پرس‌وجو یا همان Transact-SQL سرورها را اضافه یا حذف کنند، البته در صورت استفاده از Management Studio کاربر موردنظر حتماً باید عضو sysadmin هم باشد.	setupadmin
اعضای نقش bulkadmin می‌توانند دستور BULK INSERT را اجرا کنند (دستور BLANK INSERT می‌تواند با استفاده از فایل‌های txt یا csv مقادیر زیادی از اطلاعات را به جداول اضافه کند).	bulkadmin
کاربران عضو نقش diskadmin می‌توانند فایل‌های موجود بر روی دیسک را مدیریت کنند.	diskadmin
کاربران دارای نقش dbcreator می‌توانند هر پایگاه داده‌ای را ایجاد، مدیریت و حذف کنند و حتی می‌توانند پایگاه داده‌ی موردنظر را بازیابی کنند.	dbcreator
زمانی که یک کاربر ایجاد می‌کنید به‌صورت پیش‌فرض نقش public به آن تعلق می‌گیرد، این نقش به این صورت است که زمانی که مدیر یک object ایجاد می‌کند می‌تواند مشخص کند که کاربرانی که نقش public را دارند بتوانند به آن object دسترسی داشته باشند.	public

برای اینکه به کاربر ali دسترسی کامل دهیم تیک گزینه‌ی securityadmin را به‌مانند شکل ۴-۵ انتخاب می‌کنیم.



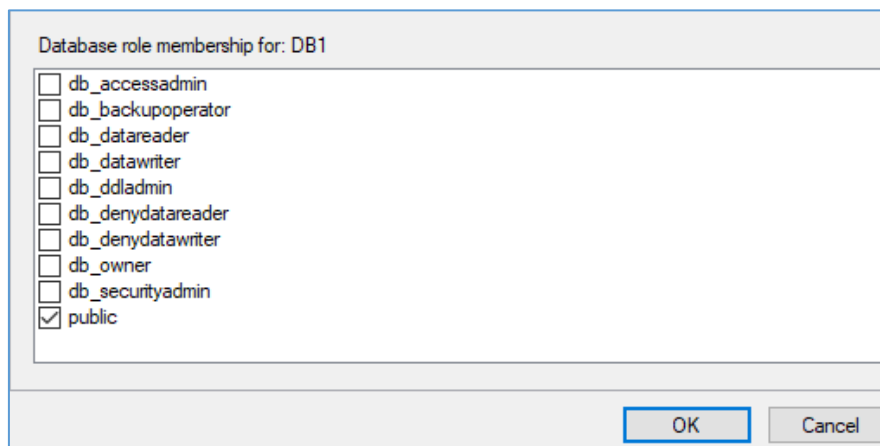
شکل ۴-۵ دسترسی به کاربر

در ادامه کار وارد تب User Mapping شوید در این صفحه که در شکل ۴-۶ مشخص شده است، شما می‌توانید مشخص کنید که کاربر موردنظر بر روی کدام دیتابیس دسترسی لازم را داشته باشد.



شکل ۴-۶ دسترسی به دیتابیس

در پایین شکل ۴-۶ که در شکل ۴-۷ مشخص شده است چندین Role وجود دارد که می‌توانید برای دسترسی به دیتابیس توسط کاربر موردنظر مشخص کنید، برای اینکه بیشتر با گزینه‌های موردنظر آشنا شویم به جدول ۴-۲ توجه کنید.



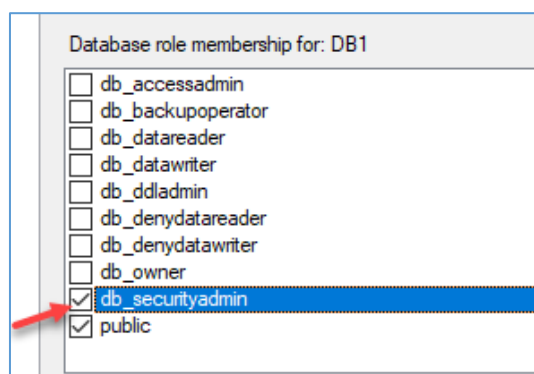
شکل ۴-۷ دسترسی به دیتابیس

جدول ۴-۲ Role دسترسی

توضیحات	نام Role
کاربرانی که نقش db_owner را دریافت می‌کنند می‌توانند بر کلیه تنظیمات و نگهداری پایگاه داده موردنظر را انجام دهند و همچنین می‌توانند پایگاه داده را در sql حذف کنند،	db_owner

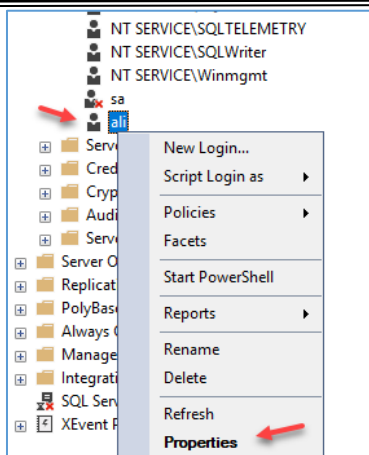
البته در بعضی از مواقع این نوع دسترسی کارایی ندارد و باید مجوز سطح سرور به آنها داده شود.	
اعضای این نقش توانایی تغییر عضویت در نقش‌ها را دارند و همچنین می‌توانند وظیفه مدیریت اختیارات را برعهده دارند.	db_securityadmin
این نقش به کاربران این اجازه را می‌دهد که بر روی Login کردن در SQL نظارت و دسترسی داشته باشد.	db_accessadmin
این قابلیت را به کاربران می‌دهد تا بتوانند از پایگاه‌داده پشتیبان تهیه کنند.	db_backupoperator
اعضای این گروه توانایی اجرای دستورات از نوع DDL را دارند.	db_ddladmin
اعضای این گروه توانایی تغییر، حذف، و اضافه‌کردن هیچ اطلاعاتی را در جدول‌های تعریف شده توسط کاربر ندارند.	db_datawriter
اعضای این نقش، توانایی خواندن اطلاعات از جدول‌های تعریف شده توسط کاربر را دارد.	db_datareader
اعضای این نقش نمی‌توانند داده‌های موجود در جداول کاربران موجود در یک پایگاه‌داده را اضافه، اصلاح یا حذف کنند.	db_denydatawriter
کاربرانی که این نقش را دارند نمی‌توانند هیچ داده‌ای را از جدول‌های تعریف شده توسط کاربر بخوانند.	db_denydatareader

در شکل ۴-۸ تیک گزینه‌ی db_securityadmin را انتخاب کنید تا کاربر مورد نظر دسترسی کامل به دیتابیس داشته باشد



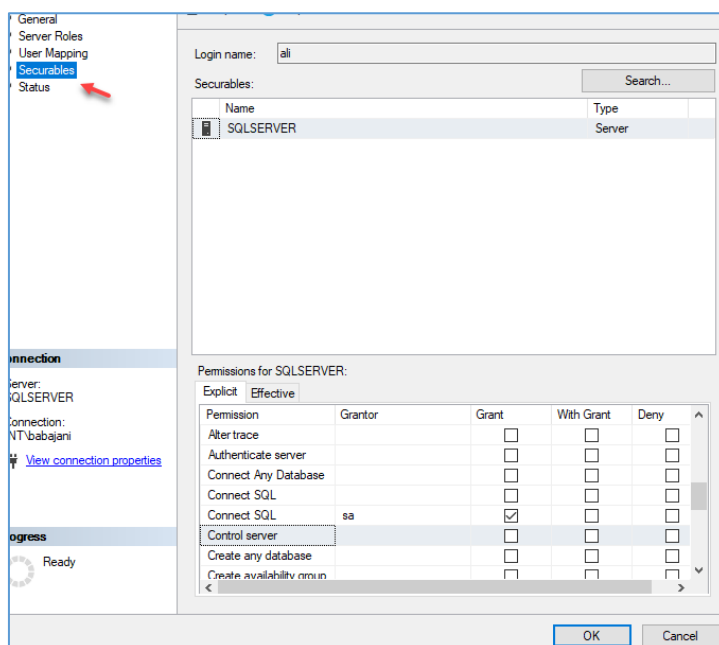
شکل ۴-۸ دسترسی به دیتابیس

بعد از انجام کار می‌توانید بر روی OK کلیک کنید تا کاربر مورد نظر ایجاد شود، بعد از ایجاد کاربر به‌مانند شکل ۹-۴ بر روی کاربر مورد نظر کلیک راست کنید و گزینه‌ی Properties را انتخاب کنید.



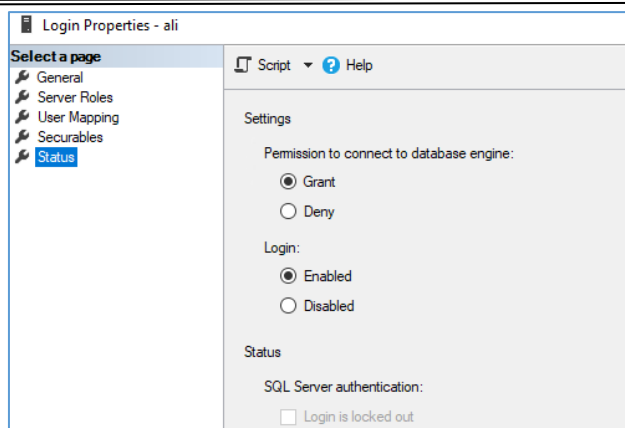
شکل ۴-۹ بررسی کاربر

در شکل ۴-۱۰ جزئیات مربوط به کاربر ali را مشاهده می‌کنید، در تب Securables یک سری دسترسی‌های پیشرفته وجود دارد که می‌توانید از طریق مدیر SQL به کاربران و دیتابیس مورد نظر اعطا کنید.



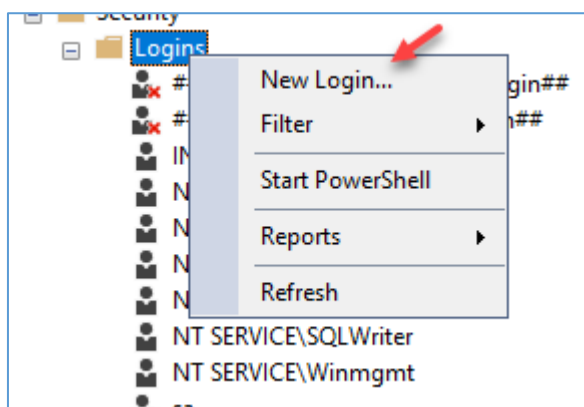
شکل ۴-۱۰ بررسی تب Securables

در شکل ۴-۱۱ مشخص شده است می‌توانید مشخص کنید که کاربر مورد نظر توانایی متصل شدن به Database Engine را داشته باشد یا نه و یا اینکه کاربر مورد نظر را فعال و یا غیرفعال کنید.



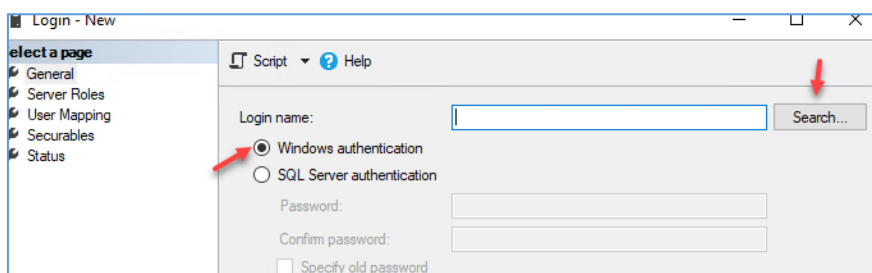
شکل ۴-۱۱ بررسی قسمت Status

اگر بخواهیم یک کاربر از طریق Active Directory را به لیست Logins اضافه کنیم باید به‌مانند شکل ۴-۱۲ بر روی Logins کلیک راست کنید و گزینه‌ی New Login را انتخاب کنید.



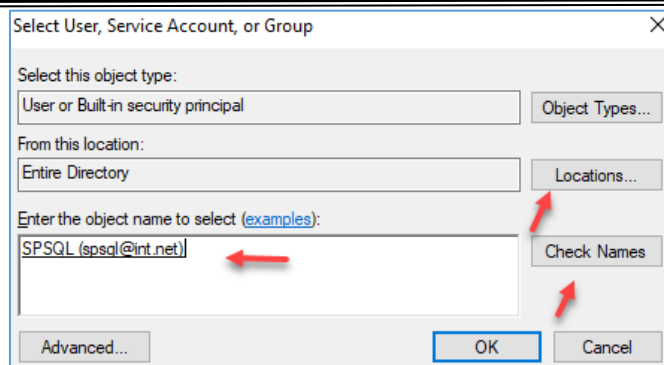
شکل ۴-۱۲ ایجاد کاربر Active Directory

در شکل ۴-۱۳ باید گزینه‌ی Windows authentication را انتخاب کنید و بر روی Search کلیک کنید.



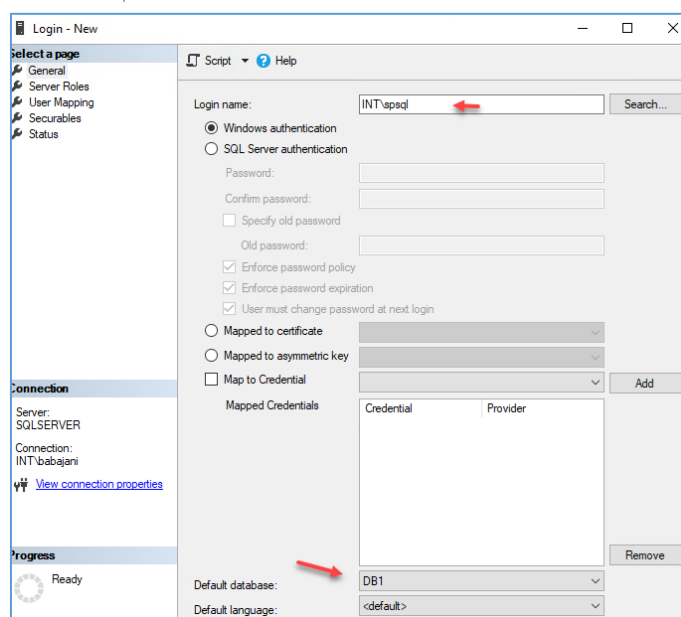
شکل ۴-۱۳ ایجاد کاربر Active Directory

در شکل ۴-۱۴ باید از قسمت Location دومین خود را انتخاب کنید و کاربر مورد نظر خود را در کادر مشخص شده وارد کنید تا با کلیک بر روی Check Names آن را پیدا کنید.



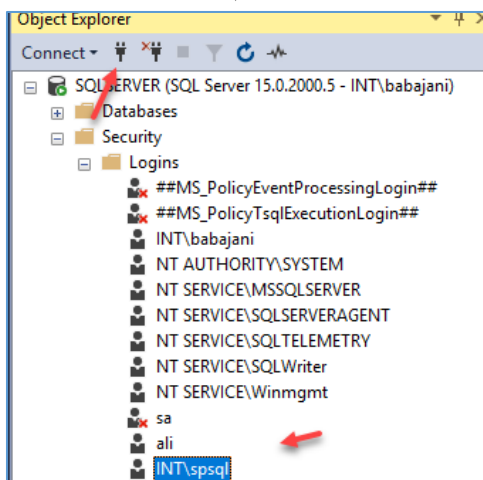
شکل ۴-۱۴ ایجاد کاربر Active Directory

اگر به شکل ۱۵-۴ توجه کنید بعد از انتخاب کاربر از نوع دومین دیگر نمی‌توانید رمز عبور و دیگر گزینه‌ها را خودتان وارد و انتخاب کنید بلکه این اطلاعات از طریق سرویس Active Directory انجام می‌گیرد.



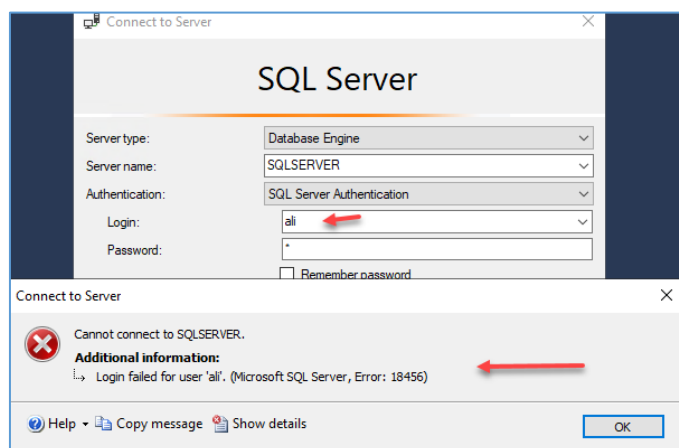
شکل ۴-۱۵ ایجاد کاربر Active Directory

همان‌طور که در شکل ۱۶-۴ مشاهده می‌کنید هم کاربر ali و هم کاربر spsql که از نوع دومین بوده به لیست SQL اضافه شده است، برای اینکه با این کاربران وارد SQL شویم باید بر روی آیکن مورد نظر کلیک کنید.



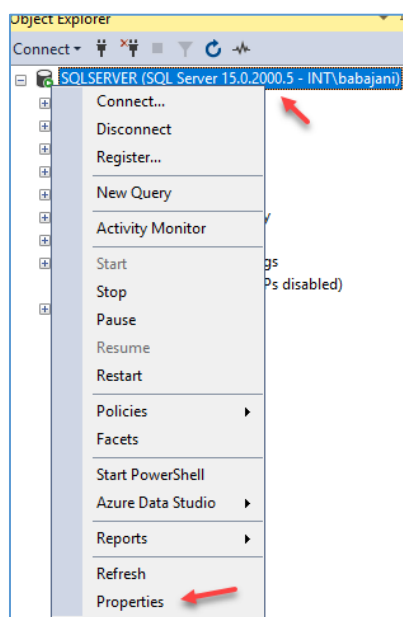
شکل ۴-۱۶ ایجاد کاربر Active Directory

قبل از اینکه ادامه دهیم باید این نکته را متذکر شویم که برای ورود با کاربر تحت دومین باید با همان کاربر اول وارد ویندوز شوید و بعد از آن می‌توانید با انتخاب windows Authentication از آن کاربر استفاده کنید ولی برای ورود با کاربر ali که از نوع SQL Authentication است باید گزینه‌ی SQL Server Authentication را انتخاب کنید و طبق شکل ۴-۱۷ کاربر ALI را وارد و بر روی Connect کلیک کنید، بعد از کلیک با خطایی روبرو خواهید شد که اجازه ورود به سرور را به کاربر ali نمی‌دهد.



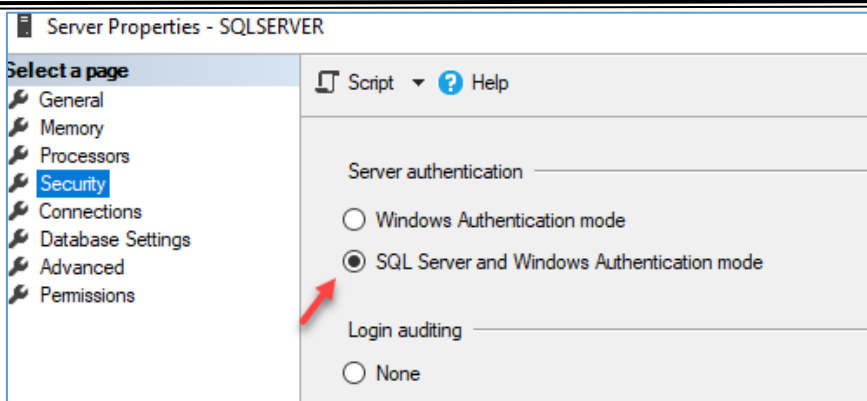
شکل ۴-۱۷ احراز هویت SQL

برای حل خطای شکل ۴-۱۷ باید به‌مانند شکل ۴-۱۸ بر روی سرور کلیک راست کنید و گزینه‌ی Properties را انتخاب کنید.



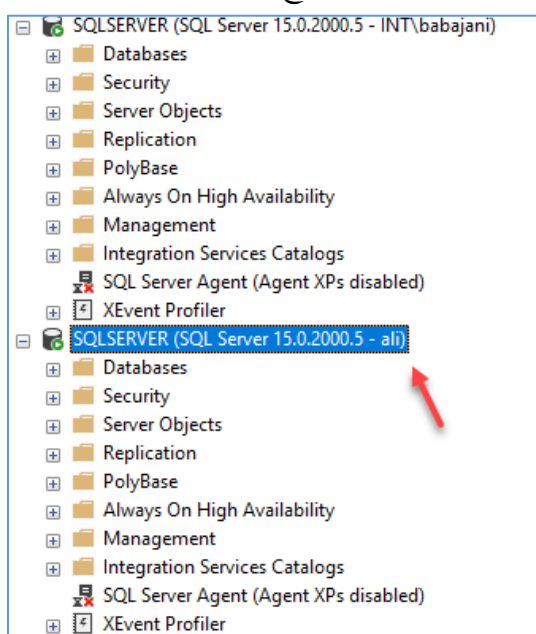
شکل ۴-۱۸ احراز هویت SQL

در شکل ۴-۱۹ وارد تب Security شوید و گزینه‌ی SQL Server and Windows Authentication mode را انتخاب کنید، با این کار هم با احراز هویت SQL و هم Windows می‌توانیم وارد سرور SQL شویم.



شکل ۴-۱۹ احراز هویت SQL

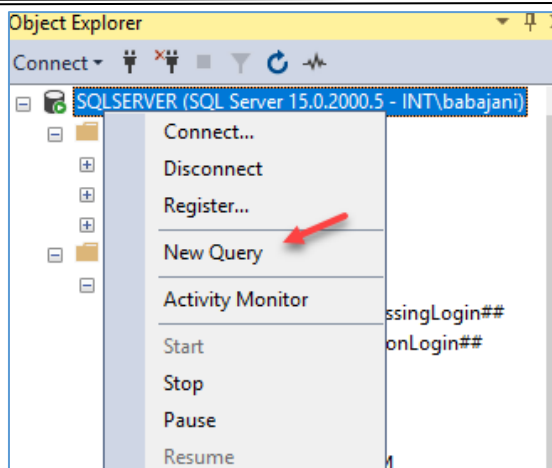
در شکل ۴-۲۰ مشخص شده است که کاربر ali که از نوع SQL Authentication است توانسته وارد سرور SQL شود.



شکل ۴-۲۰ احراز هویت SQL

۴-۳ ایجاد کاربر از طریق Query

یکی دیگر از راه‌های ایجاد کاربر یا موارد خاص دیگر استفاده از دستورات SQL است که در این قسمت می‌خواهیم این کار را انجام دهیم، برای شروع باید بر روی نام سرور SQL کلیک راست کنید و گزینه‌ی New Query را به‌مانند شکل ۴-۲۱ انتخاب کنید.



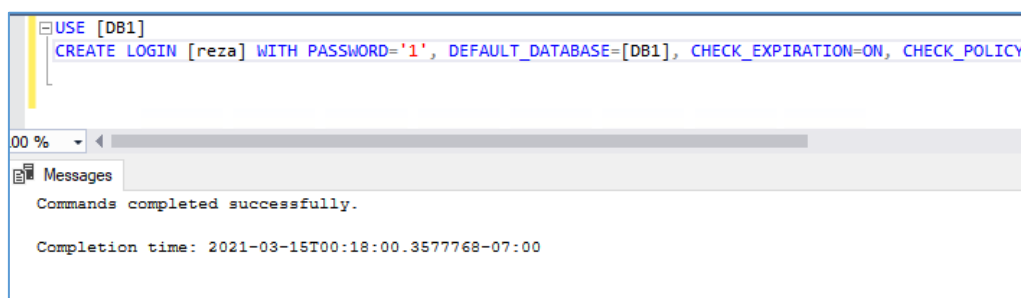
شکل ۴-۲۱ ایجاد کاربر از طریق Query

در صفحه بازشده‌ی شکل ۴-۲۲ دستورات زیر را وارد کنید و بعد از آن کلید F5 را فشار دهید:

```
USE [DB1]
CREATE LOGIN [reza] WITH PASSWORD='1', DEFAULT_DATABASE=[DB1], CHECK_EXPIRATION=ON,
CHECK_POLICY=ON
```

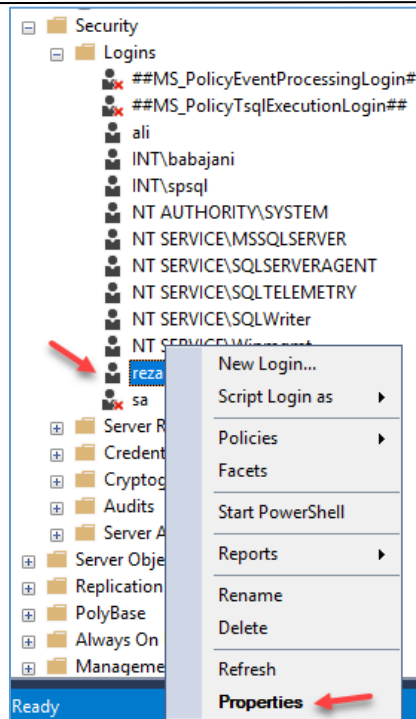
در دستور بالا و در خط اول باید مشخص کنیم که بر روی چه دیتابسی قرار است کار کنیم که در اینجا دیتابیس DB1 انتخاب شده است، در خط بعد باید دستورات اصلی را وارد کنیم، با دستور CREATE LOGIN مشخص می‌کنیم که می‌خواهیم یک LOGIN جدید ایجاد کنیم که بعد از آن هم در [] نام کاربر را که reza است وارد می‌کنیم و با دستور WITH PASSWORD رمز عبور آن را که حتماً هم باید در " باشد را مشخص می‌کنیم، در ادامه باید ویژگی‌های این کاربر را مشخص کنیم، مثلاً برای اینکه مشخص کنیم که دیتابیس پیش‌فرضی که کاربر بر روی آن کار می‌کند را باید با دستور DEFAULT_DATABASE=[DB1] مشخص کنیم و بعد از آن می‌توانیم ویژگی‌های دیگر آن مانند Enforce Password Expiration و Enforce Password Policy را فعال کنید.

بعد از درست وارد کردن دستور مورد نظر بر روی کلید F5 فشار دهید تا به‌مانند شکل ۴-۲۲ کاربر مورد نظر ایجاد شود.



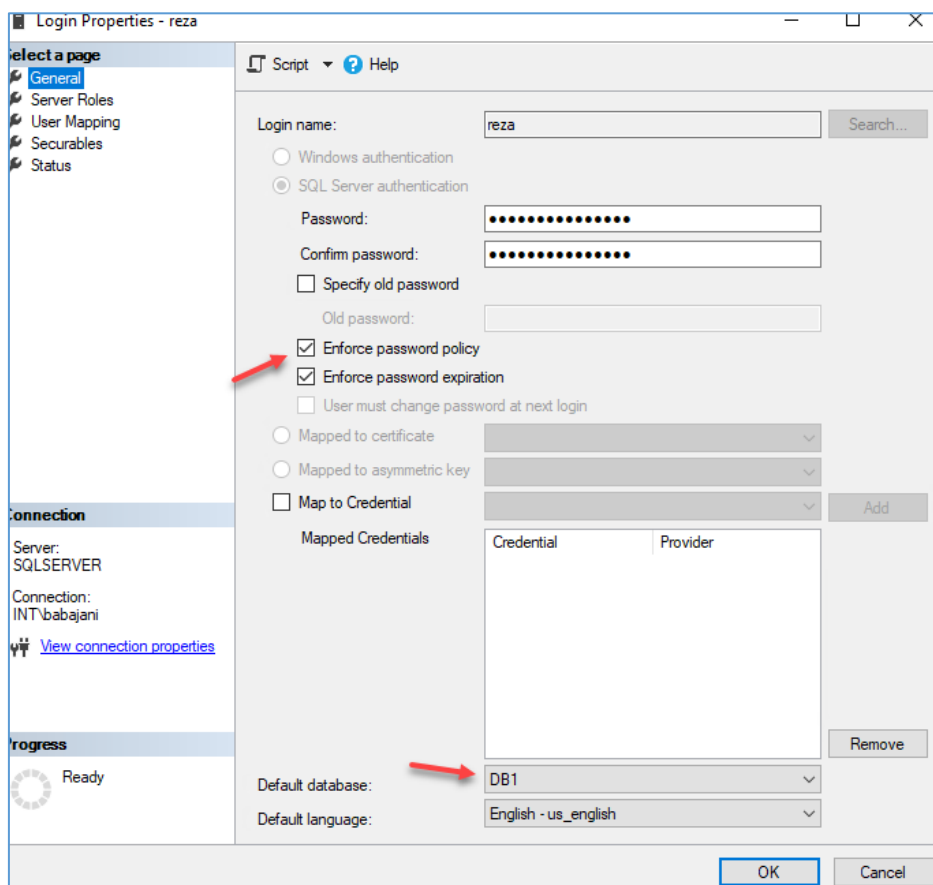
شکل ۴-۲۲ ایجاد کاربر از طریق Query

اگر به شکل ۴-۲۳ توجه کنید، کاربر مورد نظر در لیست Logins قرار گرفته است و برای اینکه بررسی بیشتری داشته باشیم بر روی آن کلیک راست کنید و گزینه‌ی Properties را انتخاب کنید.



شکل ۴-۲۳ بررسی کاربر

همان‌طور که در شکل ۴-۲۴ مشاهده می‌کنید اطلاعات کاربر reza دقیقاً همان چیزی است که در دستورات وارد کردیم.



شکل ۴-۲۴ بررسی کاربر

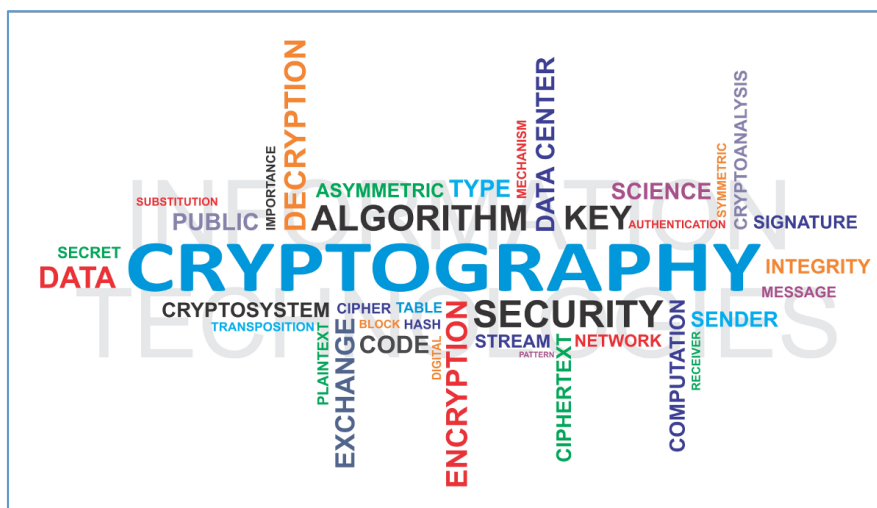
اگر بخواهیم با استفاده از دستور مشخص کنیم که کاربر مورد نظر فعال باشد و یا غیرفعال باید دستور زیر را وارد کنیم:

```
ALTER LOGIN reza ENABLE;
```

در دستور بالا کاربر reza فعال می‌شود و اگر بخواهید غیرفعال کنید باید از دستور DISABLE استفاده کنید.

۴-۴ رمز گذاری بر روی دیتابیس

یکی از مهم‌ترین کارها در SQL ایجاد امنیت برای دیتابیس‌ها و ستون‌های جدول آن است که با این کار دسترسی به اطلاعات دیتابیس کار هرکسی نخواهد بود و طرف باید رمز عبور و گواهینامه مورد نیاز را داشته باشد. مثلاً شما در جدول خود یک ستون دارید که نمی‌خواهید کس دیگری بتواند اطلاعات آن را مشاهده کند و شما با داشتن رمز عبور و گواهینامه مورد نظر میتونید اطلاعات را مشاهده کنید. رمزنگاری یا همان Cryptography به دانشی گفته می‌شود که در آن اطلاعات به صورت یک کلید رمزنگاری می‌شود که این کار توسط الگوریتم مربوط به آن انجام می‌شود و فقط کسی می‌تواند از این اطلاعات استفاده کند که اطلاعات لازم مانند کلید و الگوریتم مربوط به آن را بداند.



شکل ۴-۲۵ رمزنگاری

بررسی برخی از اصطلاحات

Decryption

برای آشکارسازی اطلاعات Encryption شده مورد استفاده قرار می‌گیرد و نام آن را رمزگشایی هم می‌نامند.

Plain text

متن اولیه که رمزنگاری نشده و یک رمز آشکار است و مهاجمان به راحتی می‌توانند به آن دست پیدا کنند.

Cipher

الگوریتمی برای رمز گذاری و رمز شکنی است و از سرعت عمل خوبی برخوردار است.

Cryptanalysis

به بازکردن قفل‌های Cipher گفته می‌شود یا خواندن متن قفل شده‌ی آن.

۱-۴-۴ کلیدهای متقارن (Symmetric) و نامتقارن (Asymmetric)

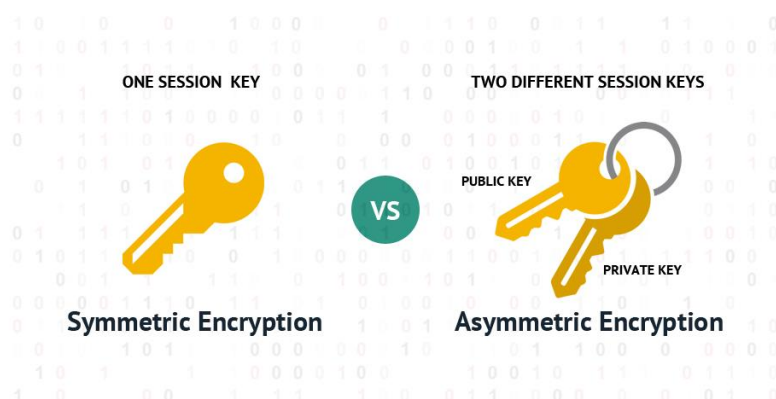
رمزنگاری کلید متقارن

رمزنگاری کلید متقارن یا تک کلیدی، به آن دسته از الگوریتم‌ها، پروتکل‌ها و سیستم‌های رمزنگاری گفته می‌شود که در آن هر دو طرف ردوبدل اطلاعات از یک کلید رمز یکسان برای عملیات رمزگذاری و رمزگشایی استفاده می‌کنند. در این قبیل سیستم‌ها، یا کلیدهای رمزگذاری و رمزگشایی یکسان هستند یا با رابطه‌ای بسیار ساده از یکدیگر قابل استخراج هستند.

واضح است که در این نوع از رمزنگاری، باید یک کلید رمز مشترک بین دو طرف تعریف گردد. چون کلید رمز باید کاملاً محرمانه باقی بماند، برای ایجاد و ردوبدل کلید رمز مشترک باید از کانال امن استفاده نمود یا از روش‌های رمزنگاری نامتقارن استفاده کرد. نیاز به وجود یک کلید رمز به‌ازای هر دو نفر درگیر در رمزنگاری متقارن، موجب بروز مشکلاتی در مدیریت کلیدهای رمز می‌گردد.

الگوریتم‌هایی که در Symmetric به کار می‌رود عبارت‌اند از:

- DES ✓
- DES^۳ ✓
- AES ✓
- IDEA ✓
- RC2, RC4, RC5, RC6 ✓
- Blowfish ✓



شکل ۲۶-۴ رمزنگاری

رمزنگاری کلید نامتقارن

رمزنگاری کلید نامتقارن، در ابتدا با هدف حل مشکل انتقال کلید در روش متقارن پیشنهاد شد. در این نوع از رمزنگاری، به‌جای یک کلید مشترک، از یک زوج کلید به نام‌های کلید عمومی و کلید خصوصی استفاده می‌شود. کلید خصوصی تنها در اختیار دارنده آن قرار دارد و امنیت رمزنگاری به محرمانه بودن کلید خصوصی بستگی دارد. کلید عمومی در اختیار کلیه کسانی که با دارنده آن در ارتباط هستند قرار داده می‌شود.

به‌مرور زمان، به‌غیر از حل مشکل انتقال کلید در روش متقارن، کاربردهای متعددی برای این نوع از رمزنگاری مطرح گردیده است. در سیستم‌های رمزنگاری نامتقارن، بسته به کاربرد و پروتکل مورد نظر، گاهی از کلید عمومی برای

رمزگذاری و از کلید خصوصی برای رمزگشایی استفاده می‌شود و گاهی نیز، برعکس، کلید خصوصی برای رمزگذاری و کلید عمومی برای رمزگشایی به کار می‌رود.

دو کلید عمومی و خصوصی با یکدیگر متفاوت هستند و با استفاده از روابط خاص ریاضی محاسبه می‌گردند. رابطه ریاضی بین این دو کلید به گونه‌ای است که کشف کلید خصوصی با در اختیار داشتن کلید عمومی، عملاً ناممکن است.

الگوریتم‌هایی که در Asymmetric به کار می‌روند عبارت‌اند از:

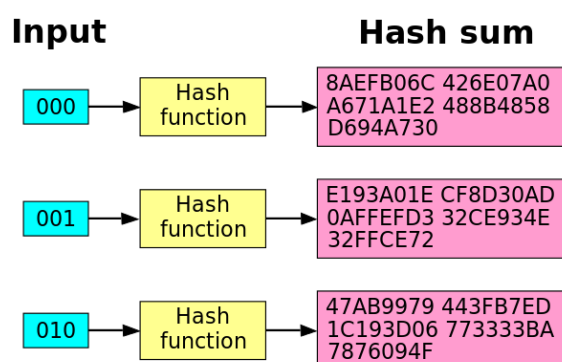
- RSA ✓
- DH ✓
- ElGamal ✓
- DSA ✓
- ECC ✓

مقایسه رمزنگاری کلید متقارن و کلید نامتقارن

اصولاً رمزنگاری کلید متقارن و کلید نامتقارن دارای دو ماهیت متفاوت هستند و کاربردهای متفاوتی نیز دارند؛ بنابراین مقایسه این دو نوع رمزنگاری بدون توجه به کاربرد و سیستم مورد نظر کار دقیقی نخواهد بود. اما اگر معیار مقایسه، به‌طور خاص، حجم و زمان محاسبات مورد نیاز باشد، باید گفت که با در نظر گرفتن مقیاس امنیتی معادل، الگوریتم‌های رمزنگاری متقارن خیلی سریع‌تر از الگوریتم‌های رمزنگاری نامتقارن می‌باشند.

۲-۴-۴ هش کردن (Hashing)

در این روش یک ورودی از اطلاعات دریافت می‌شود و بعد از اجرای یک الگوریتم بر روی آن ورودی تبدیل به اعداد و حروف خواهد شد که در شکل ۲۷-۴ این موضوع را مشاهده می‌کنید که مثلاً با ورود عدد ۰۰۰ و اعمال الگوریتم هش روی آن کد نهایی آن به صورت کامل تغییر کرده و هک کردن آن کاملاً سخت شده است.



شکل ۲۷-۴ هشینگ

انواع الگوریتم‌های هش عبارت‌اند از:

نوع الگوریتم	اندازه
BLAKE-256	256 bits
BLAKE-512	512 bits
BLAKE2s	Up to 256 bits
BLAKE2b	Up to 512 bits

ECOH	224 to 512 bits
FSB	160 to 512 bits
GOST	256 bits
Grøstl	Up to 512 bits
HAS-160	160 bits
HAVAL	128 to 256 bits
JH	224 to 512 bits
MD2	128 bits
MD4	128 bits
MD5	128 bits
MD6	Up to 512 bits
RadioGatún	Up to 1216 bits
RIPEMD	128 bits
RIPEMD-128	128 bits
RIPEMD-160	160 bits
RIPEMD-320	320 bits
SHA-1	160 bits
SHA-224	224 bits
SHA-256	256 bits
SHA-384	384 bits
SHA-512	512 bits
SHA-3 (originally known as Keccak)	arbitrary
Skein	arbitrary
Snefru	128 or 256 bits
Spectral Hash	512 bits
Streebog	256 or 512 bits
SWIFFT	512 bits
Tiger	192 bits
Whirlpool	512 bits

از بین این الگوریتم‌ها بیشترین استفاده از الگوریتم‌های MD5، SHA1، SHA2 می‌شود.

۳-۴ رمزگذاری بر روی ستون‌های جداول در SQL

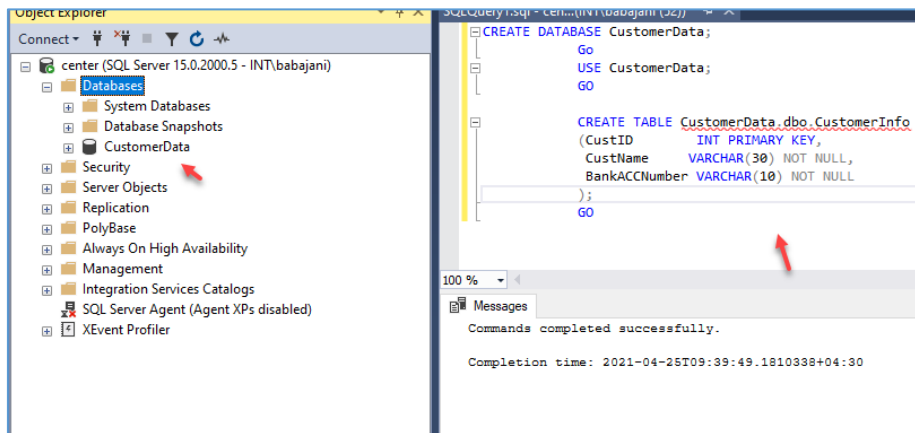
امنیت داده برای هر سازمانی یک کار اساسی و مهم است، به‌خصوص اگر اطلاعات شخصی مشتری مانند شماره تماس، آدرس ایمیل، شماره تأمین اجتماعی، شماره کارت‌های بانکی و اعتباری را ذخیره کنید. هدف اصلی ما محافظت از دسترسی غیرمجاز به داده‌ها در داخل و خارج از سازمان است. برای دستیابی به این هدف SQL Server راه‌حل‌های رمزگذاری ارائه می‌دهد. ما می‌توانیم از این رمزگذاری‌ها استفاده کنیم و از داده‌ها محافظت کنیم.

برای شروع کار می‌خواهیم یک دیتابیس جدید ایجاد کنیم و یکی از ستون‌های آن را رمزنگاری کنیم، برای اینکه یک دیتابیس جدید ایجاد کنیم می‌توانید از دستور زیر استفاده کنید:

```
CREATE DATABASE CustomerData;
Go
USE CustomerData;
GO
```

```
CREATE TABLE CustomerData.dbo.CustomerInfo
(CustID INT PRIMARY KEY,
CustName VARCHAR(30) NOT NULL,
BankACNumber VARCHAR(10) NOT NULL
);
GO
```

با اجرای دستورات بالا یک دیتابیس جدید با نام CustomerData ایجاد شده است و در ادامه دستورات یک جدول با نام CustomerInfo هم داخل دیتابیس CustomerData ایجاد شده که دارای سه ستون است که در شکل ۲۸-۴ این موضوع را مشاهده می‌کنید.

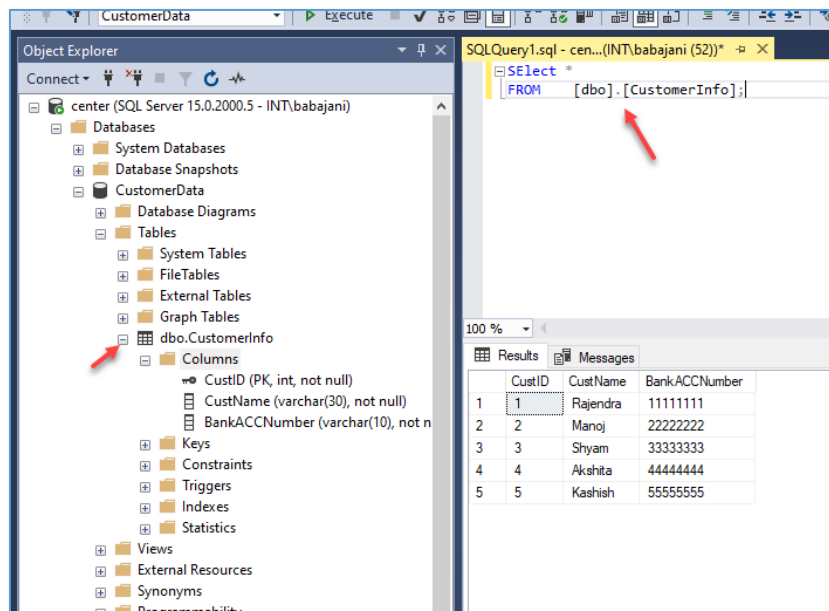


شکل ۲۸-۴ ایجاد جدول و دیتابیس

برای اینکه اطلاعات جدول را تکمیل کنیم می‌توانید از دستورات زیر استفاده کنید:

```
Insert into CustomerData.dbo.CustomerInfo (CustID,CustName,BankACCCNumber)
Select 1, 'Rajendra',11111111 UNION ALL
Select 2, 'Manoj',22222222 UNION ALL
Select 3, 'Shyam',33333333 UNION ALL
Select 4, 'Akshita',44444444 UNION ALL
Select 5, 'Kashish',55555555
```

با دستور بالا اطلاعات جدول CustomerInfo کامل خواهد شد که در شکل ۲۹-۲ این موضوع را مشاهده می‌کنید.



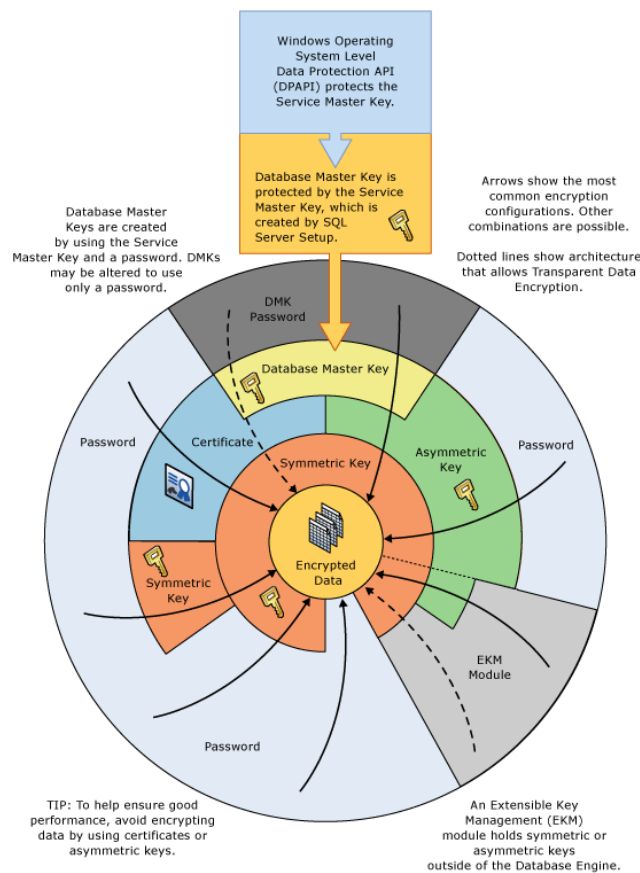
شکل ۲۹-۴ خروجی جدول CustomerInfo

از مراحل زیر برای رمزگذاری سطح ستون استفاده می‌کنیم :

- یک کلید اصلی یا همان master key برای پایگاه داده ایجاد کنید.

- برای SQL Server یک گواهینامه خود امضا ایجاد کنید.
- یک کلید متقارن را برای رمزگذاری پیکربندی کنید.
- رمزگذاری داده‌های ستون.
- رمزگذاری را جستجو و تأیید کنید.

برای انجام این مراحل دقیقاً طبق دستورات زیر پیش بروید تا مشکلی در کار پیش نیاید. اگر به شکل ۴-۳۶ توجه کنید، یک نمای کلی از ایجاد رمزگذاری در SQL Server را مشاهده می‌کنید.



شکل ۴-۳۶ نمودار رمزگذاری

۱-۳-۴-۱ ایجاد Master Key برای رمزگذاری رو ستون

برای شروع یک Master Key تعریف و یک رمز عبور برای محافظت آن مشخص می‌کنیم که این کلید یک کلید متقارن است که برای محافظت از کلیدهای خصوصی و کلید نامتقارن تعریف می‌شود که در شکل ۴-۳۶ هم مشخص شده است.

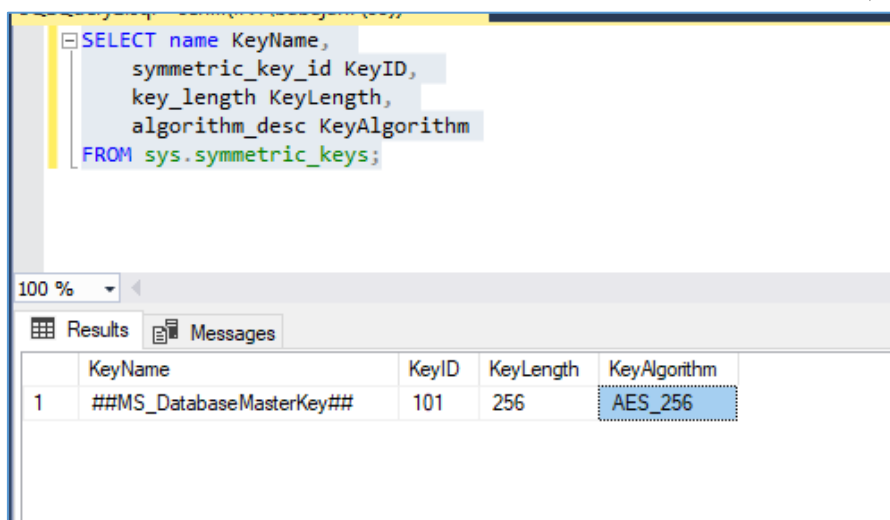
برای ایجاد یک کلید اصلی (Master Key) پایگاه‌داده از عبارت CREATE MASTER KEY استفاده می‌کنیم:

```
USE CustomerData;
GO
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Test@12345';
```

برای اینکه متوجه شویم دستور بالا به درستی اجرا شده است باید از دستور زیر استفاده کنیم:

```
SELECT name KeyName,
       symmetric_key_id KeyID,
       key_length KeyLength,
       algorithm_desc KeyAlgorithm
FROM sys.symmetric_keys;
```

همان‌طور که در شکل ۴-۳۷ مشاهده می‌کنید با اجرای دستورات بالا یک View با ستون‌های مشخص شده در خروجی به نمایش گذاشته شد که مقدار sys.symmetric_keys را نمایش می‌دهد که اگر به ستون الگوریتم توجه کنید الگوریتم ما از نوع AES و با اندازه 256 است که یک رمزنگاری قدرتمند را ارائه می‌دهد، به این نکته هم توجه کنید که خود SQL نوع الگوریتم به همراه طول آن را به صورت اتوماتیک ایجاد می‌کند.



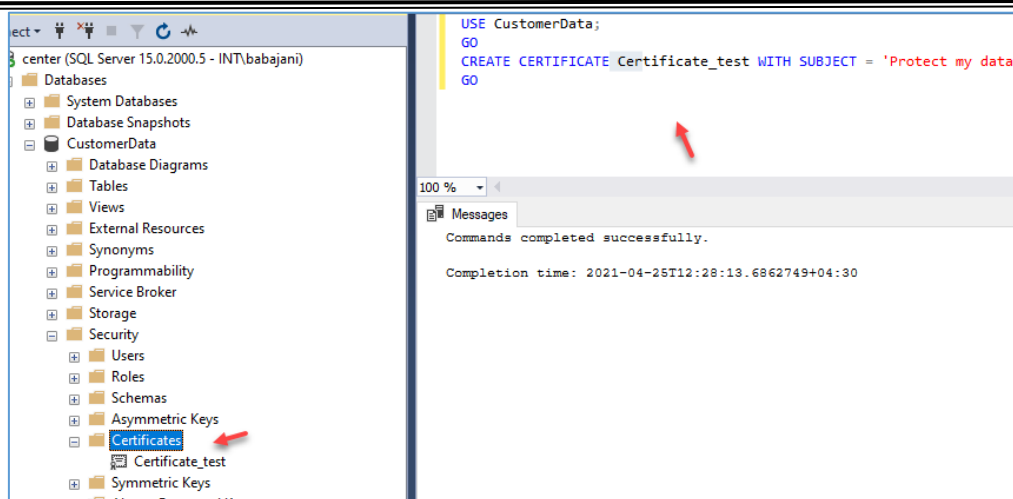
شکل ۴-۳۷

۴-۳-۲-۲ ایجاد Certificate در SQL

در ادامه کار باید یک گواهینامه خود امضا یا همان self-signed ایجاد کنیم منظور از گواهینامه‌های خود امضاء این است که ای گواهینامه داخل خود SQL ایجاد می‌شود و هیچ سازمان دیگری آن را تولید نمی‌کند یعنی دیگر SQL نیاز ندارد گواهینامه مرجع تولید شده از سازمان دیگر استفاده کند بلکه خودش تولید و استفاده می‌کند.

```
USE CustomerData;
GO
CREATE CERTIFICATE Certificate_test WITH SUBJECT = 'Protect my data';
GO
```

در دستور بالا اول به دیتابیس CustomerData متصل شدیم و بعد یک گواهینامه‌ی جدید با نام Certificate_test ایجاد کردیم که موضوع آن Protect my data است و اگر به شکل ۴-۳۸ توجه کنید این موضوع را مشاهده خواهید کرد.

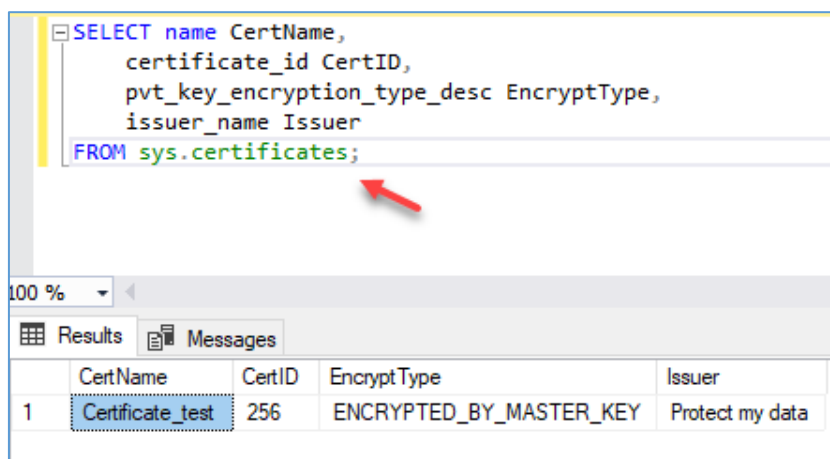


شکل ۴-۳۸ ایجاد Certificate

برای اینکه مطمئن شویم کار به درستی انجام گرفته می‌توانیم از دستورات زیر استفاده کنیم:

```
SELECT name CertName,
       certificate_id CertID,
       pvt_key_encryption_type_desc EncryptType,
       issuer_name Issuer
FROM sys.certificates;
```

اگر به شکل ۴-۳۹ توجه کنید متوجه خواهید شد که Certificate مورد نظر به درستی ایجاد شده است.



شکل ۴-۳۹ نمایش Certificate

پس اگر به ستون‌های شکل ۴-۳۹ توجه کنید در قسمت Encrypt Type مقدار ENCRYPTED_BY_MASTER_KEY قرار گرفته است که نشان می‌دهد SQL از کلیدی که ایجاد کردیم در حال استفاده است، در ستون Certname هم که نام Certificate قرار می‌گیرد، و در ستون Issuer باید نام سازمان صادرکننده گواهینامه نوشته شود که در اینجا چون خود SQL صادرکننده آن است یک نوشته خودمان قرار دادیم.

۳-۳-۴ ایجاد کلید متقارن

در مرحله بعد کار باید یک کلید متقارن ایجاد کنیم، در مورد کلید متقارن در قسمت‌های قبل توضیح دادیم، در کلید متقارن برای رمزگذاری و رمزگشایی استفاده می‌کند. برای شروع باید از دستور زیر استفاده کنید:


```
CREATE SYMMETRIC KEY SymKey_test WITH ALGORITHM = AES_256 ENCRYPTION BY CERTIFICATE Certificate_test;
```

در دستور بالا SymKey_test نام کلید متقارن است که باید ایجاد کنیم و AES_256 همان طول و نوع رمزنگاری است که مشخص شده است و در آخر باید نام Certificate که در قسمت قبل ایجاد کردیم را وارد کنید.

برای اینکه متوجه شویم دستور به درستی اجرا شده می‌توانیم از دستورات زیر استفاده کنیم:

```
SELECT name KeyName,
       symmetric_key_id KeyID,
       key_length KeyLength,
       algorithm_desc KeyAlgorithm
FROM sys.symmetric_keys;
```

همان‌طور که در شکل ۴۰-۴ مشاهده می‌کنید کلید متقارن هم به همراه کلید Master ایجاد شده است.

	KeyName	KeyID	KeyLength	KeyAlgorithm
1	##MS_DatabaseMasterKey##	101	256	AES_256
2	SymKey_test	256	256	AES_256

شکل ۴۰-۴ نمایش کلید متقارن

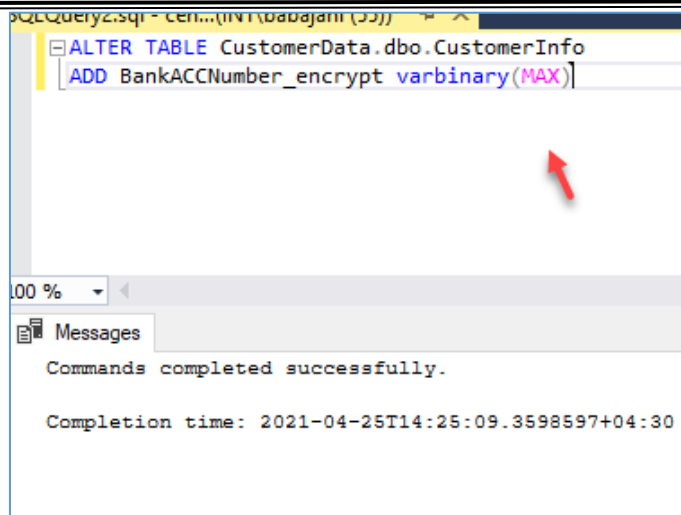
خوب تا به اینجای کار توانستیم کلیدهای رمزگذاری مورد نظر را برای این نسخه از دیتابیس ایجاد کنیم و روش ایجاد آن به‌مانند شکل ۳۶-۴ است یعنی اینکه SQL Server یک Service Master Key (SMK) ایجاد می‌کند و بعد از آن سیستم عامل (DPAPI) Windows Data Protection API از کلید Service Master Key (SMK) محافظت می‌کند، توجه داشته باشیم که کلید اصلی سرویس (SMK) از کلید اصلی پایگاه داده (DMK) محافظت می‌کند کلید اصلی پایگاه داده (DMK) از گواهی خود امضا شده که همان Certificate باشد محافظت می‌کند و این گواهی نام یا همان Certificate از کلید Symmetric محافظت می‌کند. این شد مراحل کار تا به اینجا.

۴-۳-۴ رمزگذاری داده

برای اینکه یک ستون در SQL رمزگذاری شود باید نوع داده را VARBINARY(max) در نظر بگیریم، برای این کار و در دیتابیس CustomerData یک ستون جدید با نام BankACCNumer_encrypt ایجاد و نوع آن را varbinary(MAX) در نظر می‌گیریم.

```
ALTER TABLE CustomerData.dbo.CustomerInfo
ADD BankACCNumer_encrypt varbinary(MAX)
```

همان‌طور که در شکل ۴۱-۴ مشاهده می‌کنید ستون مورد نظر با موفقیت ایجاد شده است.

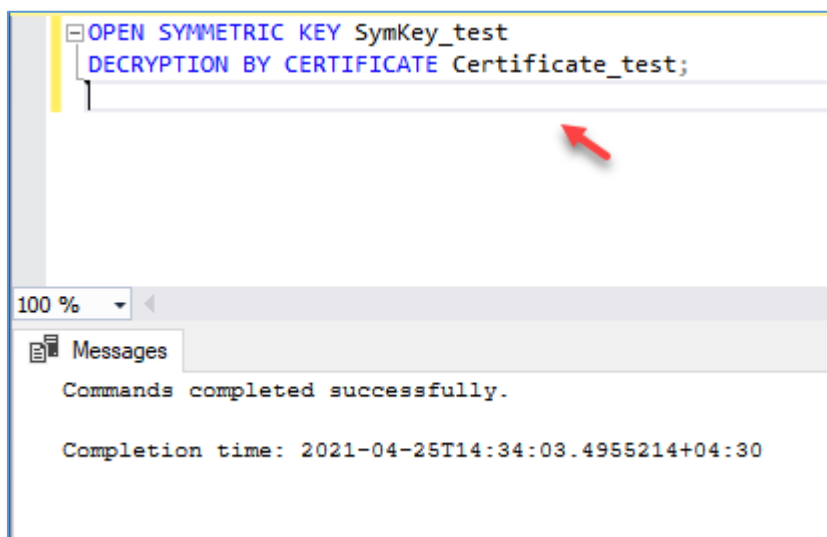


شکل ۴-۴۱ ایجاد ستون در جدول

در ادامه می‌خواهیم کار اصلی را انجام دهیم، یعنی اینکه داده‌هایی که در ستون BankACCCNumber_encrypt که در بالا ایجاد کردیم قرار می‌گیرند را رمزنگاری کنیم.

پس برای این کار باید از همان کلید متقارن و گواهینامه قبلی که ایجاد کردیم استفاده کنیم، با دستور زیر کلید و گواهینامه را فعال می‌کنیم که در شکل ۴-۴۲ این موضوع مشخص شده است.

```
OPEN SYMMETRIC KEY SymKey_test
DECRYPTION BY CERTIFICATE Certificate_test;
```



شکل ۴-۴۲ بازکردن کلید و گواهینامه

در ادامه کار اطلاعات موجود در ستون BankACCCNumber را درون ستون جدیدی که با نام BankACCCNumber_encrypt است قرار می‌دهیم البته با استفاده از کلید متقارن که ایجاد کردیم این کار را انجام می‌دهیم.

```
UPDATE CustomerData.dbo.CustomerInfo
SET BankACCCNumber_encrypt = EncryptByKey (Key_GUID('SymKey_test'),
BankACCCNumber)
FROM CustomerData.dbo.CustomerInfo;
GO
```

در ادامه باید کلید متقارن که باز کردید را ببندید که باید از این دستور استفاده کنید:

```
CLOSE SYMMETRIC KEY SymKey_test;
GO
```

حالا می‌توانیم با استفاده از دستور زیر خروجی کار را مشاهده کنیم:

```
SELECT *
FROM CustomerInfo;
```

همان‌طور که در شکل ۴-۴۳ مشاهده می‌کنید ستون جدید ایجاد شده و اطلاعاتی که داخل آن قرار گرفته‌اند رمزنگاری شده و کسی نمی‌تواند از اطلاعات را مشاهده کند.

	CustID	CustName	BankACCNumber	BankACCNumber_encrypt
1	1	Rajendra	11111111	0x0086F75FFB2DD14E80896EFEDA05BB0C0200000060F65A2...
2	2	Manoj	22222222	0x0086F75FFB2DD14E80896EFEDA05BB0C0200000073862D0...
3	3	Shyam	33333333	0x0086F75FFB2DD14E80896EFEDA05BB0C02000000DB7811B...
4	4	Akshita	44444444	0x0086F75FFB2DD14E80896EFEDA05BB0C02000000D117DFB...
5	5	Kashish	55555555	0x0086F75FFB2DD14E80896EFEDA05BB0C0200000003EA7D0...

شکل ۴-۴۳ نمایش جدول

تا اینجا توانستیم یک ستون از جدول را رمزگذاری کنیم تا کسی نتواند اطلاعات آن را مشاهده کند، اما اگر بخواهیم این اطلاعات را مشاهده کنیم یعنی رمزگشایی یا همان Decrypt کنیم چه کاری باید انجام دهیم؟ برای این کار باید از دستور DecryptByKey استفاده کنیم، برای تست این موضوع به دستورات زیر دقت کنید: اولین کاری که انجام می‌دهیم کلید متقارن و گواهی‌نامه مورد نظر را صدا می‌زنیم:

```
OPEN SYMMETRIC KEY SymKey_test
DECRYPTION BY CERTIFICATE Certificate_test;
```

بعد از بازکردن موارد مورد نظر باید از دستورات زیر استفاده کنید:

```
SELECT CustID, CustName, BankACCNumber_encrypt AS 'Encrypted data',
       CONVERT(varchar, DecryptByKey(BankACCNumber_encrypt)) AS 'Decrypted Bank
account number'
FROM CustomerData.dbo.CustomerInfo;
```

در دستورات بالا ستون‌های جدول مورد نظر انتخاب شده‌اند ولی در ستون BankACCNumber_encrypt با استفاده از دستور AS به ستون Encrypted data تغییر نام پیدا کرد و همین ستون با استفاده از دستور DecryptByKey اطلاعات آن به ستون جدید با نام Decrypted Bank account number رمزگشایی شد که این موضوع را در شکل ۴-۴۴ مشاهده می‌کنید:

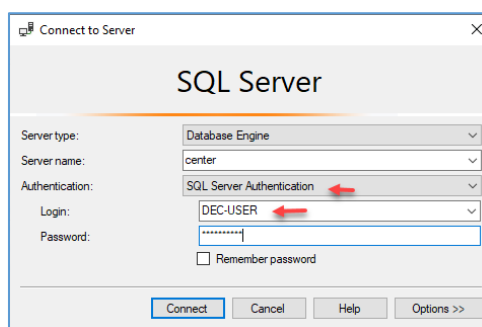
CustID	CustName	Encrypted data	Decrypted Bank account number
1	Rajendra	0x0086F75FFB2DD14E80896EFEDA05BB0C0200000060F65A2...	11111111
2	Manoj	0x0086F75FFB2DD14E80896EFEDA05BB0C0200000073862D0...	22222222
3	Shyam	0x0086F75FFB2DD14E80896EFEDA05BB0C02000000DB7811B...	33333333
4	Akshita	0x0086F75FFB2DD14E80896EFEDA05BB0C02000000D117DFB...	44444444
5	Kashish	0x0086F75FFB2DD14E80896EFEDA05BB0C0200000003EA7D0...	55555555

شکل ۴-۴ رمزگشایی ستون

بعد از رمزگذاری و رمزگشایی، حالا می‌خواهیم تست بگیریم که هر کاربری با هر دسترسی می‌تواند این عملیات را انجام دهد یا نه، برای تست این موضوع با استفاده از دستورات زیر یک کاربر جدید با نام DEC-USER ایجاد می‌کنیم که به دیتابیس CustomerData دسترسی db_datareader دارد یعنی فقط می‌تواند اطلاعات را بخواند.

```
USE [master]
GO
CREATE LOGIN [DEC-USER] WITH PASSWORD=N'Test@12345', DEFAULT_DATABASE=[CustomerData],
CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
USE [CustomerData]
GO
CREATE USER [DEC-USER] FOR LOGIN [DEC-USER]
GO
USE [CustomerData]
GO
ALTER ROLE [db_datareader] ADD MEMBER [DEC-USER]
GO
```

بعد از ایجاد کاربر DEC-USER باید با این کاربر به‌مانند شکل ۴-۵ با کاربر جدید وارد SQL شوید.



شکل ۴-۵ ورود به SQL

بعد از ورود دستور زیر را در New Query اجرا کنید:

```
OPEN SYMMETRIC KEY SymKey_test
DECRYPTION BY CERTIFICATE Certificate_test;

SELECT CustID, CustName, BankACNumber_encrypt AS 'Encrypted data',
CONVERT(varchar, DecryptByKey(BankACNumber_encrypt)) AS 'Decrypted Bank account
number'
FROM CustomerData.dbo.CustomerInfo;
```

با اجرای دستور بالا با خطای شکل ۴-۴۶ مواجه خواهید شد که اشاره به دسترسی نداشتن کاربر مورد نظر دارد.

```

OPEN SYMMETRIC KEY SymKey_test
DECRYPTION BY CERTIFICATE Certificate_test;

SELECT CustID, CustName, BankACNumber_encrypt AS 'Encrypted data',
CONVERT(varchar, DecryptByKey(BankACNumber_encrypt)) AS 'Decrypted Bank account n
FROM CustomerData.dbo.CustomerInfo;

```

00 %

Results Messages

CustID	CustName	Encrypted data	Decrypted Bank account
Msg 15151, Level 16, State 1, Line 1			
Cannot find the symmetric key 'SymKey_test', because it does not exist or you do not have permission.			
(5 rows affected)			
Completion time: 2020-01-04T21:19:00.1803149+05:30			

شکل ۴-۴۶ تست دسترسی کاربر

برای حل این مشکل باید به کاربر مورد نظر دسترسی لازم را بدهیم تا بتواند هم به کلید متقارن و هم به گواهینامه مورد نظر دسترسی داشته باشد.

```

GRANT VIEW DEFINITION ON SYMMETRIC KEY::SymKey_test TO "DEC-USER";
GO
GRANT VIEW DEFINITION ON Certificate::[Certificate_test] TO "DEC-USER";
GO
GRANT CONTROL ON Certificate::[Certificate_test] TO "DEC-USER";

```

نتیجه‌ی دستورات را در شکل ۴-۴۷ مشاهده می‌کنید.

```

GRANT VIEW DEFINITION ON SYMMETRIC KEY::SymKey_test TO "DEC-USER";
GO
GRANT VIEW DEFINITION ON Certificate::[Certificate_test] TO "DEC-USER";
GO
GRANT CONTROL ON Certificate::[Certificate_test] TO "DEC-USER";

```

00 %

Messages

Commands completed successfully.

Completion time: 2021-04-25T16:22:30.4064488+04:30

شکل ۴-۴۷ افزایش دسترسی کاربر

بعد از دادن دسترسی‌های لازم دوباره دستورات بالا را در این قسمت وارد و اجرا می‌کنیم:

```

OPEN SYMMETRIC KEY SymKey_test
DECRYPTION BY CERTIFICATE Certificate_test;

```

```
SELECT CustID, CustName, BankACCNumer_encrypt AS 'Encrypted data',
CONVERT(varchar, DecryptByKey(BankACCNumer_encrypt)) AS 'Decrypted Bank account
number'
FROM CustomerData.dbo.CustomerInfo;
```

همانطور که در شکل ۴-۴۸ مشاهده می‌کنید دستور مورد نظر با موفقیت اجرا شده و خروجی به نمایش گذاشته شده است.

The screenshot displays a SQL query window with the following script:

```
OPEN SYMMETRIC KEY SymKey_test
DECRYPTION BY CERTIFICATE Certificate_test;
SELECT CustID, CustName, BankACCNumer_encrypt AS 'Encrypted data',
CONVERT(varchar, DecryptByKey(BankACCNumer_encrypt)) AS 'Decrypted Bank account number'
FROM CustomerData.dbo.CustomerInfo;
CLOSE SYMMETRIC KEY SymKey_test;
GO
```

The Results pane shows the following data:

CustID	CustName	Encrypted data	Decrypted Bank account number
1	Rajendra	0x008A02FB717BE9479FBD4FEF542A8E9C020000004E6E5D...	11111111
2	Manoj	0x008A02FB717BE9479FBD4FEF542A8E9C02000000E7585B1...	22222222
3	Shyam	0x008A02FB717BE9479FBD4FEF542A8E9C02000000058041...	33333333
4	Akshita	0x008A02FB717BE9479FBD4FEF542A8E9C0200000074B7F0E...	44444444
5	Kashish	0x008A02FB717BE9479FBD4FEF542A8E9C02000000080D5C6...	55555555

شکل ۴-۴۸ تست دسترسی کاربر

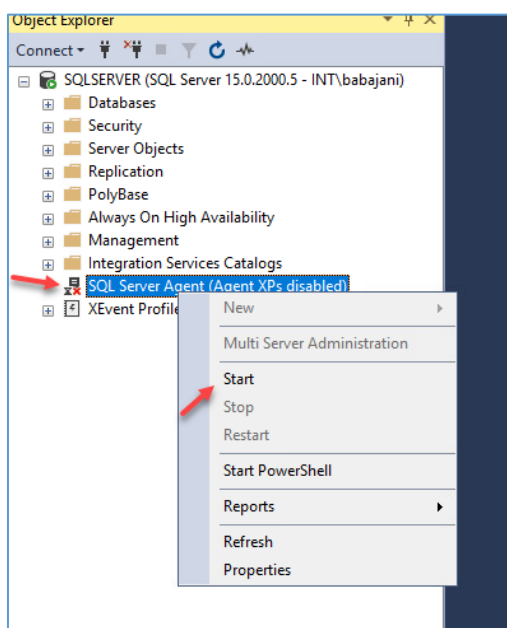
فصل پنجم

پشتیبان‌گیری و بازیابی

می‌توان یکی از مهم‌ترین بخش‌های یک سیستم نرم‌افزاری را بخش پشتیبانی و بازیابی آن بیان کرد. در سازمان‌های بزرگ مانند بانک‌ها در یک ثانیه چندین هزار رکورد در دیتابیس و جداول ثبت می‌شود و اگر چنانچه از دیتابیس مورد نظر پشتیبان نداشته باشید با ازدست‌رفتن اطلاعات اصلی دیگر نمی‌توانید به آن اطلاعات دسترسی داشته باشید پس باید مسیر را دقیقاً مشخص کرد تا بتوانیم در سریع‌ترین زمان ممکن اطلاعات را برگردانیم. روش‌های پشتیبان‌گیری از دیتابیس بسیار زیاد است که می‌توانید از خود نرم‌افزار SQL استفاده کنید و یا اینکه از نرم‌افزارهای جانبی استفاده کنید که در اینجا همه‌ی آنها را بررسی می‌کنیم.

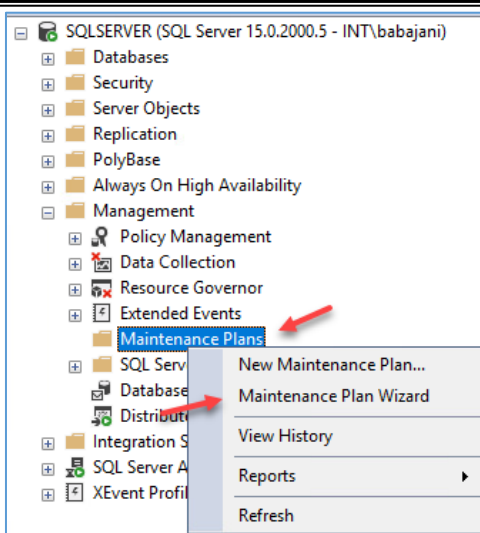
۵-۱ پشتیبان‌گیری از طریق نرم‌افزار SQL

برای شروع کار باید سرویس SQL Server Agent را فعال کنیم، برای این کار باید وارد SQL شوید و به‌مانند شکل ۵-۱ بر روی سرویس Agent کلیک راست کنید و گزینه‌ی Start را انتخاب کنید، بعد از آن پنجره‌ای باز خواهد شد که باید بر روی Yes کلیک کنید.



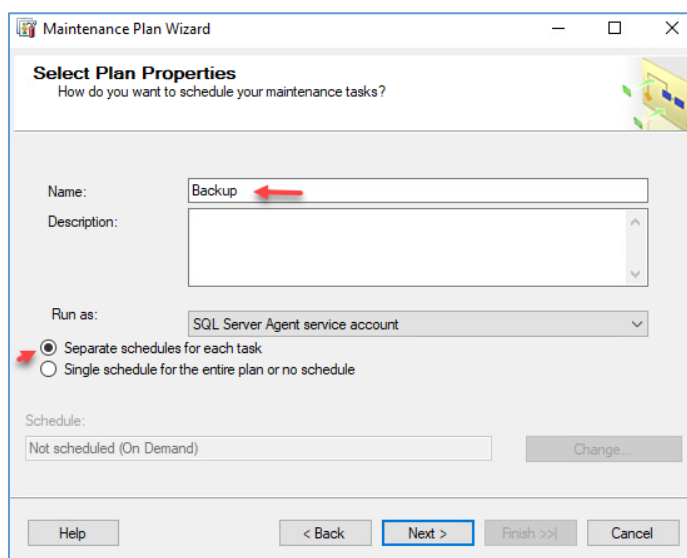
شکل ۵-۱

در ادامه می‌خواهیم ابزارهای Database Maintenance را با هم بررسی کنیم، برای اینکه سرویس Backup را فعال کنیم باید به‌مانند شکل ۵-۲ وارد قسمت Management شوید و بر روی Maintenance Plans کلیک راست کنید و گزینه‌ی Maintenance Plan Wizard را انتخاب کنید.



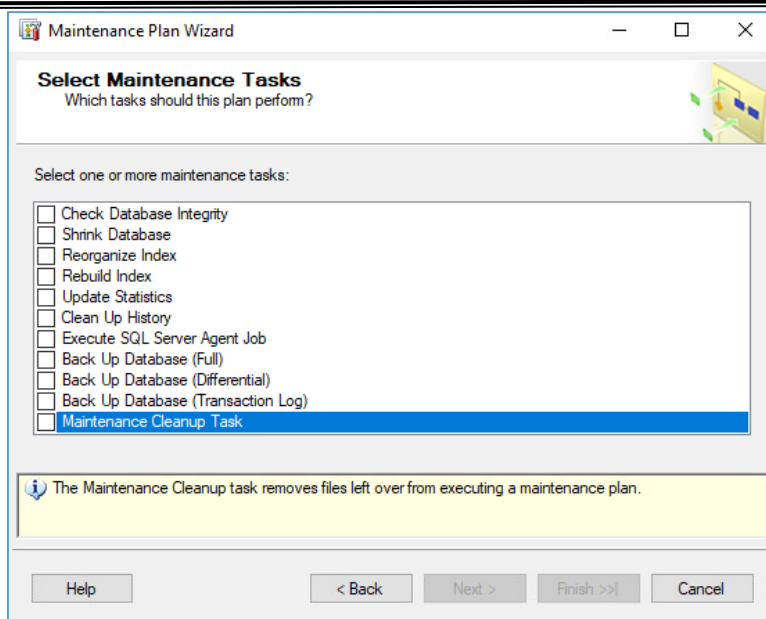
شکل ۵-۲

در شکل ۵-۳ باید یک نام به دلخواه وارد کنید و در قسمت زیری آن برای زمان بندی فعلاً گزینه‌ی Separate را انتخاب کنید.



شکل ۵-۳

در شکل ۵-۴ گزینه‌های مختلفی را مشاهده می‌کنید که هر کدام برای یک کار طراحی شده‌اند که در زیر آنها را بررسی می‌کنیم.



شکل ۴-۵

۱- Check Database Integrity

این گزینه برای این است که بفهمیم یک دیتابیس مشکلی ندارد و می‌توانیم از آن پشتیبان تهیه کنیم که این موضوع واقعاً برای پشتیبان‌گیری مهم است اگر چنانچه دیتابیس مورد نظر مشکل داشته باشد حتی بعد از انجام پشتیبان‌گیری نمی‌توانید از آن دیتابیس در آینده استفاده کنید، پس حتماً باید از این گزینه استفاده کنید.

۲- Shrink Database

زمانی که یک دیتابیس ایجاد می‌شود دو فایل در محل ذخیره‌سازی ایجاد می‌شود که یکی فایل اصلی و دیگری فایل log است، این فایل Log بسته به نوع کار در طی زمان حجم آن افزایش پیدا خواهد کرد که باید توسط این گزینه حجم آن را کاهش دهید تا حجم اضافه در خروجی پشتیبان‌گیری ایجاد نشود.

۳- Rebuild index و Reorganize index

زمانی که یا دستورت مختلف بر روی دیتابیس خودکار می‌کنید استفاده از این دستورات مانند Insert, Delete و... به مرور زمان باعث ایجاد Fragmentation یا همان پارگی می‌شوند و همین امر باعث می‌شود که اطلاعات به‌درستی در هارددیسک یا محل ذخیره‌سازی قرار نگیرند به طور ساده‌تر باید گفت که مثلاً از یک محل ذخیره‌سازی با حجم ۱ گیگابایت داشته باشید، زمانی که Fragmentation یا پارگی ایجاد شود یک دیتابیس ۵۰۰ مگابایت در یک این فضای یک گیگابایت ذخیره خواهد شد و استفاده درستی از آن نخواهد شد و مهم‌ترین مشکلی که پارگی ایجاد می‌کند این است که به شدت سرعت دسترسی و استفاده از دیتابیس را کاهش می‌دهد.

۴- Update Statistics

این گزینه برای بهبود عملکرد پرس‌وجو طراحی شده است و باعث به‌روزرسانی اطلاعات برای انجام پرس‌وجو خواهد شد.

۵- Clean Up History

زمانی که دیتابیس‌های مختلفی را در طول زمان ایجاد می‌کنید یک سری داده در جدول سیستم ذخیره می‌کند و این داده‌ها به‌مرور زمان منسوخ و از کار افتاده خواهند شد و باید با دستوری آن اطلاعات قدیمی را از دیتابیس پاک کرد، البته اگر این داده‌ها را حذف نکنید به‌مرور زمان فضای ذخیره‌سازی را پر خواهند کرد.

۶- execute sql server agent job

این دستور برای اجرای کارهایی است که در agent ایجاد کردید.

۷- Backup Database (FULL)

برای اینکه به‌طور کامل از دیتابیس‌های خود پشتیبان تهیه کنیم، باید از این گزینه استفاده کنیم، داشتن حداقل یک پشتیبان Full برای هر یک از دیتابیس‌ها ضروری است و اگر Full وجود نداشته باشد شما توانایی برگرداندن دیتابیس خراب شده را نخواهید داشت.

۸- Backup Database (Differential)

این روش یک روش برای پشتیبان‌گیری از تغییرات است، مثلاً اگر یک پشتیبان Full تهیه کرده باشد با حجم ۱۰ گیگابایت در موقعی که از پشتیبان Differential استفاده کنید حجم آن بسیار کم خواهد شد چون فقط از تغییراتی که بعد از Full ایجاد شده پشتیبان تهیه می‌شود، البته اگر تنها از روش Differential استفاده کنید در اولین باری که از دیتابیس پشتیبان تهیه می‌کند از تمام اطلاعات پشتیبان می‌گیرد.

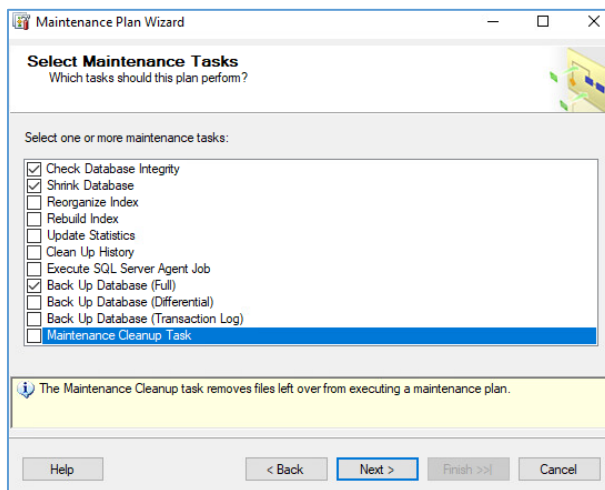
۹- Backup Database (Transaction Log)

زمانی که یک دیتابیس ایجاد می‌کنید، باید حالت Recovery Model آن را مشخص کنید، اگر بر روی حالت Full قرار داشته باشد از دیتابیس مورد نظر یک فایل Log هم گرفته می‌شود که قبلاً این موضوع را توضیح دادیم، اگر از این روش پشتیبان‌گیری استفاده کنید از داده‌های log هم پشتیبان تهیه خواهد شد و یک مزیت آن این است که بعد از انجام این نوع پشتیبان‌گیری حجم فایل log دیتابیس کاهش پیدا خواهد کرد و دیگر نیاز نیست خودتان آن را کم کنید.

۱۰- Maintenance Cleanup Task

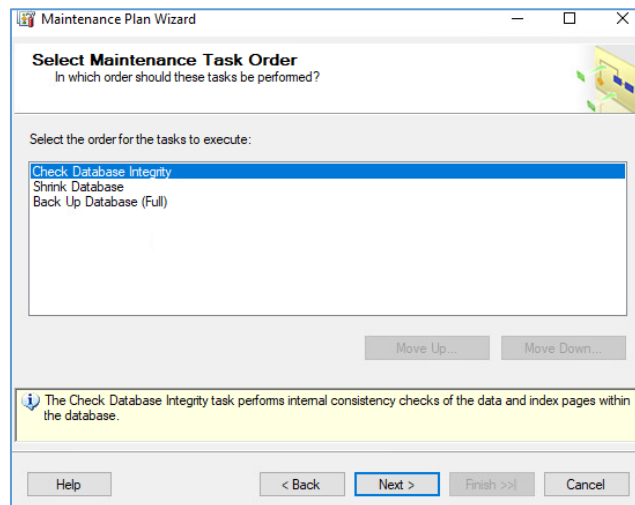
این گزینه برای حذف پرونده‌های مربوط به برنامه‌های نگهداری مانند پرونده‌های دیتابیس و... کاربرد دارد.

بعد از بررسی گزینه‌های مورد نظر برای تست کار به‌مانند شکل ۵-۵ سه گزینه‌ی مورد نظر را انتخاب کنید و بر روی Next کلیک کنید.



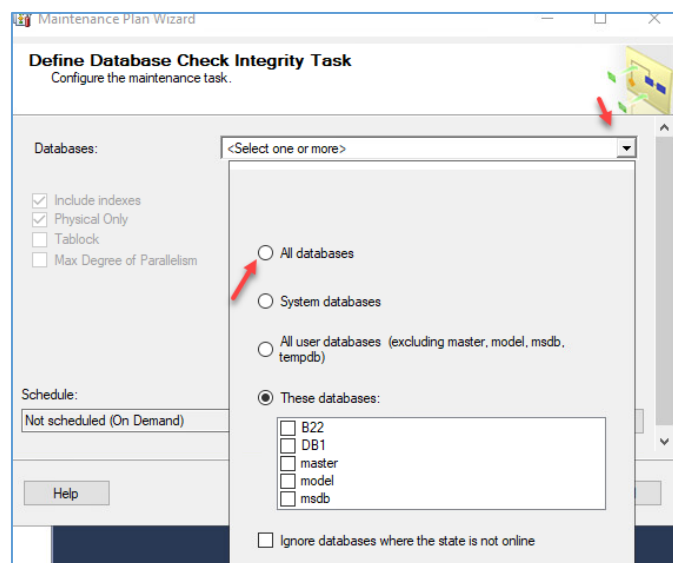
شکل ۵-۵

همان‌طور که در شکل ۵-۶ مشاهده می‌کنید سه گزینه‌ی مورد نظر انتخاب شده است و برای ادامه بر روی Next کلیک کنید.



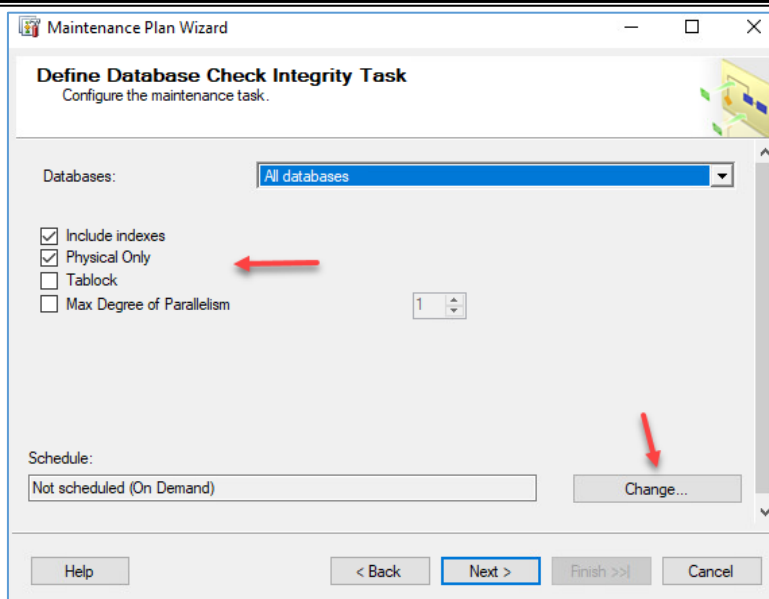
شکل ۵-۶

به‌مانند شکل ۵-۷ باید مشخص کنید که چه دیتابسی باید انتخاب شود تا عملیات Check Integrity بر روی آن اعمال شود که می‌توانید با انتخاب گزینه‌ی All databases همه‌ی آنها را انتخاب کنید و یا System Database و یا خودتان انتخاب کنید که بهتر است گزینه‌ی All databases را انتخاب کنید.



شکل ۵-۷

بعد از انتخاب All database چند گزینه را در شکل ۵-۸ مشاهده می‌کنید که در ادامه آنها را معرفی می‌کنیم.



شکل ۵-۱

Include Indexes

این گزینه برای بررسی درستی و یکپارچگی ایندکس‌های جداول در دیتابیس است که انجام می‌شود.

Physical Only

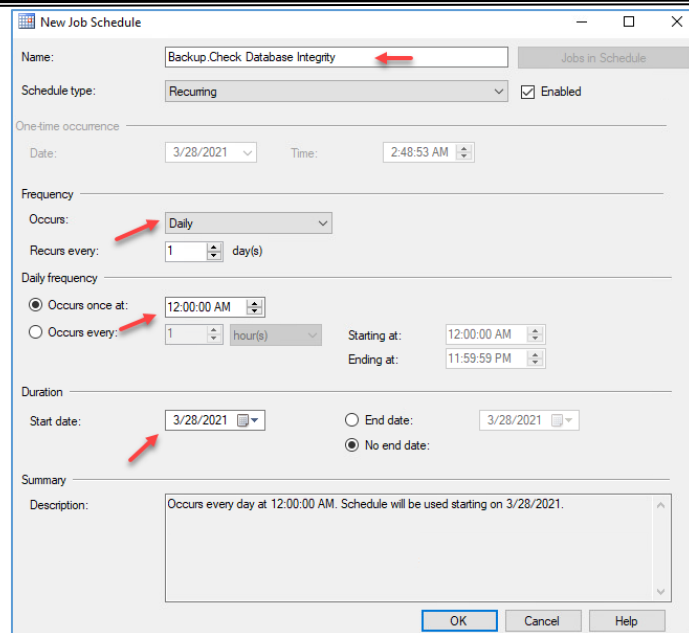
این گزینه بعد از اجرا فقط ساختار Pageها و سرصفحه‌های رکوردهای موجود را بررسی می‌کند.

Tablock

با فعال کردن این گزینه دیتابیس مورد نظر برای مدت‌زمان کوتاهی قفل شده و دستور بررسی سلامت دیتابیس یعنی DBCC CHECKDB فعال می‌شود و بعد از اجرا دستور دیتابیس مورد نظر از قفل باز می‌شود، بکی از ویژگی‌های این روش بعد از اجرا باعث افزایش سرعت بررسی دیتابیس خواهد شد.

Max Degree Of Parallelism

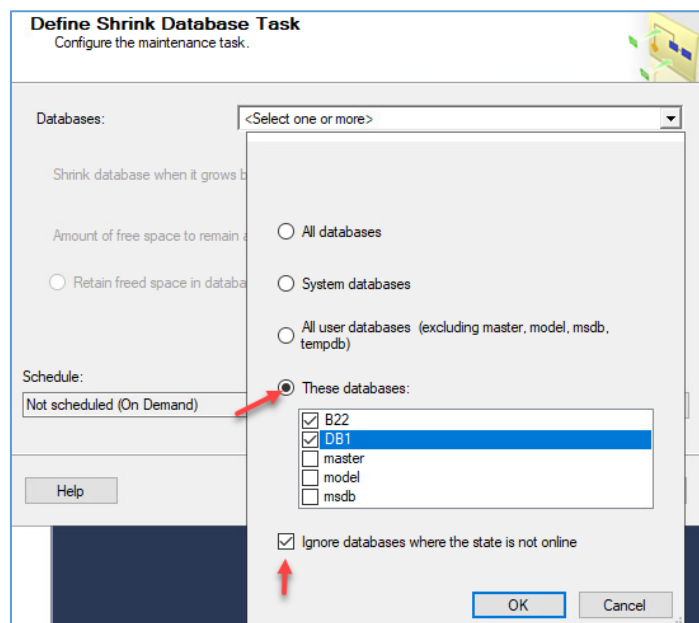
زمانی که از دستور بررسی سلامت دیتابیس یعنی DBCC CHECKDB استفاده می‌کنید SQL از تمام توان CPU استفاده خواهد کرد، با این گزینه می‌توانید مشخص کنید که در زمان اجرا از چند هسته‌ی CPU استفاده شود. اگر به پایین شکل ۵-۹ توجه کنید، شما می‌توانید برای این وظیفه یک زمان‌بندی مشخص ایجاد کنید، برای این کار بر روی Change کلیک کنید.



شکل ۹-۵

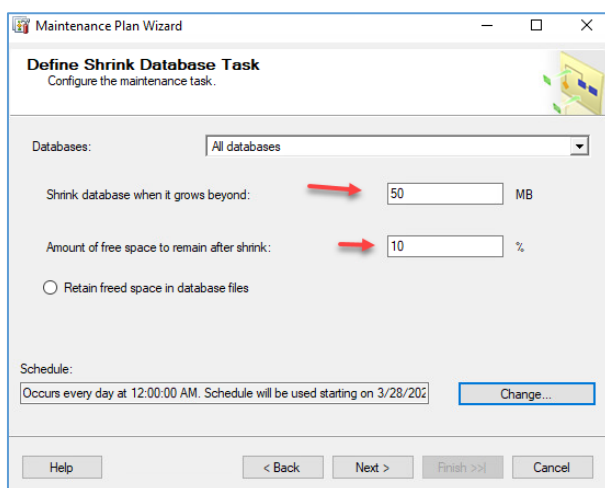
در شکل ۹-۵ باید یک نام برای زمان‌بندی خود وارد کنید و بعد باید مشخص کنید که این وظیفه در چه زمان‌هایی اجرا شود مثلاً می‌توانید هفتگی انتخاب کنید یا به‌صورت روزانه که در اینجا روزانه را انتخاب می‌کنیم، در قسمت سوم می‌توانیم ساعت اجرا این وظیفه را مشخص کنیم که ساعت ۱۲ بامداد انتخاب شده است و در قسمت آخر می‌توانید مشخص کنید که این زمان‌بندی از چه تاریخی اجرا شود که به‌صورت پیش‌فرض تاریخ امروز انتخاب خواهد شد، بر روی OK کلیک کنید.

در شکل ۱۰-۵ باید مشخص کنید که چه دیتابیس‌هایی را می‌خواهید بر روی آن عملیات shrink انجام دهید که دو دیتابیس که با هم ایجاد کردیم را انتخاب می‌کنیم، اگر در پایین صفحه تک‌گزینه‌ی ignore.. را انتخاب کنید در زمان Shrink از دیتابیس‌هایی که آفلاین هستند صرف‌نظر خواهد شد.



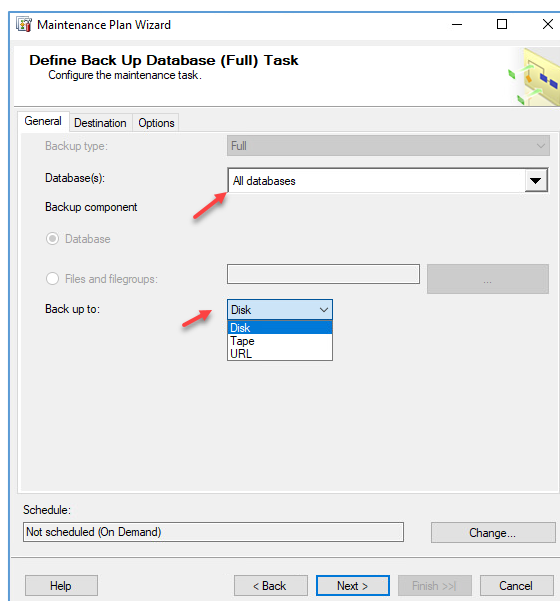
شکل ۱۰-۵

بعد از انتخاب دیتابیس به‌مانند شکل ۱۱-۵ دو گزینه وجود دارد، گزینه‌ی اول عدد ۵۰ را نشان می‌دهد و این نشان‌دهنده‌ی این است که روی دیتابیس‌هایی که بیشتر از ۵۰ مگابایت حجم داشته باشند عملیات Shrink انجام خواهد شد و اگر کمتر از این مقدار باشد عملیات Shrink انجام نخواهد شد، در قسمت بعدی عدد ۱۰ درصد را مشاهده می‌کنید که نشان‌دهنده‌ی این است که چند درصد از فضای دیتابیس بعد از Shrink خالی بماند، گزینه‌ی Retain هم اگر انتخاب شود فقط صفحه‌های انتهایی را به ابتدای فایل انتقال خواهد داد و با صفحات میانی کاری ندارد، در آخر صفحه هم می‌توانید زمان‌بندی اجرا را به‌مانند قبل مشخص کنید.



شکل ۱۱-۵

در شکل زیر نحوه انجام Full backups را باید مشخص کنید، برای این کار باید دیتابیس مورد نظر خود را انتخاب کنید که بهترین حالت این است که از گزینه‌ی All Databases استفاده کنید تا همه‌ی دیتابیس‌ها اعم از سیستمی و کاربر را تخصیص دهد. در پایین شکل ۱۲-۵ نوع ذخیره‌سازی فایل را باید مشخص کنید که دارای سه حالت است که در زیر بررسی می‌کنیم.



شکل ۱۲-۵

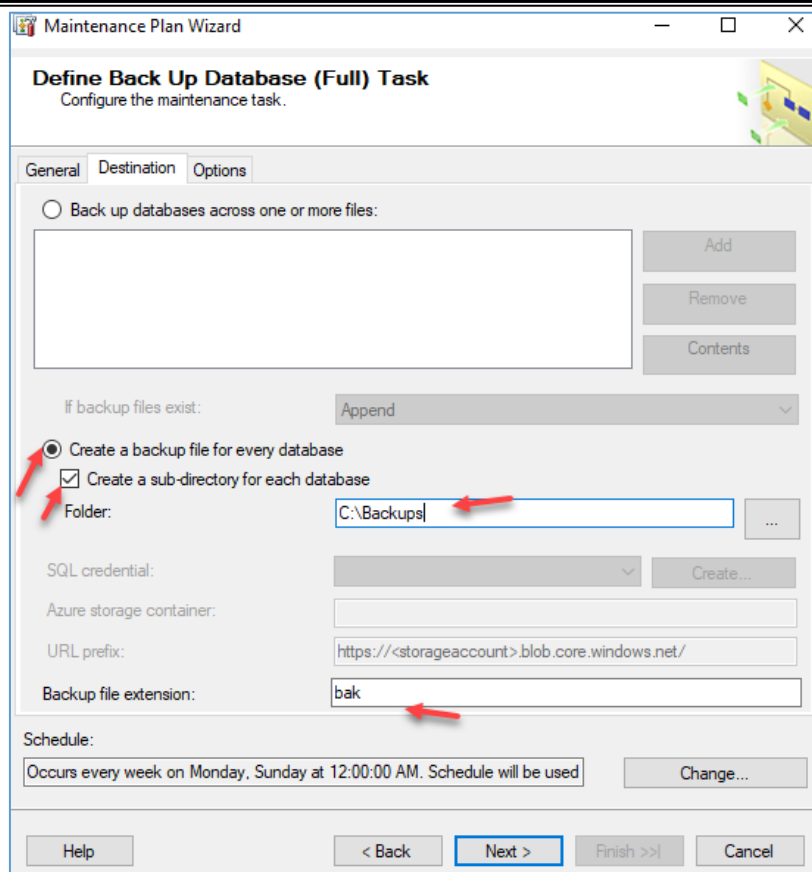
گزینه‌ی Disk برای ذخیره کردن فایل‌های پشتیبان در هارددیسک سیستم خود و یا در یک آدرس شبکه است که در ادامه بررسی می‌کنیم.

گزینه‌ی tape برای ذخیره اطلاعات بر روی نوار مغناطیسی می‌شد که یک روش بسیار مقرون‌به‌صرفه است، هرچند که بسیار قدیمی شده ولی به علت ظرفیت بالای آن و قیمت پایین آن حتی در شرکت‌های بزرگ هم در حال استفاده است.

گزینه‌ی URL برای ذخیره کردن اطلاعات در یک آدرس تحت وب طراحی شده است مانند آدرس زیر:
'https://mystorage.blob.core.windows.net/mycontainer/TestDbBackupSetNumber2_0.bak'
در ادامه کار بعد از انتخاب گزینه‌ی Disk در پایین صفحه می‌توانید زمان‌بندی مورد نظر خود را مشخص کنید، در شکل ۵-۱۳ باید زمان‌بندی انجام Full Backup را مشخص کنید، برای انجام Full Backup بهتر است گزینه‌ی هفتگی را انتخاب کنید دلیل آن هم این است که اگر تعداد زیادی دیتابیس با حجم بالا داشته باشید انجام Full Backup می‌تواند حجم زیادی را از حافظه‌ی شما پر کند و بهترین کار این است که به‌صورت هفتگی این کار را انجام دهید و به Differential Backup را به‌صورت روزانه انجام دهید.

شکل ۵-۱۳

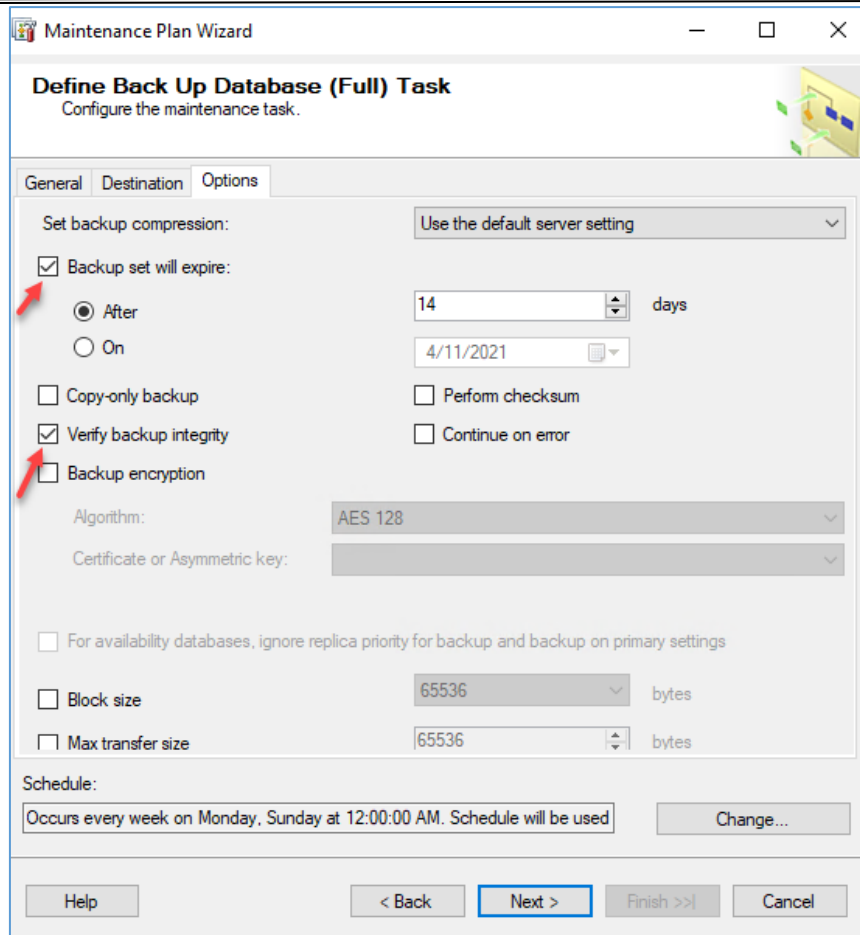
در ادامه کار به‌مانند شکل ۵-۱۴ وارد تب Destination شوید و در این تب باید مشخص کنید که فایل‌های Backup در کدام مسیر ذخیره شود که در اینجا مسیر C:\Backups انتخاب شده است توجه داشته باشید برای اینکه نرم‌افزار دسترسی ایجاد SubFolder را داشته باشید بهتر است تیک گزینه‌ی Create a sub-directory را انتخاب کنید، در پایین صفحه می‌توانید پسوند فایل Backup خود را به‌دلخواه وارد کنید که به‌صورت پیش‌فرض پسوند bak وارد شده است.



شکل ۱۴-۵

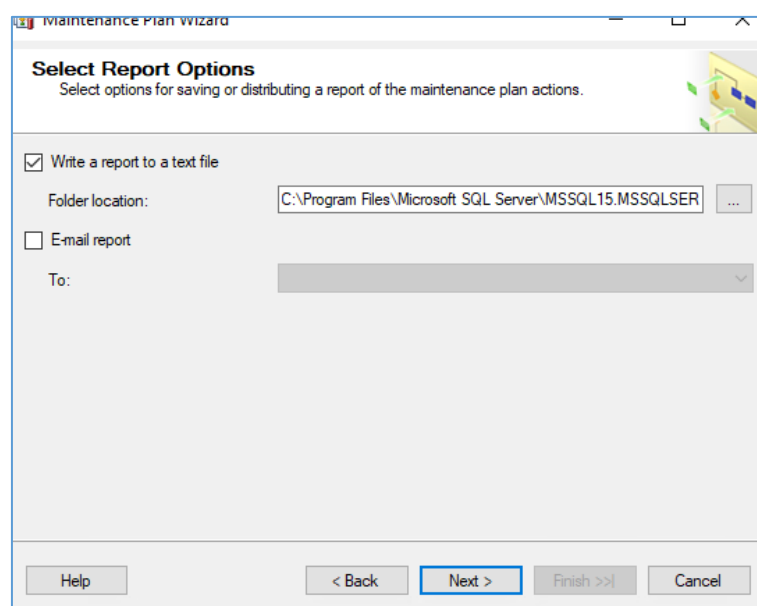
در ادامه به مانند شکل ۱۵-۵ وارد تب Options شوید، در این تب می‌توانید با انتخاب گزینه‌ی Backup set will Expore مشخص کنید که فایل‌های Backup بعد از مدت مشخص شده‌ی ۱۴ روز از روی هارد پاک شوند و فایل Backup جدید جایگزین آن شود.

یکی از مهم‌ترین قسمت‌هایی که باید مراقب آن باشد این است که تیک گزینه‌ی Verify Backup Integrity را انتخاب کنید، این گزینه بعد از انجام Backup فایل Backup را تست می‌کند تا سلام باشد و بعد از آن پردازش به پایان می‌رسد، در جاهایی دیده شده که فایل‌های Backup قابل بازگردانی نبودن و مدیر شبکه با مشکل بزرگی روبرو شده است، پس حتماً تیک این گزینه را بزنید، البته اگر حجم Backup زیاد باشد مقدار زمان انجام عملیات بیشتر خواهد شد. گزینه‌ی Backup Encryption را اگر انتخاب کنید بر روی فایل‌های Backup یک رمز عبور به همراه هشینگ قدرتمند قرار می‌دهد تا کسی نتواند اطلاعات را به سرقت ببرد.



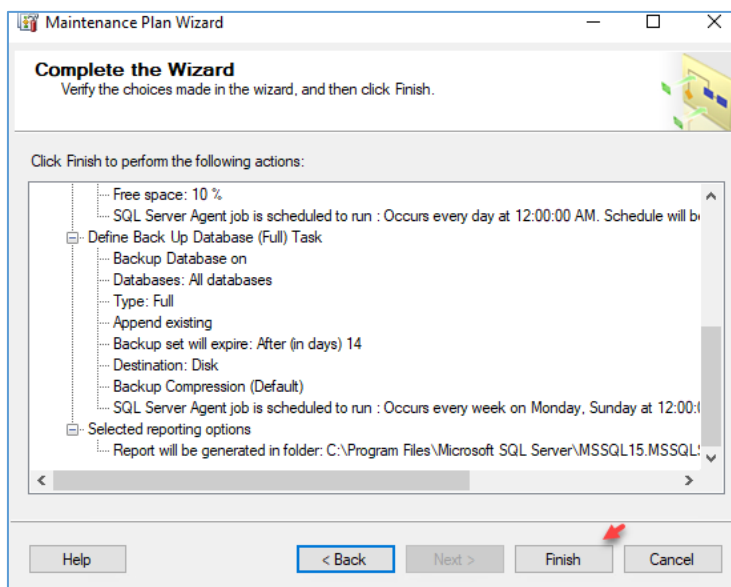
شکل ۵-۱۵

در ادامه بر روی Next کلیک کنید، در شکل ۵-۱۶ می‌توانید مشخص کنید که گزارش‌گیری از اطلاعات این کار در چه مسیری ذخیره شود، البته این فایل به صورت TXT است و اگر هم بخواهید به یک آدرس خاص ایمیل شود باید تیک گزینه‌ی E-mail report را انتخاب کنید.



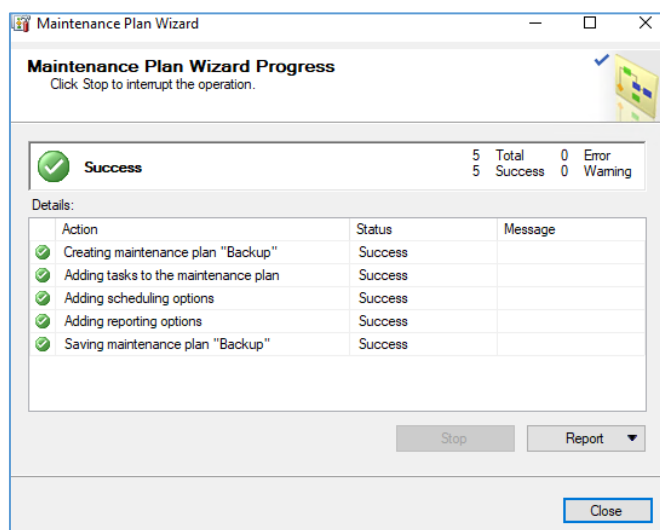
شکل ۵-۱۶

اگر اطلاعاتی در شکل ۵-۱۷ مورد قبول بود می‌توانید بر روی Finish کلیک کنید.



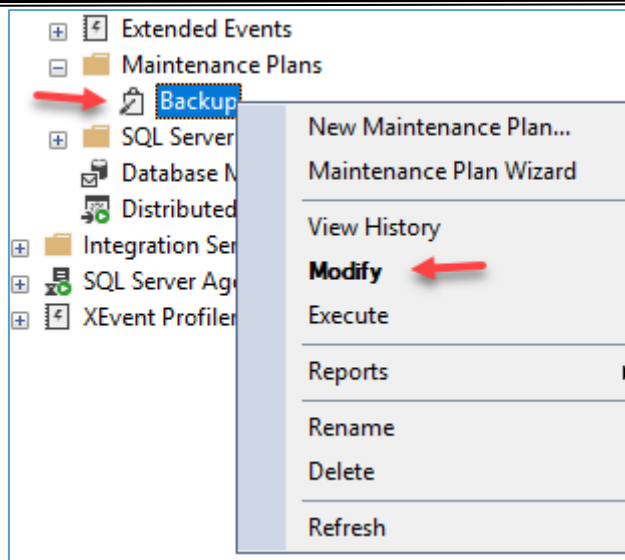
شکل ۵-۱۷

همان‌طور که در شکل ۵-۱۸ مشاهده می‌کنید اطلاعات به‌درستی تأیید و ایجاد شده است.



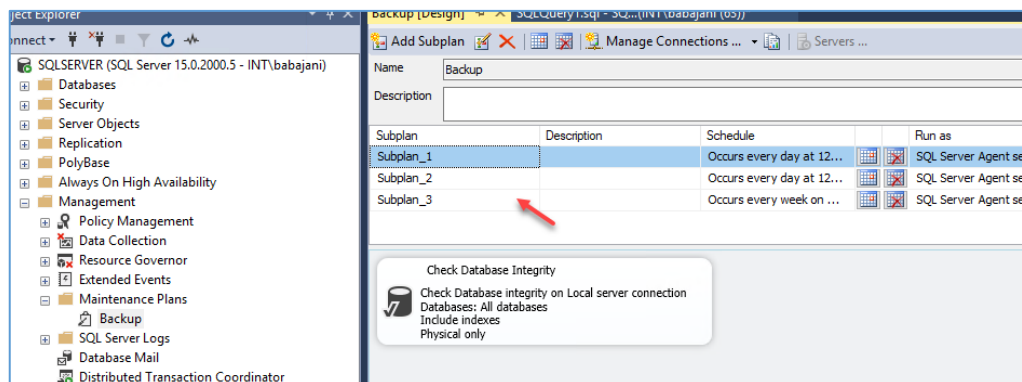
شکل ۵-۱۸

در شکل ۵-۱۹ Plan مورد نظر ایجاد شده است و برای اینکه بررسی کامل‌تری داشته باشیم بر روی آن کلیک راست کنید و گزینه‌ی Modify را انتخاب کنید.



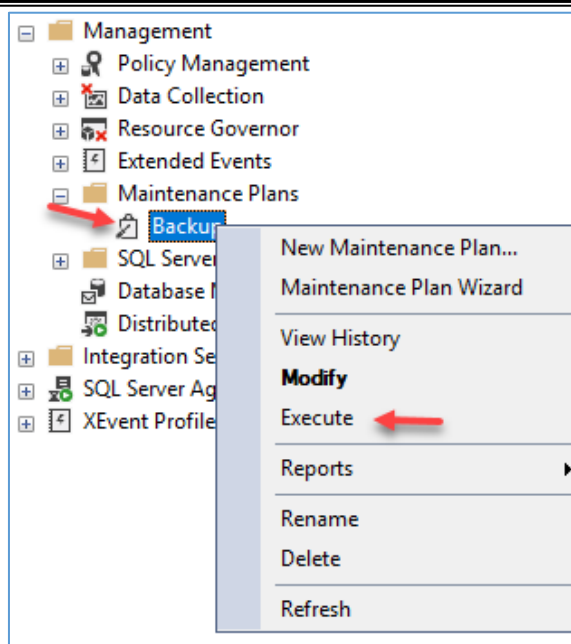
شکل ۵-۱۹

در شکل ۵-۲۰ هر سه قسمتی که در قسمت‌های قبل ایجاد کردیم را مشاهده می‌کنید اگر بر روی آنها کلیک کنید عملکرد آنها را مشاهده خواهید کرد، در جلوی آنها زمان‌بندی را می‌توانید تغییر یا حذف کنید و حتی می‌توانید نام هر یک از Ruleها را تغییر دهید.



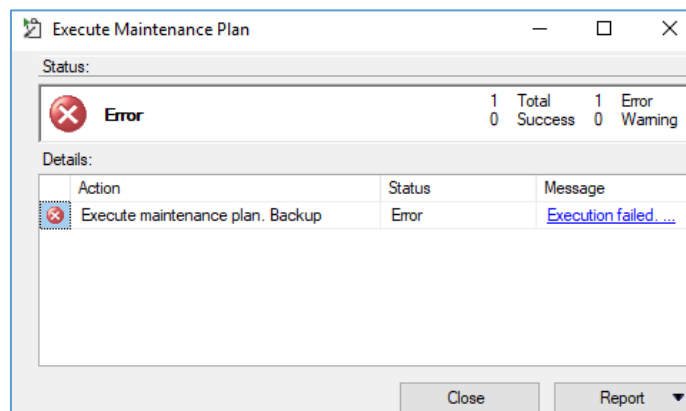
شکل ۵-۲۰

بعد از ایجاد Plan باید آن را اجرا کنیم، برای اجرا به‌مانند شکل ۵-۲۱ بر روی Plan مورد نظر کلیک راست کنید و گزینه‌ی Execute را انتخاب کنید.



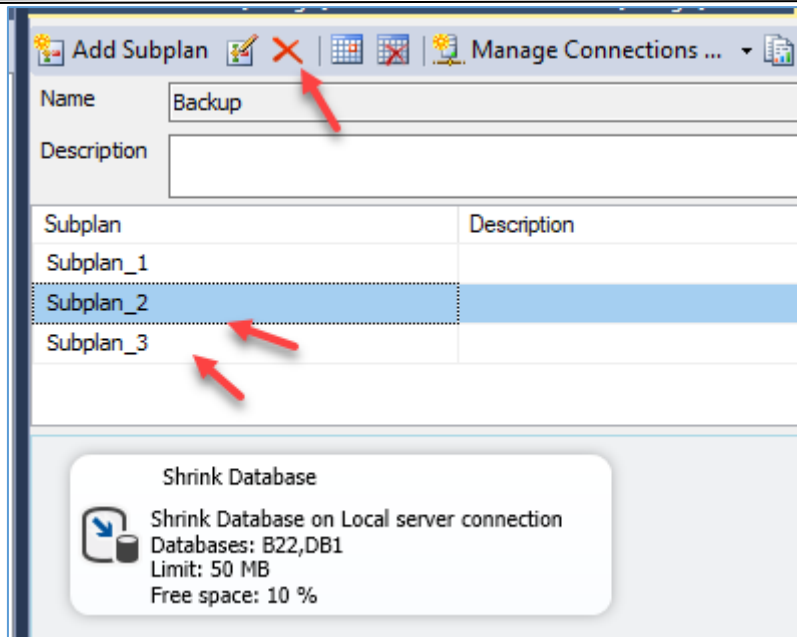
شکل ۵-۲۱

بعد از اجرا به احتمال زیاد با خطای شکل ۵-۲۲ مواجه خواهید شد که آن هم به این خاطر است که ما برای هر یک از قسمت‌ها یک زمان‌بندی تعریف کردیم و به خاطر همین این سه قسمت از هم جدا شدن، برای حل این مشکل باید آنها را در یک قسمت قرار دهیم.



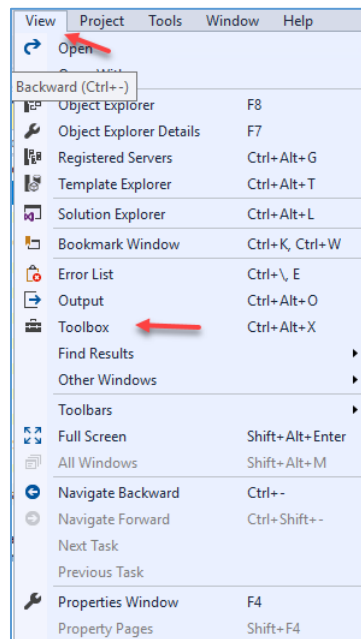
شکل ۵-۲۲

برای حل مشکل به مانند شکل ۵-۲۳ وارد Modify شوید و Subplan_2 و Subplan_3 را از لیست حذف کنید.



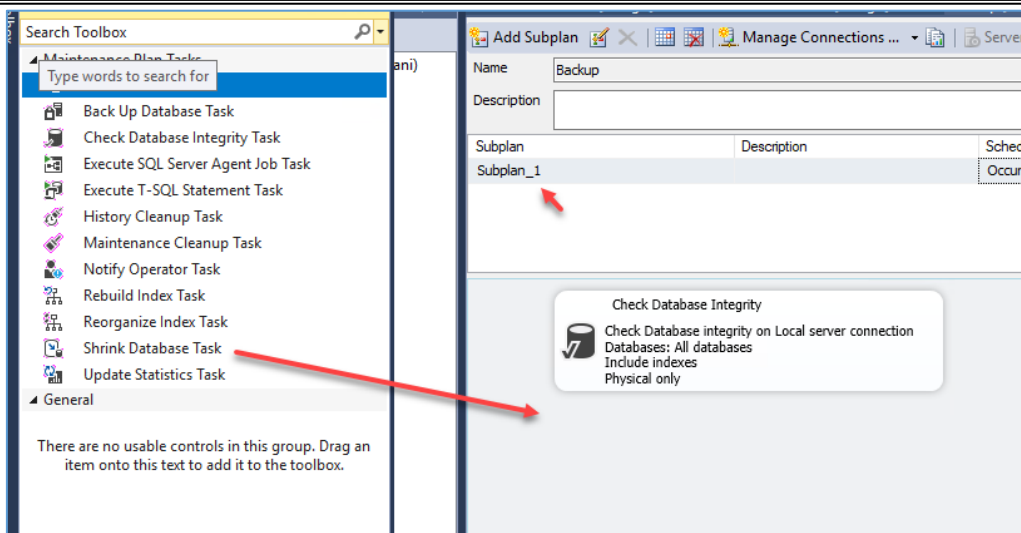
شکل ۵-۲۳

در ادامه به‌مانند شکل ۵-۲۴ وارد منوی View شوید و بر روی Toolbox کلیک کنید، البته از سمت چپ هم می‌توانستید Toolbox را اجرا کنید.



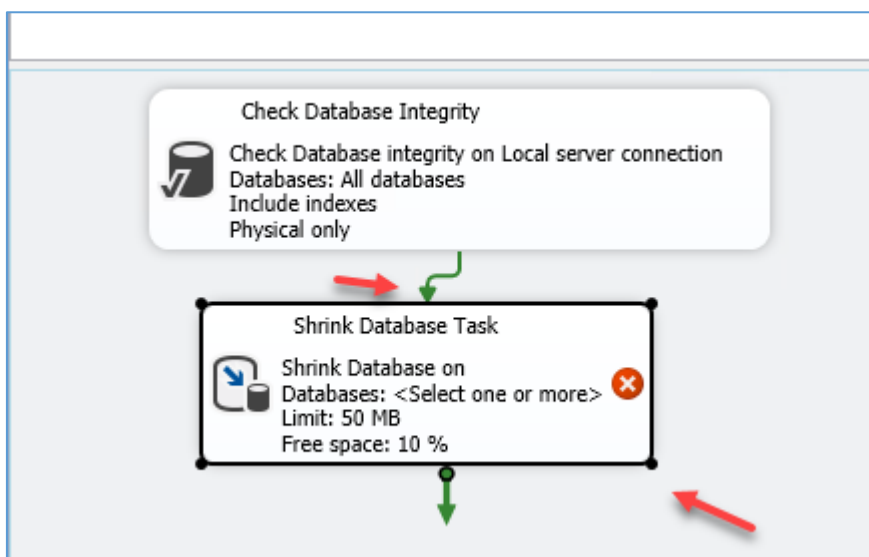
شکل ۵-۲۴

به‌مانند شکل ۵-۲۵ در subplan_1 باید گزینه‌ی Shrink را کشیده و در محل مورد نظر رها کنید.



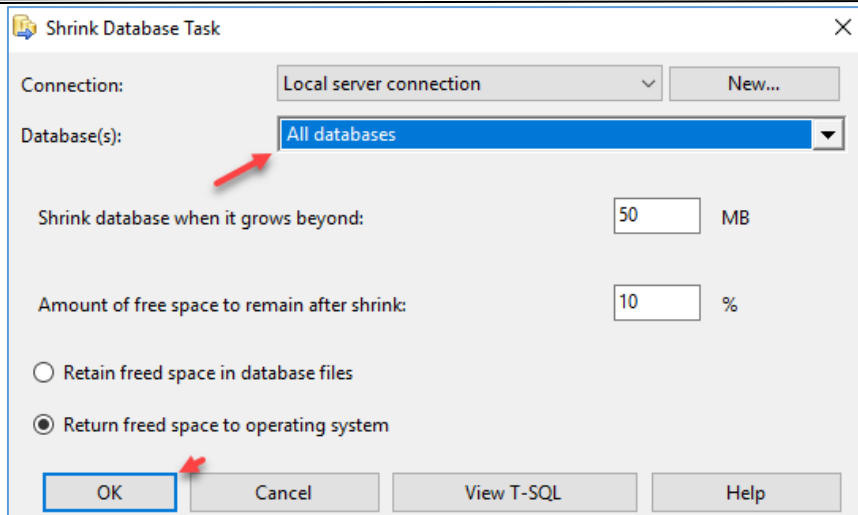
شکل ۵-۲۵

بعد از اضافه کردن Shrink باید فلش بالایی را به آن متصل کنید و بعد برای تنظیم آن دو بار بر روی Shrink کلیک کنید.



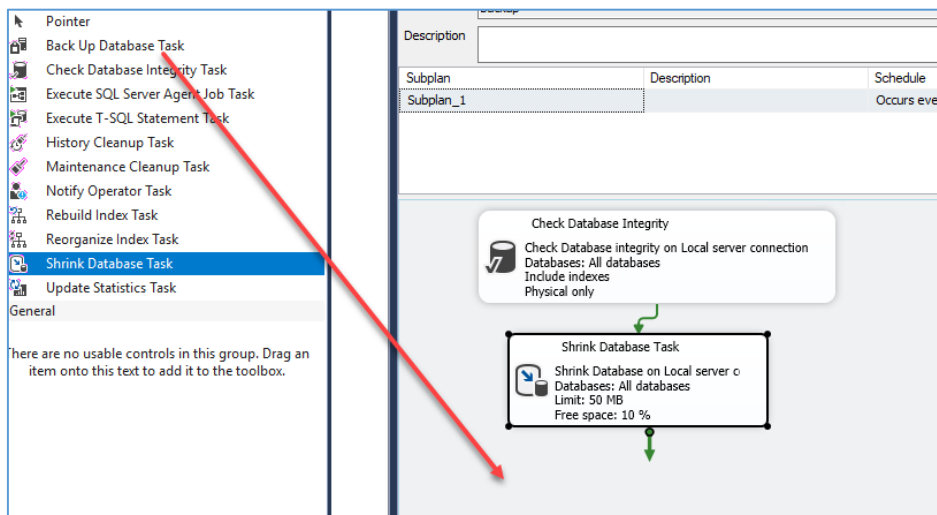
شکل ۵-۲۶

در شکل ۵-۲۷ باید All databases را انتخاب کنید و بر روی OK کلیک کنید.



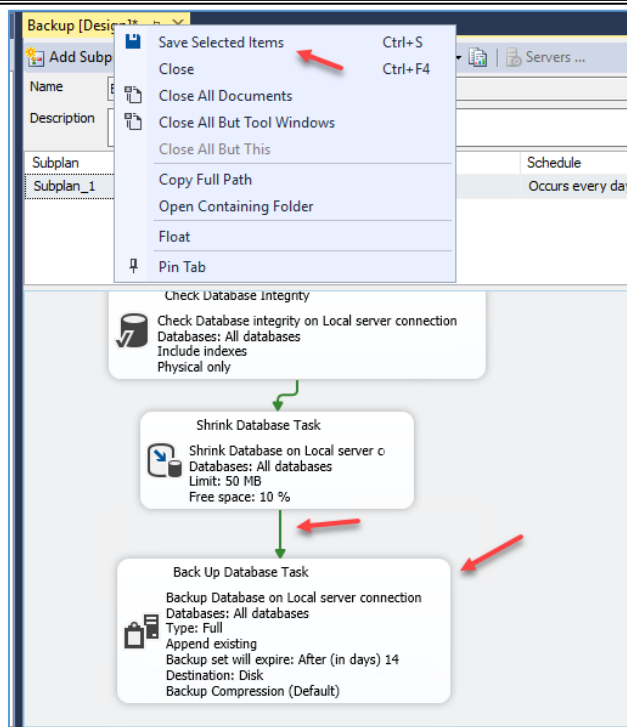
شکل ۵-۲۷

در ادامه باید گزینه‌ی Back UP Database Task را هم به لیست اضافه کنیم که این کار را باید به‌مانند شکل ۵-۲۸ انجام دهید.



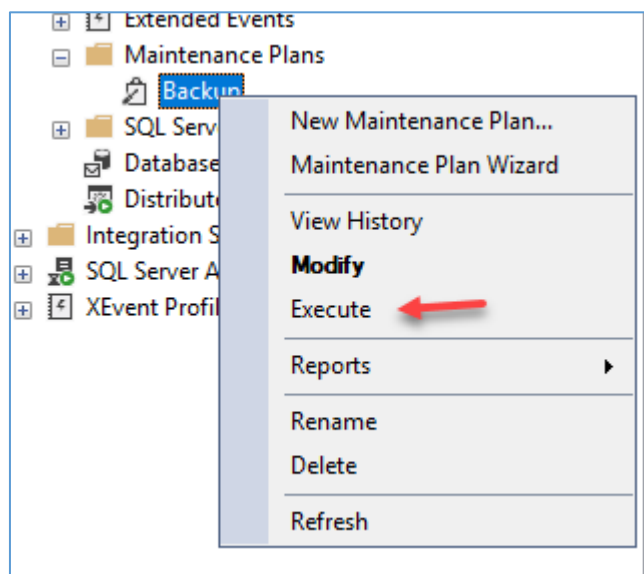
شکل ۵-۲۸

به‌مانند شکل ۵-۲۹ گزینه‌ی آخر را هم متصل کنید و با دو بار کلیک بر روی آن تنظیمات آن را هم به نسبت قبل انجام دهید و اطلاعات را ذخیره کنید.

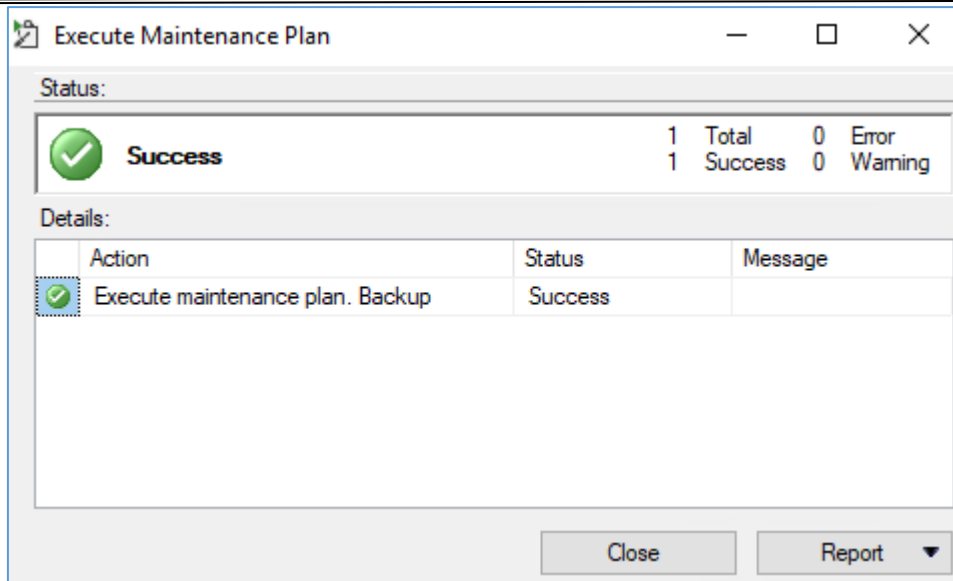


شکل ۵-۲۹

بعد از اجرای مراحل بالا اگر به مانند شکل ۵-۳۰ بر روی Execute کلیک کنید عملیات به مانند شکل ۵-۳۱ با موفقیت انجام شود.

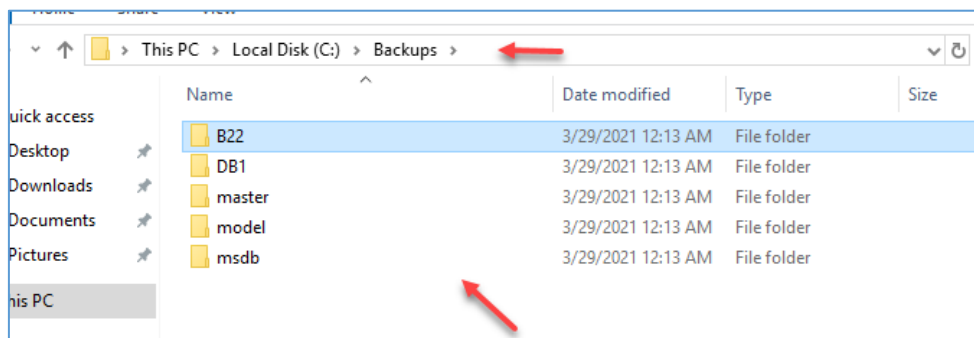


شکل ۵-۳۰



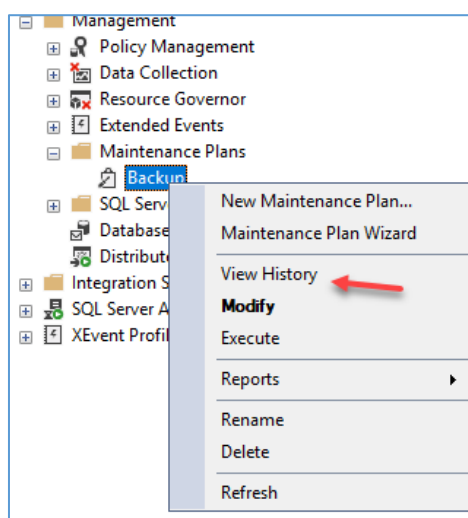
شکل ۵-۳۱

به‌مانند شکل ۵-۳۲-۵ وارد آدرس ذخیره‌سازی Backup شوید و مشاهده خواهید کرد فایل‌های Backup به‌درستی ایجاد شده‌اند.



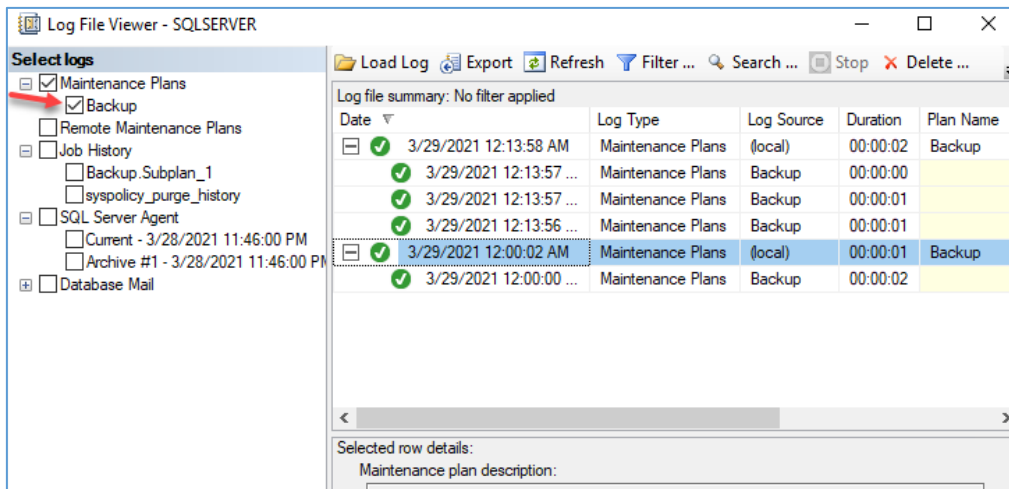
شکل ۵-۳۲

برای اینکه متوجه شوید Maintenance Plans چند بار اجرا شده و درست کار کرده یا نه باید به‌مانند شکل ۵-۳۳-۵ بر روی Plan مورد نظر کلیک راست و گزینه‌ی View History را انتخاب کنید.



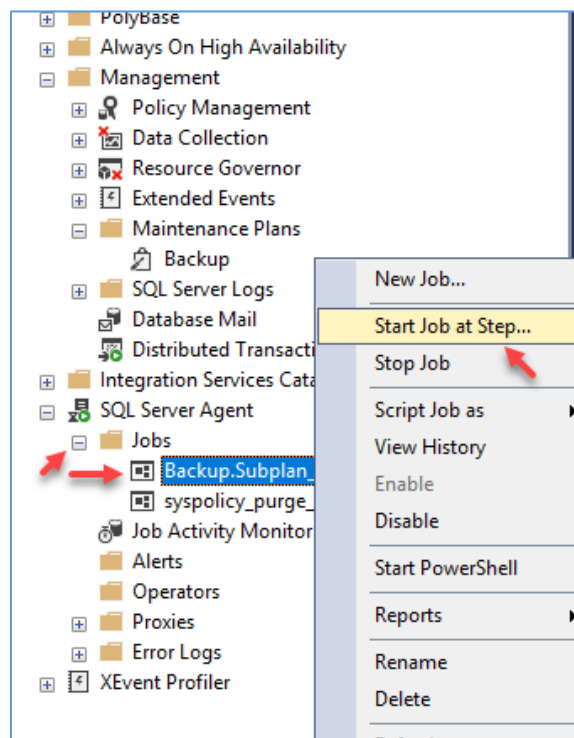
شکل ۵-۳۳

در شکل ۵-۳۴ نتیجه کار را مشاهده می‌کنید، توجه داشته باشید برای اینکه فقط Plan مورد نظر شما نمایش داده شود باید تیک آن را انتخاب کنید.



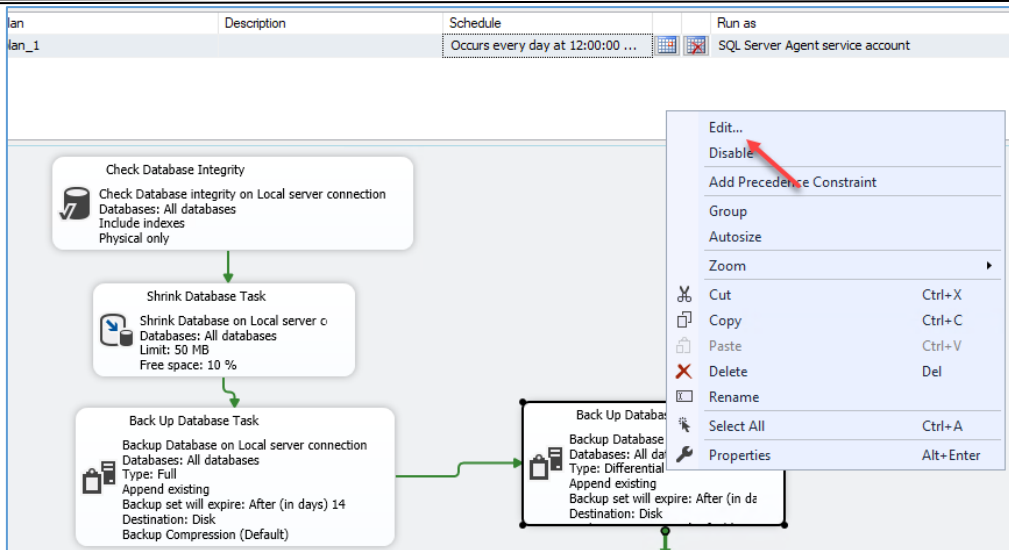
شکل ۵-۳۴

همان‌طور که گفتیم عملیات Maintenance Plans توسط سرویس Agent انجام می‌شود اگر به‌مانند شکل ۵-۳۵ وارد قسمت SQL Server agent شوید می‌توانید Plan مورد نظر را مشاهده و آن را اجرا و تنظیمات آن را تغییر دهید.



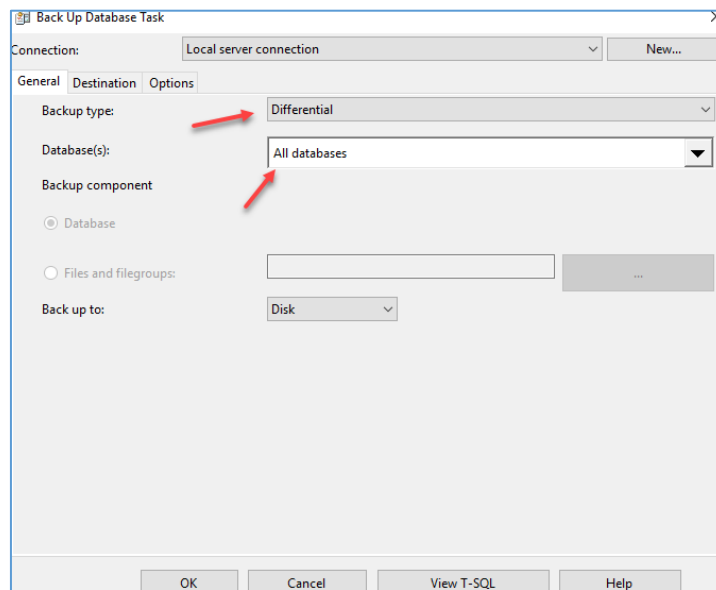
شکل ۵-۳۵

برای اینکه بتوانیم از Differential Backup استفاده کنیم باید به‌مانند شکل ۵-۳۶ یک Back UP Database به لیست اضافه کنید و بر روی آن کلیک راست کنید و گزینه‌ی Edit را انتخاب کنید، یا اینکه دوبار کلیک کنید.



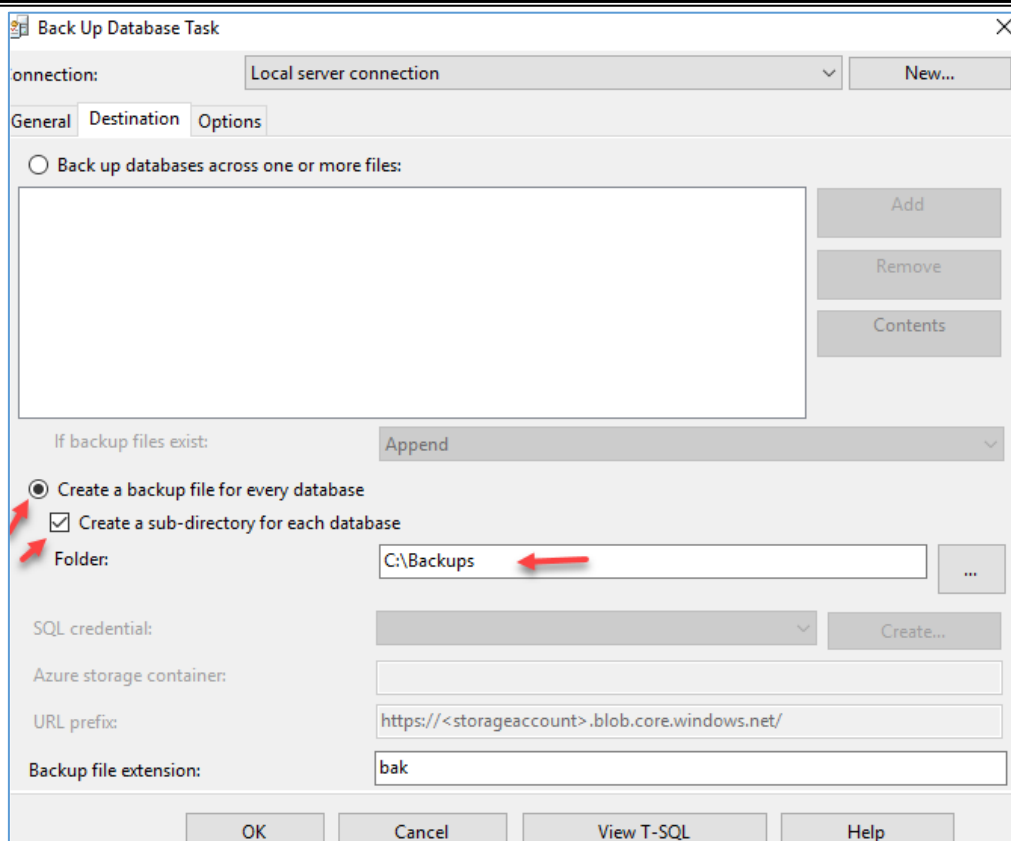
شکل ۵-۳۶

در شکل ۵-۳۷ باید در قسمت Backups Type گزینه‌ی Differential را انتخاب کنید و All Database را هم انتخاب کنید.



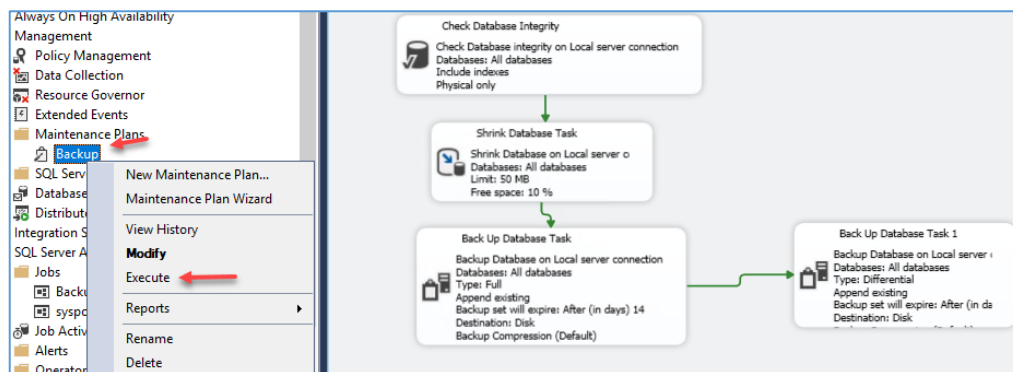
شکل ۵-۳۷

به‌مانند شکل ۵-۳۸ در تب Destination هم همان آدرس قبلی را انتخاب و تیک گزینه‌ی مورد نظر را انتخاب کنید، در تب Options هم همان گزینه‌های قبلی را انتخاب و بر روی OK کلیک کنید.



شکل ۵-۳۸

بعد از انجام مراحل بالا می‌توانید به‌مانند شکل ۵-۳۹ Plan مورد نظر خود را اجرا کنید.

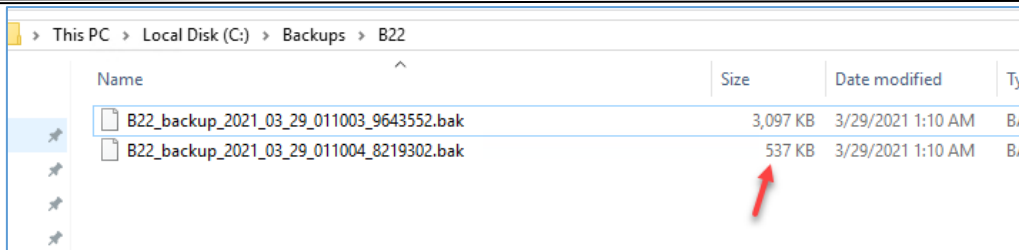


شکل ۵-۳۹

بعد از اجرا این سرویس به‌مانند شکل ۵-۴۰ در هر اجرا دو فایل پشتیبان ایجاد می‌کند که یکی Full است و دیگری Differenctail که حجم آنها مشخص‌کننده فایل مورد نظر است.

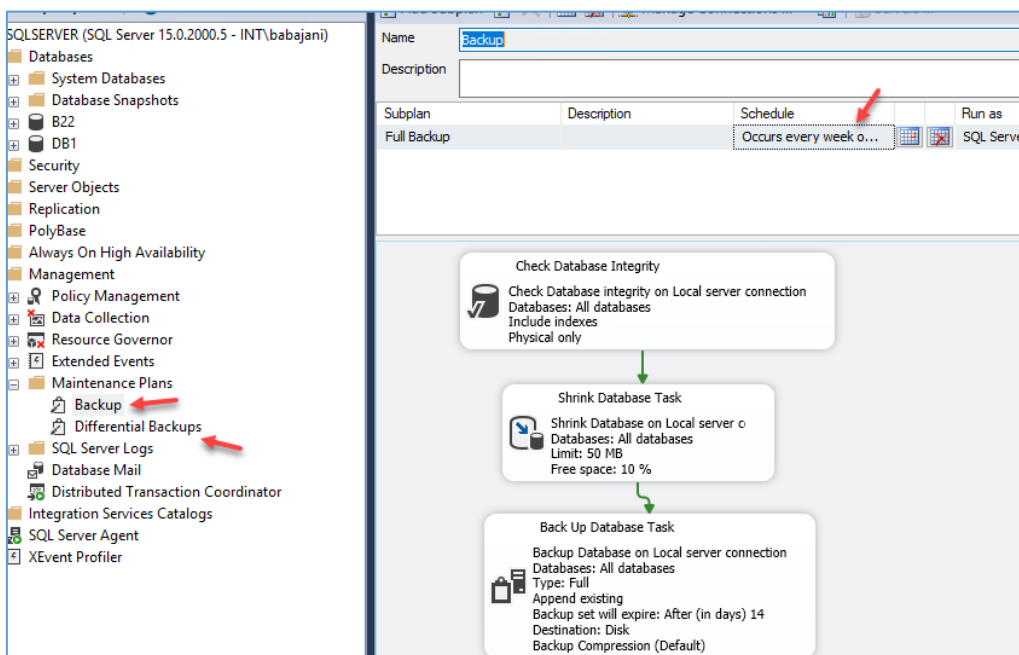
آیا به نظر شما این روش پشتیبان‌گیری درست است؟

نه این روش کاملاً اشتباه است، به‌خاطر اینکه این Plan که ایجاد کردیم هر دو فایل را در هر بار اجرا تولید می‌کند و این کار بسیار بر حجم فضای ذخیره‌سازی تاثیر گذار خواهد بود، برای حل این مشکل بهتر است هر کدام را جداگانه در یک Plan مختلف با زمانبندی مختلف قرار دهیم.



شکل ۴۰-۵

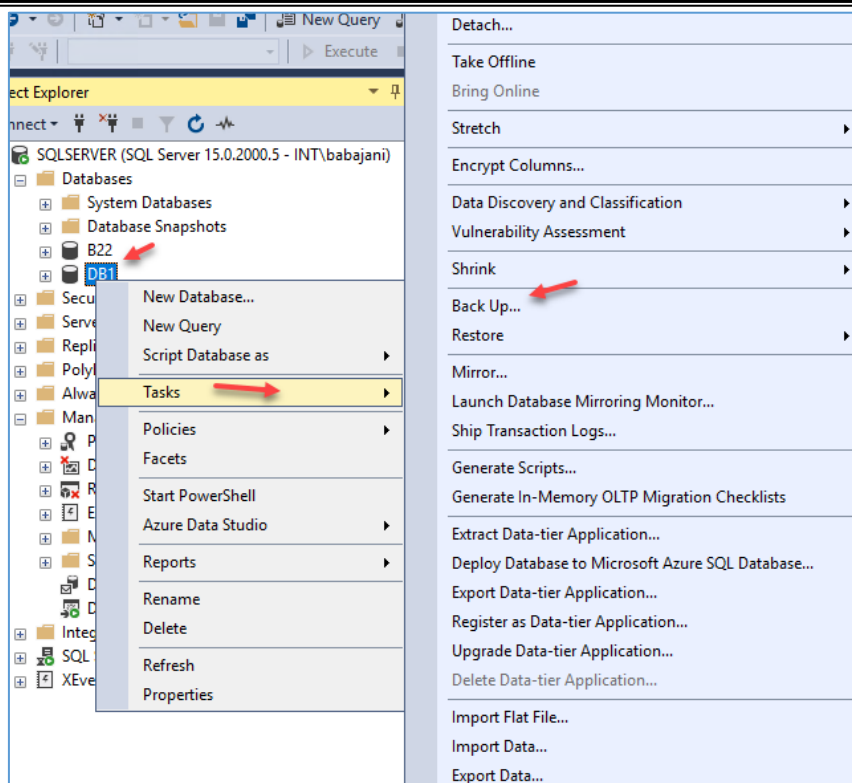
در شکل ۴۱-۵ دو Plan متفاوت ایجاد کردیم و در زمانبندی آنها برای Full Backup هفتگی را مشخص کردیم و برای Differential روزانه را مشخص کردیم، با این کار هر هفته یک Full و هر روز یک Differential گرفته خواهد شد.



شکل ۴۱-۵

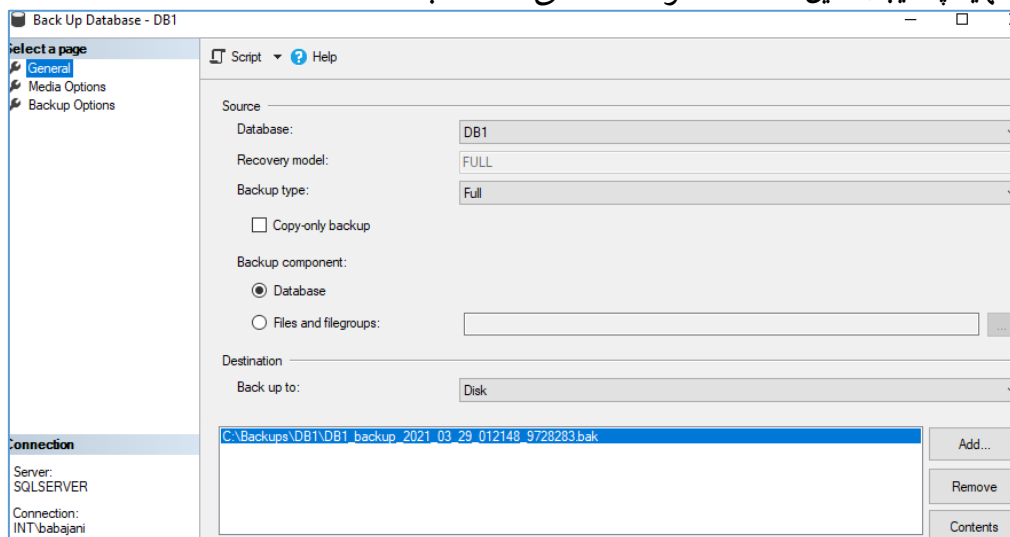
۱-۱-۵ پشتیبان‌گیری به صورت دستی در SQL

تا به اینجا توانستیم از طریق Maintenance Plans و به صورت اتوماتیک از دیتابیس‌ها پشتیبان تهیه کنیم، اما اگر بخواهید به صورت دستی و سریع از دیتابیس مورد نظر پشتیبان تهیه کنید باید به صورت زیر عمل کرد. به مانند شکل ۴۲-۵ بر روی دیتابیس مورد نظر خود کلیک راست کنید و از قسمت Tasks گزینه Backup را انتخاب کنید.



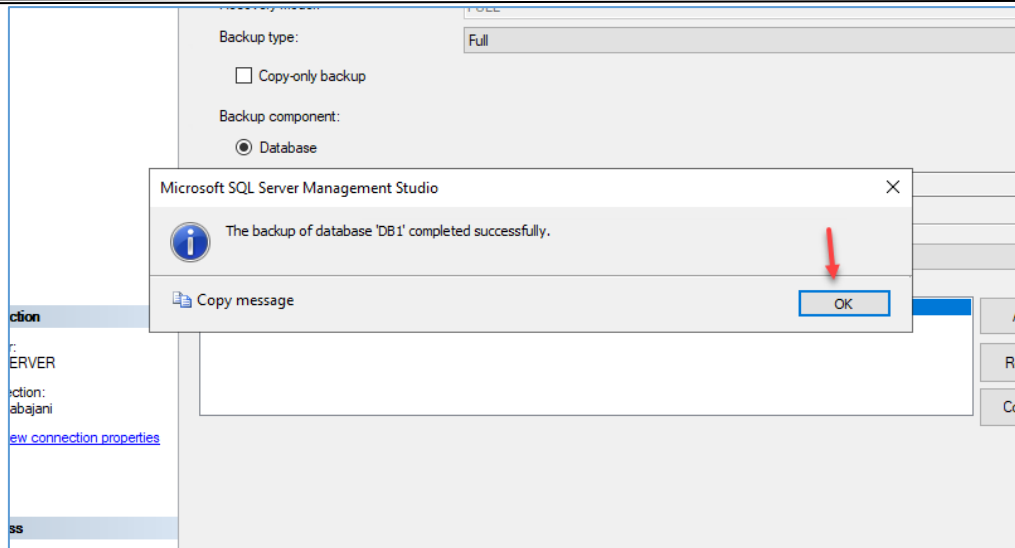
شکل ۴۲-۵

در شکل ۴۳-۵ باید نوع Backup را انتخاب کنید که گزینه‌ی Full را انتخاب می‌کنیم و در پایین صفحه باید مسیر ذخیره شدن فایل Backup را مشخص کنید که به صورت پیش فرض در مسیر نصب SQL قرار خواهد گرفت، ولی چون در قسمت قبلی از Plan استفاده کردیم مسیر را تغییر داده است، در تب‌های دیگر هم می‌توانید مشخص کنید که بعد از تهیه پشتیبان فایل‌ها تست شوند تا مشکلی نداشته باشند.



شکل ۴۳-۵

بعد از کلیک بر روی OK در شکل ۴۳-۵ عملیات شروع شده و تایید آن در شکل ۴۴-۵ مشخص می‌شود.

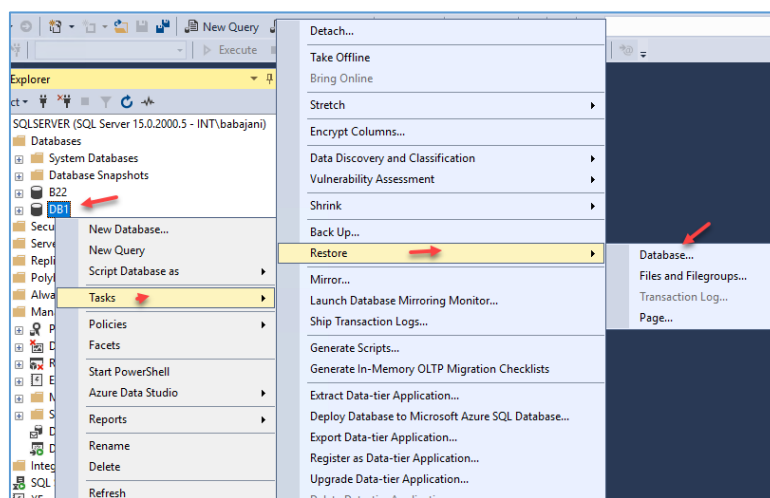


شکل ۵-۴۴

۵-۱-۲ نحوه‌ی بازگرداندن فایل‌های Backup

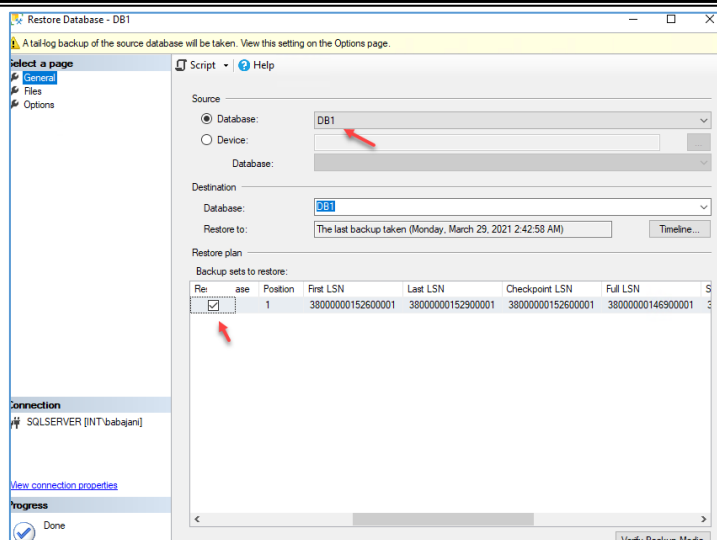
بعد از انجام پشتیبان‌گیری از دیتابیس‌های SQL باید توانایی این را داشته باشید در صورت خرابی هر یک از دیتابیس‌ها بتوانید آن را برگردانید.

برای این‌جمله دستور Restore باید به‌مانند شکل ۵-۴۵ بر روی دیتابیس مورد نظر که مشکل دارد کلیک راست کنید و از قسمت Tasks و بعد Restore گزینه‌ی Database را انتخاب کنید.



شکل ۵-۴۵

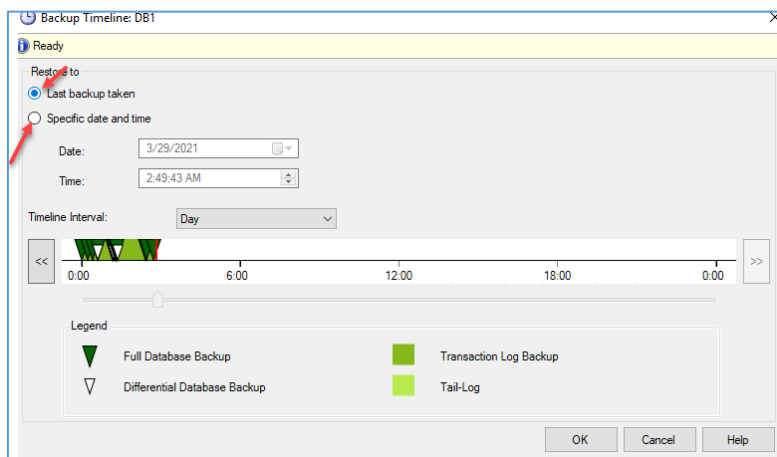
در شکل ۵-۴۶ باید نام دیتابیس به همراه پشتیبانی که از قبل تهیه کردید را انتخاب کنید.



شکل ۵-۴۶

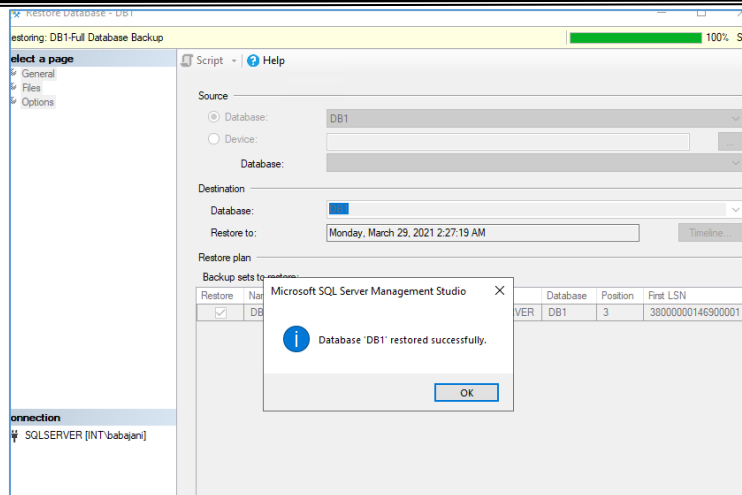
توجه داشته باشید که برای اینکه فایل پشتیبان برای تاریخ خاصی را انتخاب کنید باید در شکل ۵-۴۶ بر روی Timeline کلیک کنید.

در شکل ۵-۴۷ می‌توانید با انتخاب گزینه‌ی Last Backup taken از آخرین Backup گرفته شده استفاده کنید و یا اینکه اگر تاریخ خاصی مد نظر شما هست باید گزینه‌ی Specific date and time را انتخاب کنید و بعد تاریخ مورد نظر خود را مشخص کنید، البته در قسمت Timeline می‌توانید به صورت گرافیکی بر روی آیکن‌های مورد نظر کلیک کنید که آیکن سبز پر رنگ نشان‌دهنده‌ی Full Backup است.



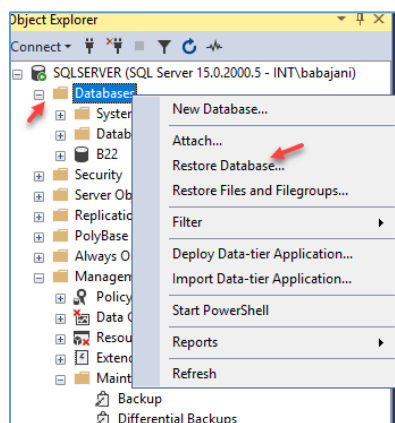
شکل ۵-۴۷

بعد از اینکه بر روی OK کلیک کردید باید عملیات Restore انجام شود که در شکل ۵-۴۸ این موضوع را مشاهده می‌کنید.



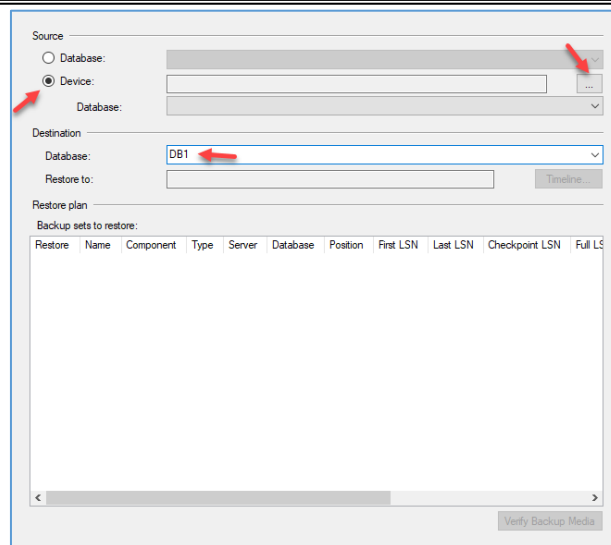
شکل ۴۸-۵

۳-۱-۵ بازگرداندن دیتابیس حذف شده در بعضی مواقع امداداً یا سهواً دیتابیس از داخل SQL حذف می‌شود و دیگر نمی‌توانیم به آن دسترسی داشته باشیم، در این زمان باید نسخه پشتیبان را بازگردانیم. به‌مانند شکل ۴۹-۵ بر روی پوشه‌ی Databases کلیک راست کنید و گزینه‌ی Restore Databases را انتخاب کنید.



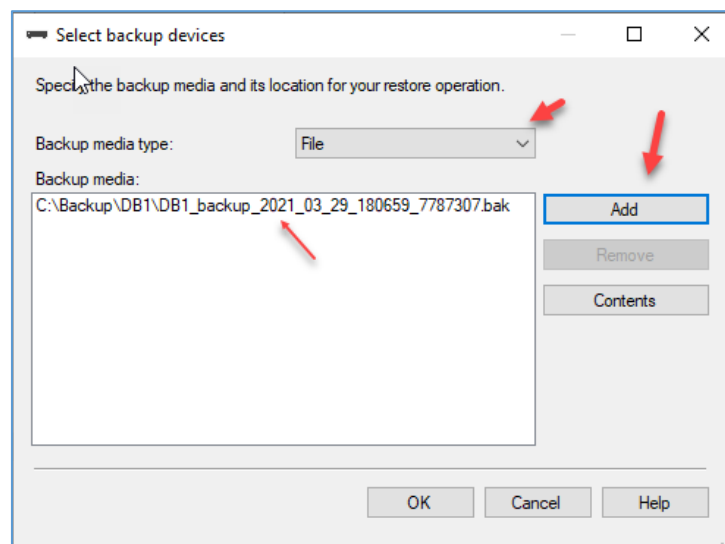
شکل ۴۹-۵

در شکل ۵۰-۵ شما باید دیتابسی که قرار است برگردانید را از لیست انتخاب کنید که این کار به علت حذف آن دیتابیس امکان پذیر نیست به‌خاطر همین موضوع باید در قسمت Database نام دیتابیس خود را وارد کنید و در ادامه باید گزینه‌ی Device کلیک کنید و بعد بر روی Browse کلیک کنید.



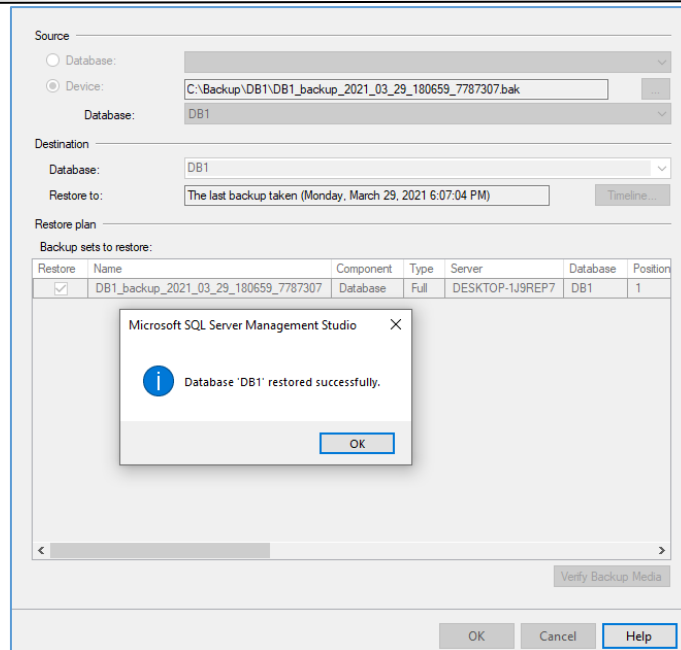
شکل ۵-۵۰

در شکل ۵-۵۱ باید بر روی Add کلیک کنید و فایل پشتیبان Full را به لیست اضافه و بر روی OK کلیک کنید.



شکل ۵-۵۱

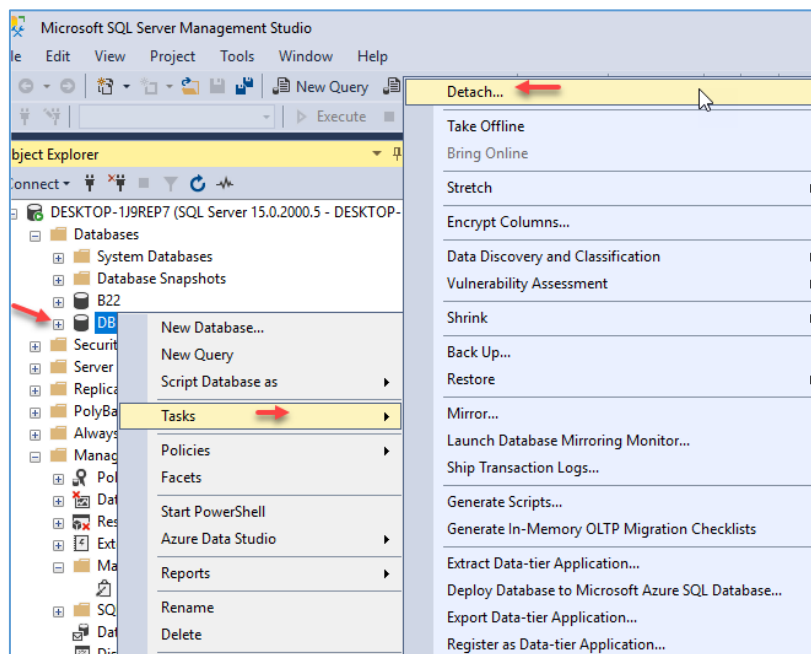
به‌مانند شکل ۵-۵۲ بعد از کلیک بر روی OK به‌درستی دیتابیس مورد نظر به لیست اضافه خواهد شد.



شکل ۵-۵۲

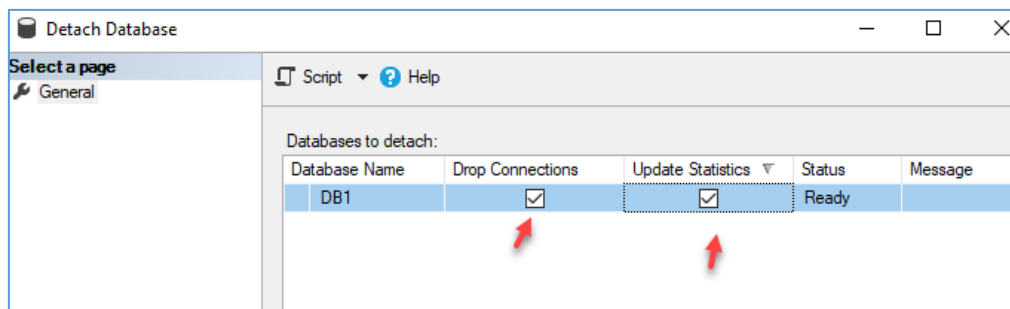
۴-۱-۵ Attach و Detach کردن دیتابیس

برای اینکه بتوانید یک دیتابیس را از لیست SQL مخفی و یا آن را اضافه کنید باید از این دستورات استفاده کنید. برای این کار به‌مانند شکل ۵-۵۳ بر روی دیتابیس مورد نظر کلیک راست کنید و از قسمت Tasks گزینه‌ی Detach را انتخاب کنید.



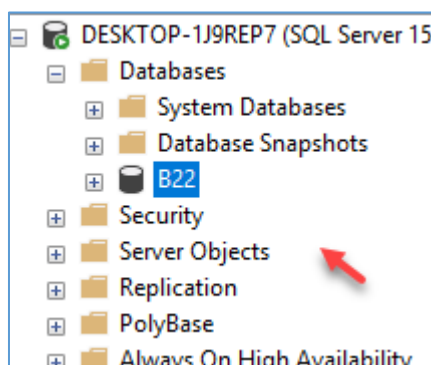
شکل ۵-۵۳

در شکل ۵-۵۴ باید دو تیک مرود نظر را انتخاب کنید و بعد بر روی OK کلیک کنید، گزینه‌ی Drop Connection ارتباط این دیتابیس را با هر نرم‌افزاری قطع می‌کند و گزینه‌ی بعدی برای آپدیت اطلاعات است.



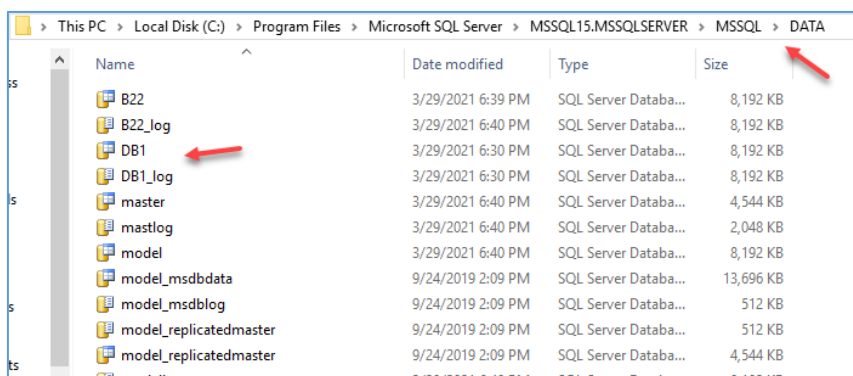
شکل ۵-۵۴

همان‌طور که در شکل ۲-۵۵ مشاهده می‌کنید دیتابیس DB1 از لیست حذف شده است این بدان معنا نیست که دیتابیس از هارد حذف شده باشد بلکه فقط از لیست حذف شده است که این موضوع را در شکل ۵-۵۶ مشاهده می‌کنید



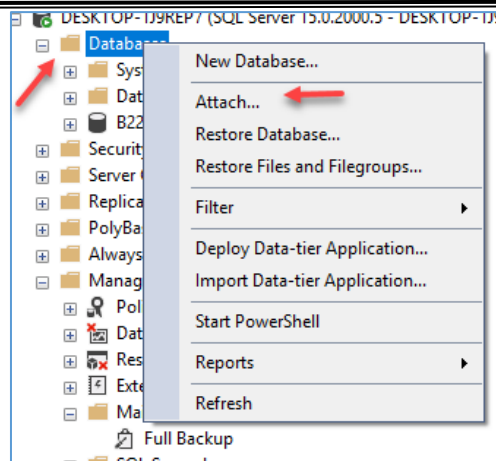
شکل ۵-۵۵

اگر به‌مانند شکل ۵-۵۶ وارد آدرس مورد نظر شوید دیتابیس DB1 را مشاهده خواهید کرد.



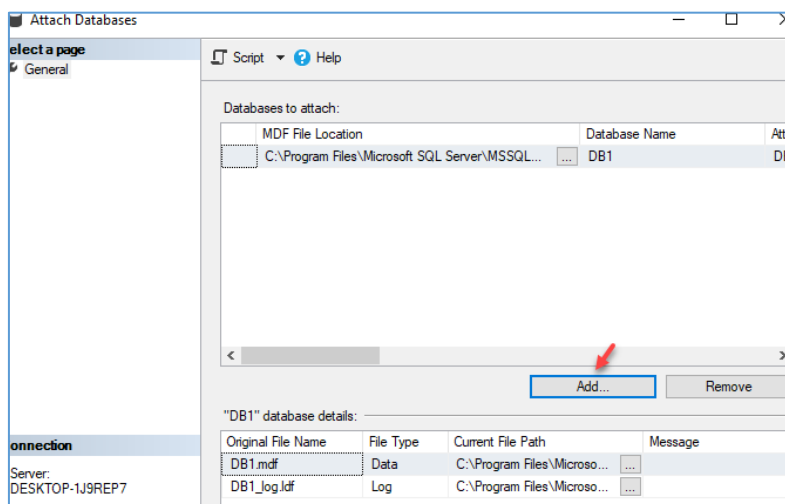
شکل ۵-۵۶

برای اینکه دوباره دیتابیس را به SQL اضافه کنیم به‌مانند شکل ۵-۵۷ بر روی پوشه‌ی Databases کلیک راست کنید و گزینه‌ی Attach را انتخاب کنید.



شکل ۵-۵۷

در شکل ۵-۵۸ بر روی Add کلیک کنید و دیتابیس DB1 را از آدرس مورد نظر که در شکل ۵-۵۶ مشخص کردیم انتخاب کنید و بر روی OK کلیک کنید.



شکل ۵-۵۸

استفاده‌های دیگر این ابزار زمانی است بخواهید سرور SQL خود را تغییر دهید و دیتابیس‌ها را از یک سرور به سرور دیگر انتقال دهید، البته راه‌های بهتر و متنوع‌تری هم برای این کار وجود دارد.

۵-۲ پشتیبان‌گیری از طریق نرم‌افزار Veam Backup

یک نرم‌افزار قدرتمند در پشتیبان‌گیری است که حداکثر اطمینان را برای شما به ارمغان می‌آورد؛ با این نرم‌افزار فوق‌العاده می‌توانید از نرم‌افزارها، فایل‌ها، سایت‌ها و... پشتیبان تهیه کنید و کمتر از چند ثانیه آن را برگردانید. نرم‌افزار Veam، یکی از بهترین‌ها در بازار نرم‌افزارها است و به‌خاطر اطمینان کامل در بازگردانی اطلاعات، رقیبی مقابل خود نمی‌بیند.

۵-۲-۱ نصب نرم‌افزار Veam Backup and Replication

این نرم‌افزار به‌مانند دیگر نرم‌افزارها دارای نیازمندی‌هایی برای نصب است که در زیر آنها را مورد بررسی قرار می‌دهیم.

جدول ۵-۱ نیازمندی سخت‌افزاری

پردازنده بهتر است، حداقل دو هسته‌ای و ۲ گیگاهرتز سرعت داشته باشد.	پردازنده
حداقل رم باید ۴ گیگابایت باشد تا عملکرد متوسط از سرور را داشته باشیم.	رم
بهترین حالت استفاده از کارت شبکه، گیگابایت است تا حداکثر سرعت را در انتقال فایل داشته باشیم.	کارت شبکه
برای نصب و راه‌اندازی، حداقل باید ۶ گیگابایت محل ذخیره‌سازی داشته باشیم، البته برای اجرای Backup و ذخیره این فایل‌ها باید هاردهایی با ظرفیت بالا به نسبت شبکه داشته باشیم تا کل اطلاعات شبکه ما را پشتیبانی کند.	هارد دیسک

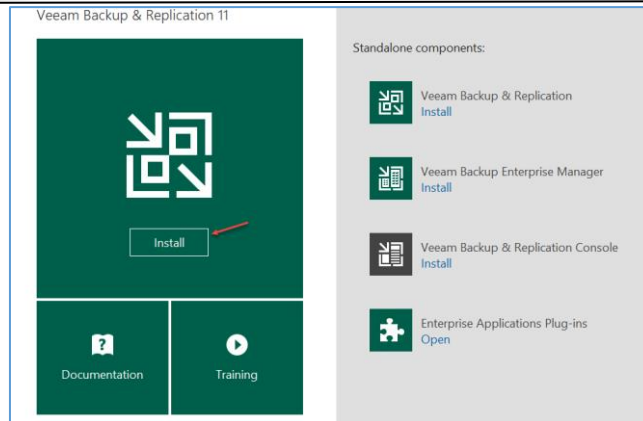
جدول ۵-۲ نیازمندی نرم‌افزاری

از سیستم عامل ویندوز که لیست آن در زیر آمده است، می‌توان استفاده کرد: Microsoft Windows Server 2019 Microsoft Windows Server 2016 Microsoft Windows Server 2012 R2 Microsoft Windows Server 2012 Microsoft Windows Server 2008 R2 SP1 Microsoft Windows Server 2008 SP2 Microsoft Windows 10 Microsoft Windows 8.x Microsoft Windows 7 SP1	سیستم عامل
این نرم‌افزارهای اولیه برای نصب Veeam مورد نیاز است: Microsoft .NET Framework 4.5.2 Microsoft Windows Installer 4.5 Microsoft SQL Server Management Objects Microsoft SQL Server System CLR Types Microsoft Visual C++ 2010 Service Pack 1 redistributable package	نرم‌افزار
در زیر، لیستی از دیتابیس‌های مورد استفاده نرم‌افزار Veeam را مشاهده می‌کنید: Microsoft SQL Server 2017 Microsoft SQL Server 2016 Microsoft SQL Server 2014 Microsoft SQL Server 2012 (Microsoft SQL Server 2012 SP3 Express Edition is included in the setup) Microsoft SQL Server 2008 R2 Microsoft SQL Server 2008	دیتابیس

برای دانلود این نرم‌افزار می‌توانید از لینک زیر استفاده کنید:

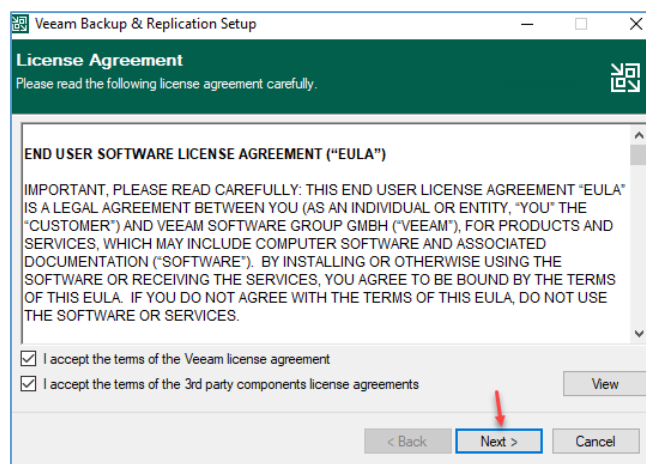
<https://soft98.ir/software/backup/2098-veeam-backup.html>

بعد از اجرای فایل Setup بر روی Install کلیک کنید، گزینه‌های دیگری نیز وجود دارد که گزینه Enterprise برای مدیریت چندین Backup & Replication Veeam کاربرد دارد و گزینه Console نیز برای ارتباط از سیستم خودتان با سرورهای Veeam است که فعلاً کاری با آنها نداریم.



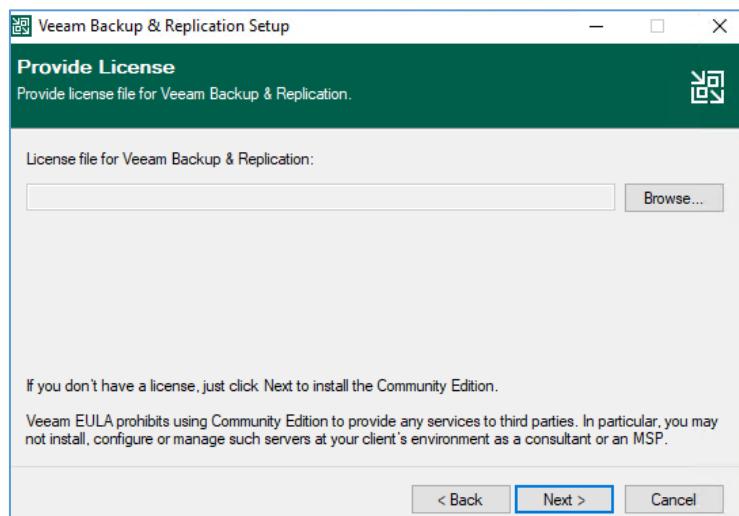
شکل ۵-۵۹ نصب Veeam

در شکل ۵-۶۰، قرار دادنامه را بررسی و تأیید کنید و بر روی Next کلیک کنید.



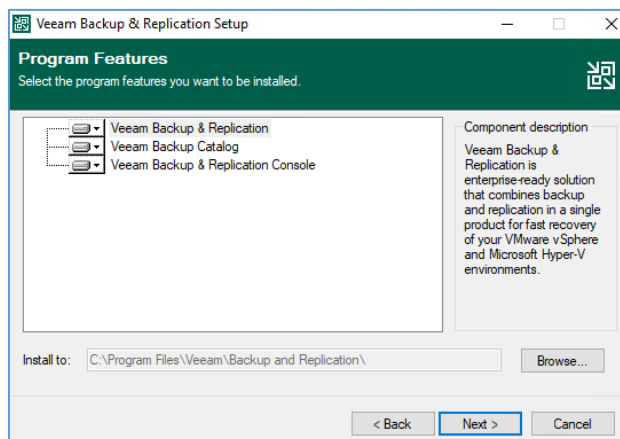
شکل ۵-۶۰ نصب Veeam

در شکل ۵-۶۱، اگر لایسنس نرم‌افزار را خریداری کردید، بر روی Browse کلیک کنید و فایل لایسنس را معرفی کنید و اگر لایسنسی در اختیار ندارید، می‌توانید از آن صرف‌نظر کنید و بر روی Next کلیک کنید که البته یک لایسنس Trial را برای شما در نظر می‌گیرد و بعد از چند روز غیرفعال خواهد شد.



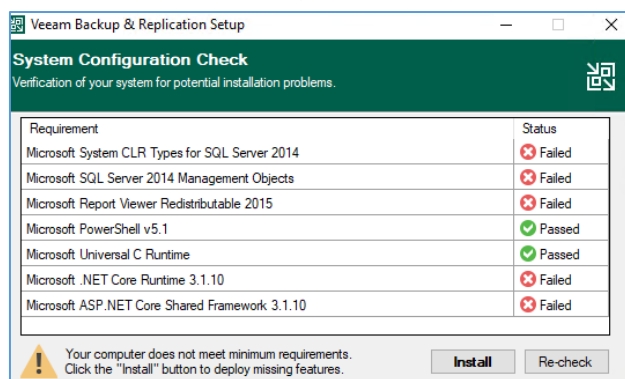
شکل ۵-۶۱ نصب Veeam

در شکل ۵-۶۲ می‌توانید مسیر نصب و اجرای نرم‌افزار را تغییر دهید، حتماً مطمئن شوید که مسیر مورد نظر دارای فضای کافی برای نصب باشد، بر روی Next کلیک کنید.



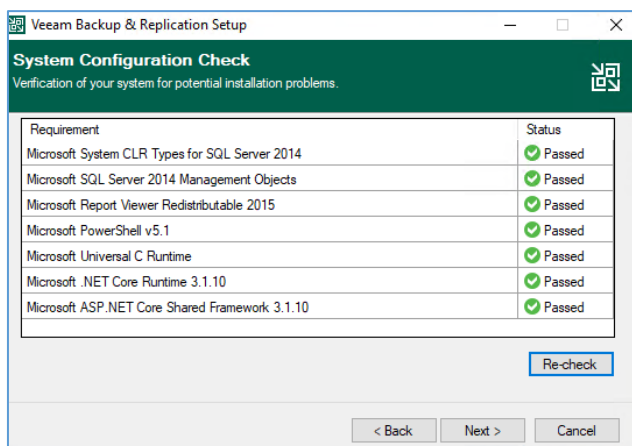
شکل ۵-۶۲ نصب Veeam

در شکل ۵-۶۳، اگر پیش‌نیازهای نرم‌افزار بر روی سیستم شما نصب باشد، خطایی نمی‌دهد، اما اگر با خطا مواجه شدید می‌توانید بر روی Install کلیک کنید تا کار نصب انجام شود.



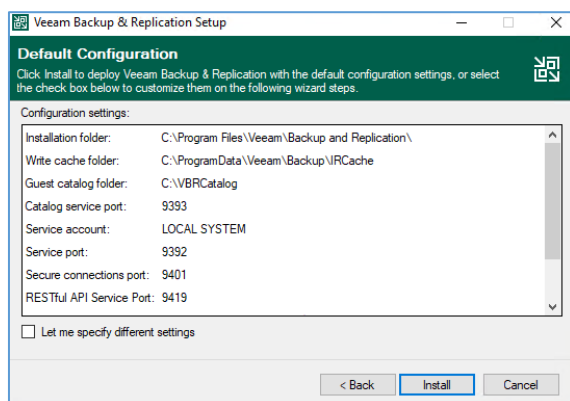
شکل ۵-۶۳ نصب Veeam

همان‌طور که در شکل ۵-۶۴ مشاهده می‌کنید، پیش‌نیازهای نرم‌افزار به‌درستی نصب شده است، بر روی Next کلیک کنید.



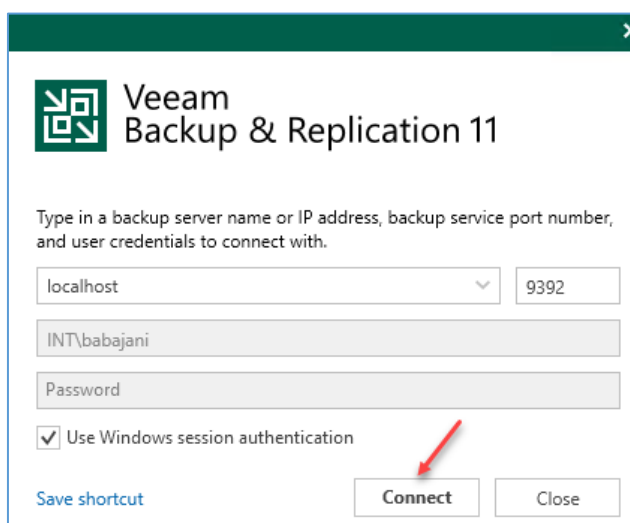
شکل ۵-۶۴ نصب Veeam

در شکل ۵-۶۵ بر روی Install کلیک کنید تا کار نصب آغاز شود.



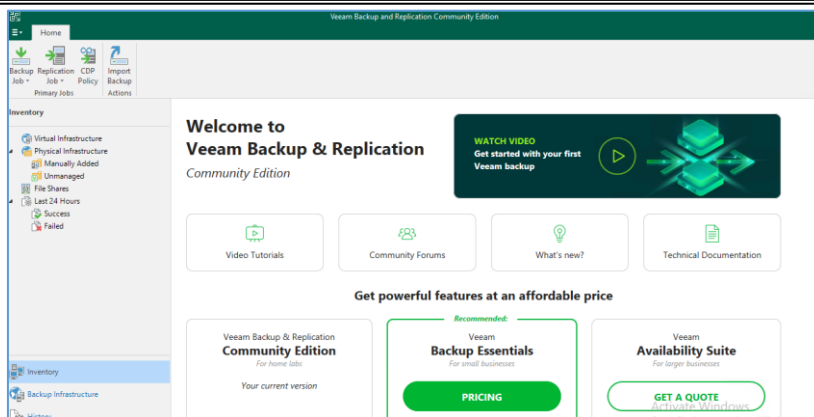
شکل ۵-۶۵ نصب Veeam

بعد از نصب، نرم‌افزار Veeam backup & Replication Console را اجرا کنید، به‌مانند شکل ۵-۶۶ در قسمت اول، کلمه Localhost نوشته شده است که اشاره می‌کند به سیستمی که در حال کار با آن هستید و نرم‌افزار بر روی آن نصب شده است، اگر نرم‌افزار بر روی سرور دیگری نصب شده باشد باید به‌جای Localhost، نام سرور مورد نظر را وارد کنید، برای ورود از همان کاربری استفاده می‌کند که با آن وارد سیستم شده‌اید؛ بر روی Connect کلیک کنید تا نرم‌افزار اجرا شود.



شکل ۵-۶۶ اجرای Veeam

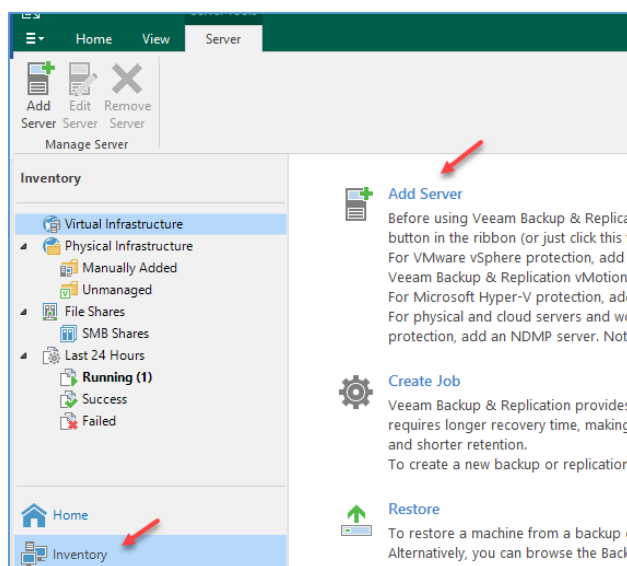
همان‌طور که در شکل ۵-۶۷ مشاهده می‌کنید، نرم‌افزار اجرا شده و آماده کار است، البته لایسنس آن باید فعال شود تا چند روز دیگر انقضا نشود، این نرم‌افزار دارای ابزارهای مختلفی است که می‌تواند عملکرد مشخصی را ارائه دهد، در ادامه نحوه‌ی پشتیبان‌گیری از ماشین مجازی و فایل‌های آنها مخصوصاً دیتابیس SQL را بررسی خواهیم کرد.



شکل ۵-۶۷ اجرای نرم‌افزار

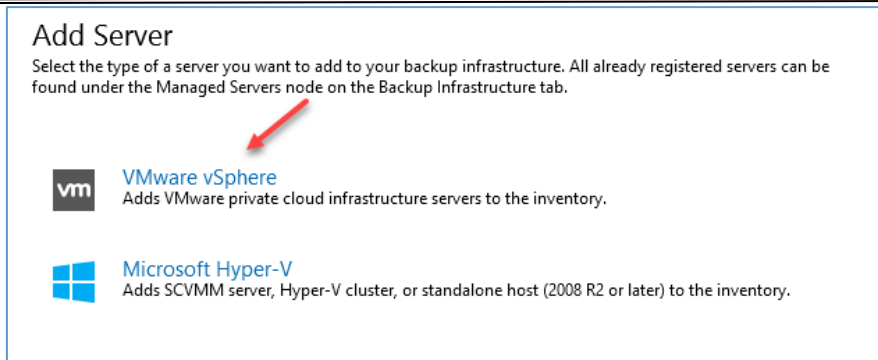
۲-۲-۵ اضافه کردن سرورها برای پشتیبان‌گیری

در شکل ۵-۶۸، برای اینکه سرورهای و فایل‌هایی که قرار است از آنها پشتیبان تهیه کنیم را باید در قسمت Inventory تعریف کنیم، حالا می‌تواند یک ماشین مجازی باشد و یا یک فایل خاص در مسیر مشخص از شبکه، برای معرفی سرور خود می‌توانید بر روی Add Server کلیک کنید.



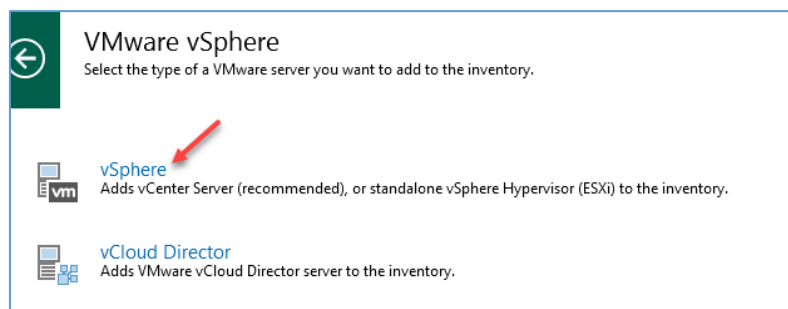
شکل ۵-۶۸

در شکل ۵-۶۹ باید مشخص کنید که سیستم مجازی‌سازی شما مربوط به کدام شرکت است اگر از VMware vSphere استفاده می‌کنید گزینه‌ی اول را انتخاب کنید و یا اگر از Hyper-v شرکت مایکروسافت استفاده می‌کنید گزینه‌ی دوم را انتخاب کنید، در مورد هر دو نرم‌افزار در کتاب‌های مدیر شبکه و VMware Systems آموزش کافی داده شده و برای دریافت می‌توانید به سایت بنده مراجعه کنید، برای ادامه کار بر روی گزینه‌ی اول کلیک کنید.



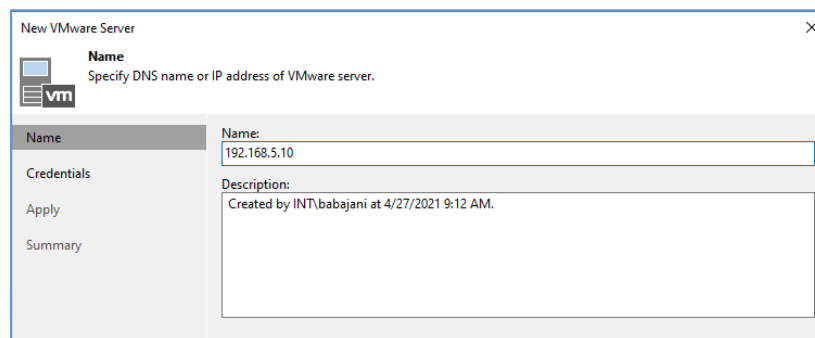
شکل ۵-۶۹ معرفی سرور vSphere

در صفحه‌ی شکل ۵-۷۰ بر روی vSphere کلیک کنید تا سرور vCenter یا ESXi خود را معرفی کنیم.



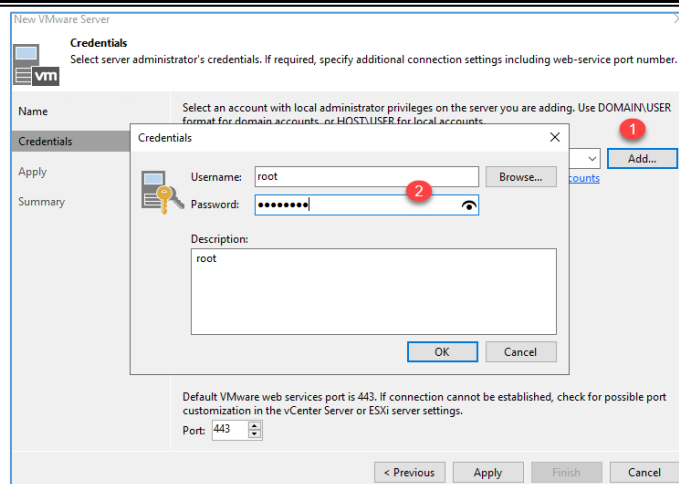
شکل ۵-۷۰ معرفی سرور vSphere

در شکل ۵-۷۱ باید آدرس سرور vCenter یا ESXi خود را وارد کنید، توجه داشته باشید، البته شاید برای راه‌اندازی سرور SQL از نرم‌افزار VMware Workstation یا یک سرور فیزیکی استفاده کرده باشید که در ادامه کتاب نحوه پشتیبان‌گیری از آنها هم آموزش داده شده است.



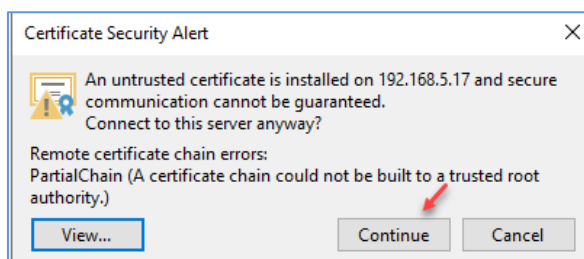
شکل ۵-۷۱ معرفی سرور ESXi

در شکل ۵-۷۲ باید نام کاربری که برای ورود به سرور استفاده می‌کنید را وارد کنید، برای این کار بر روی Add کلیک کنید و در قسمت شماره دو، نام کاربر را وارد و بر روی OK کلیک کنید، پورت پیش‌فرض برای ارتباط با سرور، ۴۴۳ است که در صورت تغییر آن باید شماره پورت خود را وارد کنید.



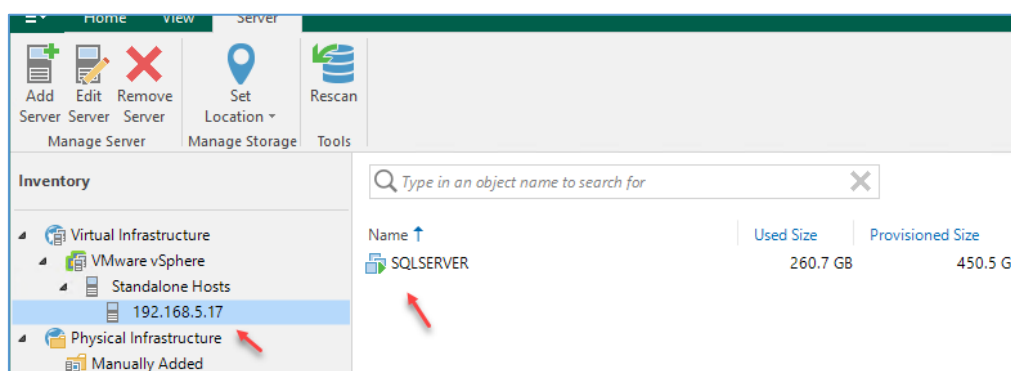
شکل ۵-۷۲- تعریف نام کاربری و رمز عبور

بعد از اینکه بر روی Apply کلیک کنید شکل ۵-۷۳ ظاهر خواهد شد که باید گواهینامه‌ی سرور را مورد تأیید قرار دهید.



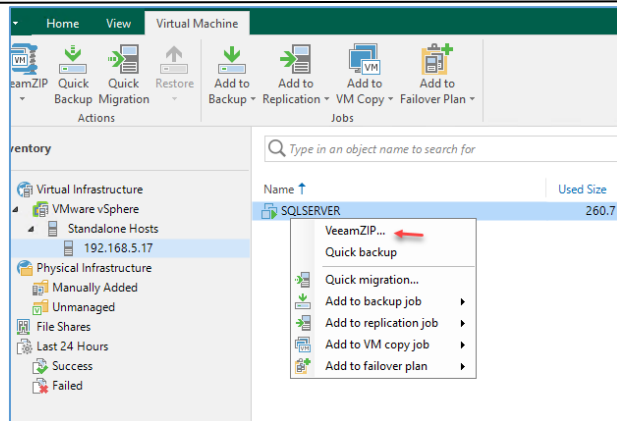
شکل ۵-۷۳- تأیید گواهینامه

همان‌طور که در شکل ۵-۷۴ مشاهده می‌کنید سرور مورد نظر به لیست اضافه شده است و با کلیک بر روی آن همه‌ی ماشین‌های مجازی مورد نظر را مشاهده خواهید کرد.



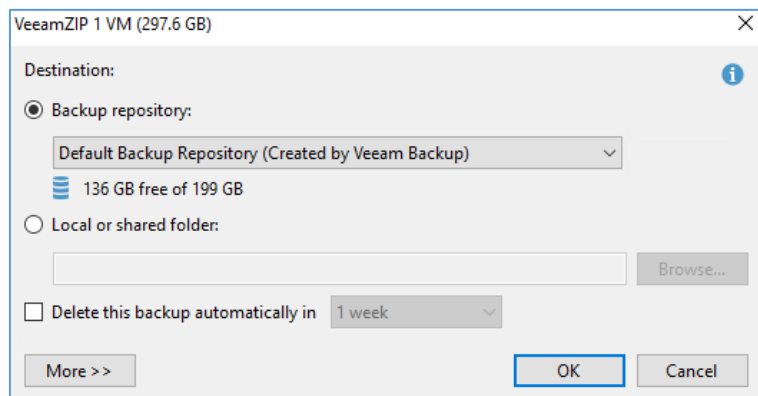
شکل ۵-۷۴- سرور ESXi

به‌مانند شکل ۵-۷۵، برای اینکه تستی از نرم‌افزار داشته باشیم، می‌توانیم بر روی ماشین مجازی خود در سرور VMware کلیک راست کنیم و از آن پشتیبان تهیه کنیم، قسمت شماره ۱ و ۲، ابزارهایی برای این کار وجود دارد؛ برای شروع بر روی VeeamZip کلیک کنید.



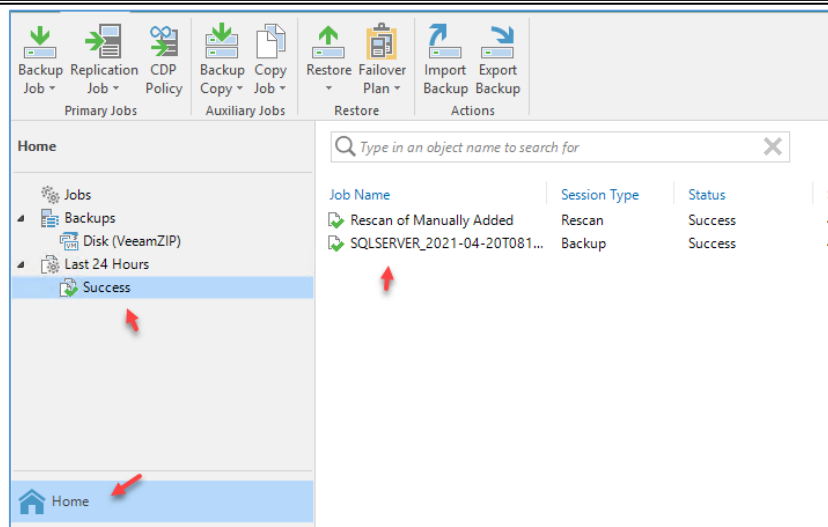
شکل ۵-۷۵ / ایجاد پشتیبان دستی

در شکل ۵-۷۶ باید آدرس ذخیره شدن آن را بر روی سیستم خود مشخص و بر روی ok کلیک کنید، توجه داشته باشید که برای ذخیره اطلاعات و فایل‌های پشتیبان باید در قسمت Repository یک مسیر ذخیره‌سازی مشخص کنیم که در اینجا به صورت پیش‌فرض یک مسیر در سروری که Veeam نصب شده است انتخاب شد ولی در ادامه و در قسمت Repository حتماً مراحل را به صورت کامل بررسی خواهیم کرد.



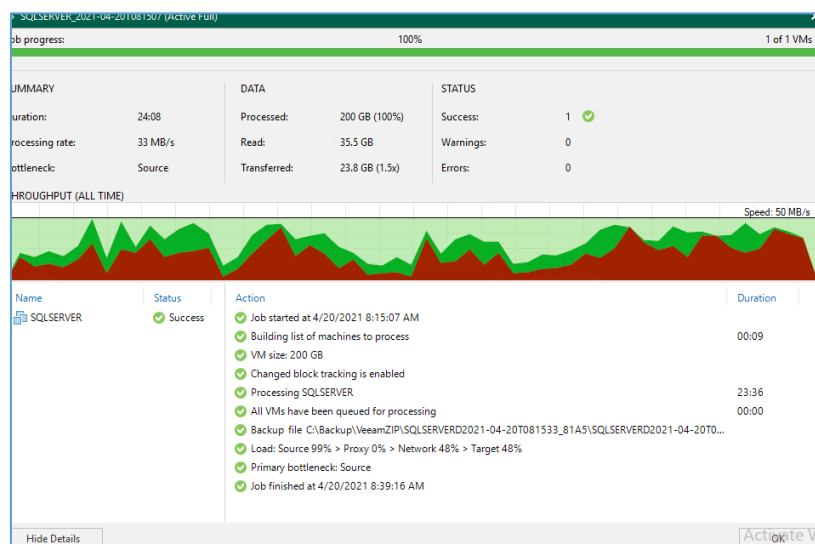
شکل ۵-۷۶ / ایجاد پشتیبان دستی

برای اینکه متوجه شویم کاری که انجام دادیم به درستی عمل کرد یا نه باید از سمت چپ بر روی Home کلیک کنید و در قسمت Last 24 Hours می‌توانیم آخرین تغییرات را مشاهده کنیم، در شکل ۵-۷۷ مشخص شده است که عملیات پشتیبان‌گیری از سرور SQL با موفقیت انجام شده است.



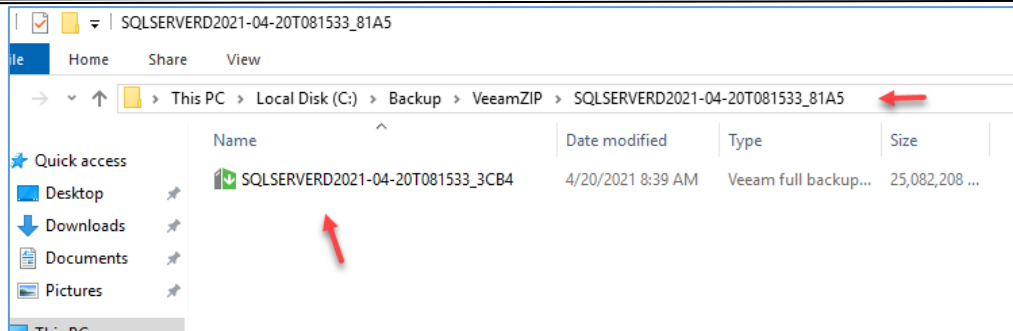
شکل ۷۷-۵ تغییرات فایل

اگر بر روی هرکدام از Log های مورد نظر در شکل ۷۷-۵ کلیک کنید، صفحه‌ای به‌مانند شکل ۷۸-۵ ظاهر می‌شود، در این قسمت، اطلاعات مربوط به فایل‌هایی که از آن پشتیبان تهیه کردید را مشاهده خواهید کرد، در این قسمت می‌توانید اندازه بک آپ را در قسمت Transfer مشاهده کنید که برابر با ۲۳.۸ گیگابایت است.



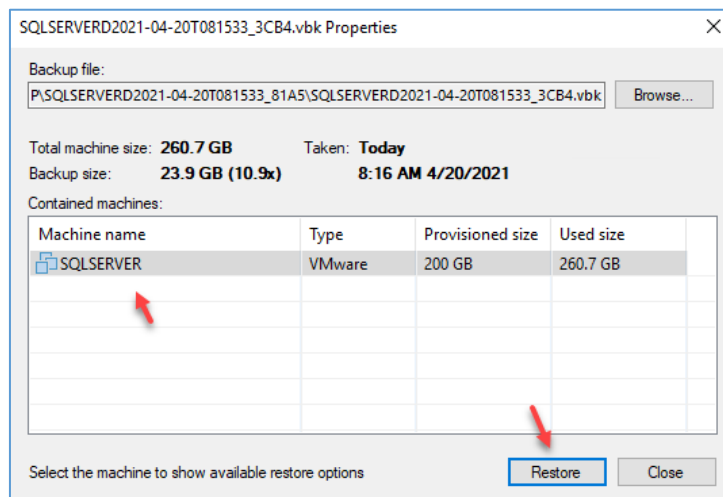
شکل ۷۸-۵ بررسی Log

اگر بخواهیم فایل پشتیبان را مشاهده کنید یا نه وارد مسیر ذخیره‌سازی مورد نظر شوید که به‌مانند شکل ۷۹-۵ به‌صورت یک فایل تگ‌ی در قسمت مشخص‌شده ذخیره شده است و حجم آن نیز ۲۳.۸ گیگابایت است، برای اینکه از این پشتیبان استفاده کنید باید بر روی آن دو بار کلیک کنید.



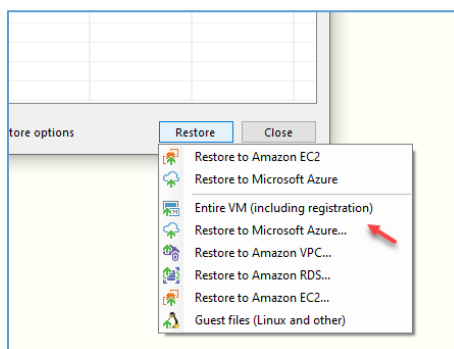
شکل ۵-۷۹ بررسی فایل پشتیبان

باتوجه به شکل ۵-۸۰، اندازه فایل پشتیبان و تاریخ پشتیبانی آن مشخص شده است که برای برگشت به آن روز باید فایل مورد نظر را از لیست، انتخاب و بر روی Restore کلیک کنید، با این کار بسته به سرعت شبکه و یا هارددیسک سرور ماشین مورد نظر شما برگشت داده خواهد شد



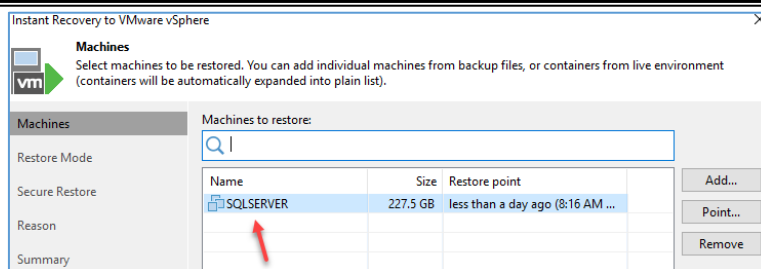
شکل ۵-۸۰ بررسی فایل پشتیبان

بعد از کلیک بر روی Restore منوی شکل ۵-۸۱ ظاهر خواهد شد که برای اینک ماشین مورد نظر را در همان مسیر Restore کنیم باید گزینه ی Entire VM را انتخاب کنید.



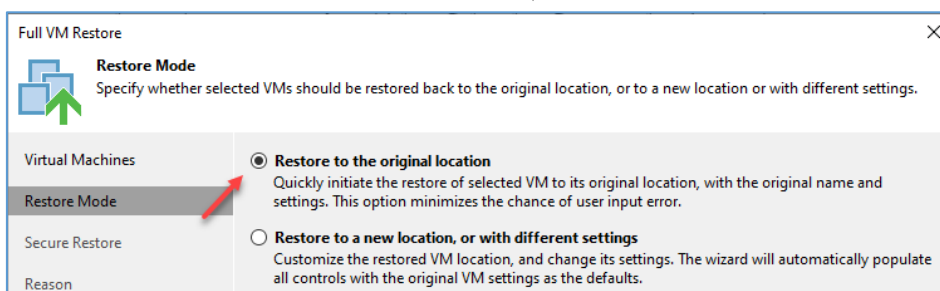
شکل ۵-۸۱ بررسی فایل پشتیبان

در شکل ۵-۸۲، آخرین بک آپ مربوط به ماشین مورد نظر را مشاهده می کنید که می توانید تاریخ مورد نظر خود را انتخاب کنید و بر روی Next کلیک کنید.



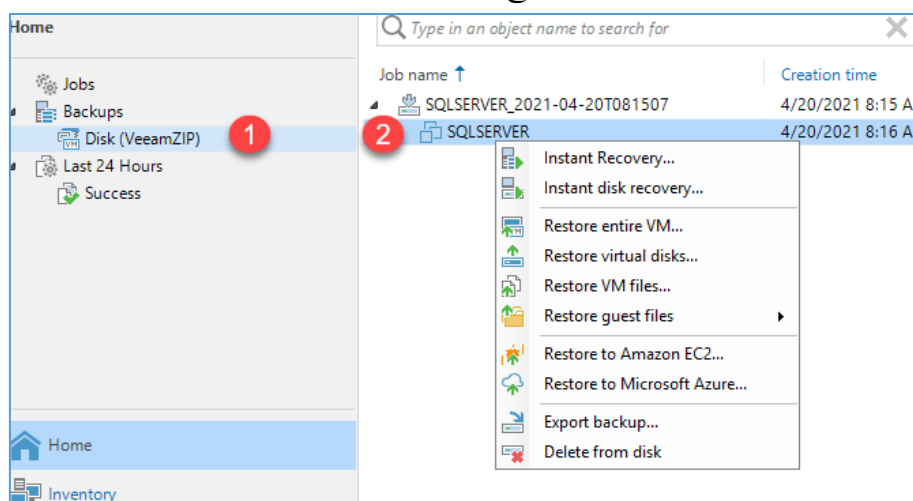
شکل ۵-۸۲ برگشت دادن فایل پشتیبان

در شکل ۵-۸۳ اگر گزینه‌ی اول را انتخاب کنید فایل پشتیبان دقیقاً جای همان ماشین مورد نظر را خواهد گرفت و اگر بخواهید مسیر آن را عوض کنید باید گزینه‌ی دوم را انتخاب کنید.



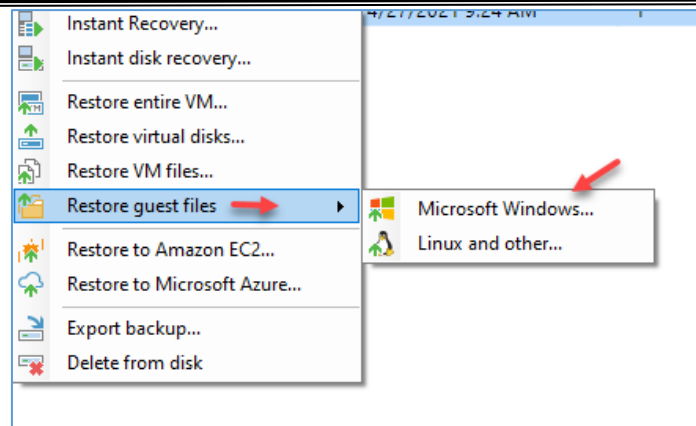
شکل ۵-۸۳ برگشت دادن فایل پشتیبان

روش دیگر برای استفاده از فایل بک آپ، این است که به مانند شکل ۵-۸۴ از قسمت شماره یک وارد Backup & Replication شوید و از قسمت شماره دو، Disk را انتخاب کنید؛ با انتخاب این گزینه، صفحه‌ی مربوط به فایل‌های پشتیبان‌گیری شده را مشاهده خواهید کرد، بر روی فایل مورد نظر کلیک راست کنید؛ در این منو گزینه‌های مختلف وجود دارد، با انتخاب گزینه اول می‌توانید خیلی سریع بک آپ مورد نظر را برگردانید.



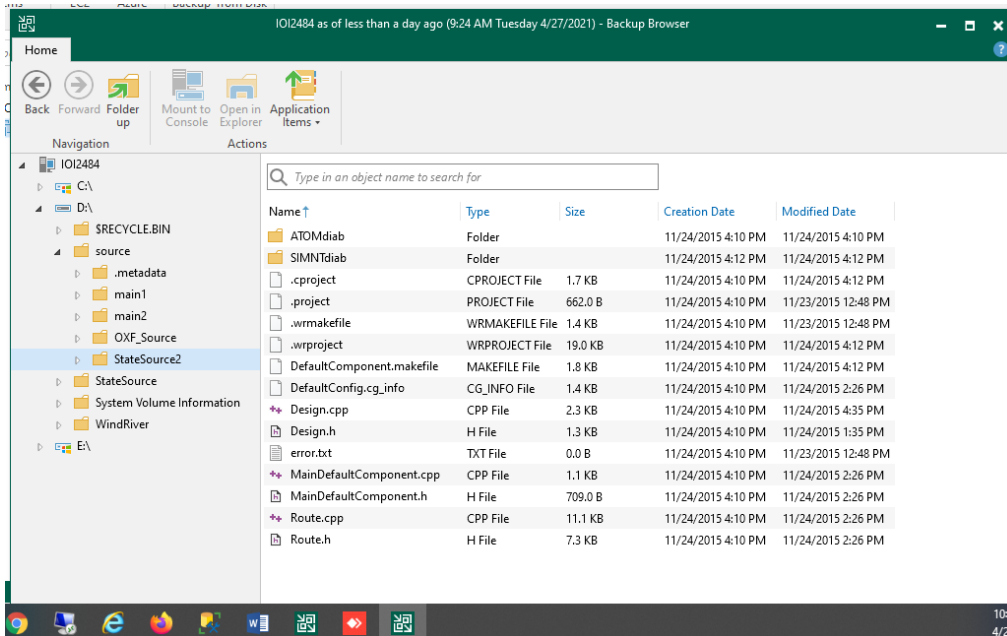
شکل ۵-۸۴ برگشت دادن فایل پشتیبان

اگر بخواهید اطلاعات ماشین مجازی را مشاهده کنید، مثلاً می‌خواهید یک فایل خاصی را از درون ماشین مجازی و از یکی از درایوهایش بردارید، برای این کار باید به مانند شکل ۵-۸۵ از قسمت Restore guest files گزینه‌ی Microsoft Windows را انتخاب کنید.



شکل ۵-۸۵- نمایش اطلاعات ماشین مجازی

همان‌طور که در شکل ۵-۸۶ مشاهده می‌کنید، فایل‌های مربوط به ماشین مجازی را مشاهده می‌کنید و اگر بخواهید فایل خاصی را برگردانید باید فایل مورد نظر را انتخاب کنید و در تب بالایی بر روی Restore کلیک کنید.



شکل ۵-۸۶- نمایش اطلاعات ماشین مجازی

این روش‌هایی که با هم بررسی کردیم، کار ساده‌ای است که این نرم‌افزار می‌تواند به‌خوبی انجام دهد، در ادامه بیشتر با ویژگی‌های قدرتمند آن آشنا خواهیم شد.

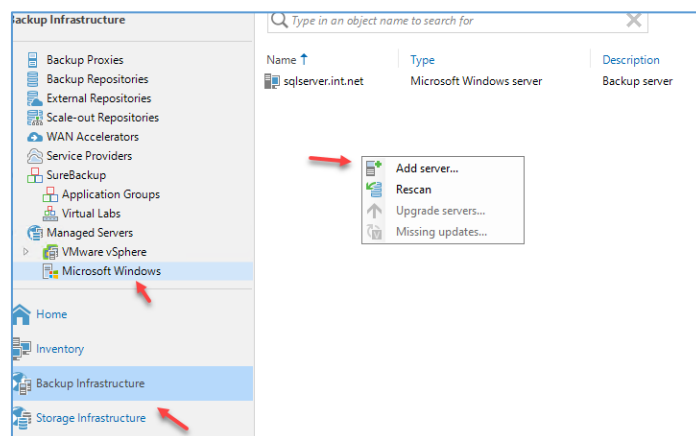
۵-۲-۳ اضافه کردن سرور Backup

در این قسمت می‌خواهیم کمی کار را باکیفیت‌تر کنیم؛ کار اصولی و درست این است که شما یک سرور پشتیبان یا همان Repository را به نرم‌افزار معرفی کنید و ماشین‌ها و اطلاعات را به آن سرور انتقال دهید، در این قسمت می‌خواهیم این کار را با هم انجام دهیم. سروری که برای این کار انتخاب می‌کنید باید از سرعت شبکه خوبی برخوردار باشد، یعنی باید حداقل سرعت گیگابایت داشته باشد تا در موقع پشتیبان‌گیری با عملکرد ضعیف روبرو نشوید، در بخش دیگر باید هارددیسک با ظرفیت بالا برای آن در نظر بگیرید تا با کمبود فضا روبرو نشوید، اگر برای شرکت

شما، هزینه ملاک نیست می‌توانید از هارد دیسک‌های SAS با سرعت بالا استفاده کنید که البته نیاز به سرورهایی دارد که آنها را پشتیبانی کند.

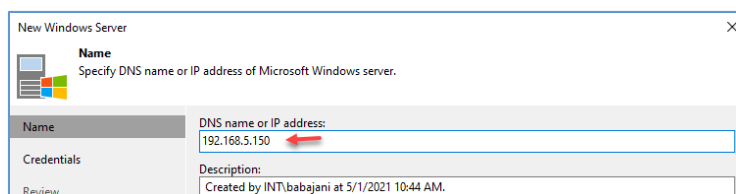
برای شروع کار می‌خواهیم یک سرور معمولی را به نرم‌افزار اضافه کنیم تا در ادامه کار بتوانیم به‌عنوان یک پشتیبان از آن استفاده کنیم.

به‌مانند شکل ۵-۸۷ وارد قسمت Backup Infrastructure شوید و از سمت چپ بر روی Microsoft Windows کلیک کنید، به‌خاطر این موضوع این گزینه را انتخاب کردیم که سیستم‌عامل ما از نوع ویندوز است، بر روی Add Server کلیک کنید.



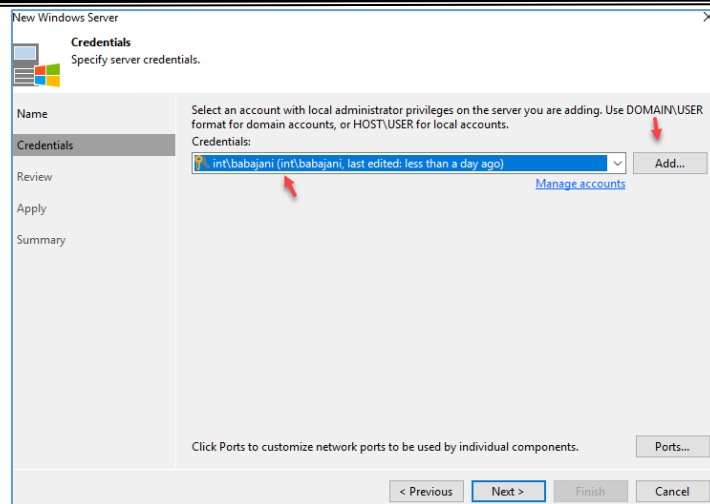
شکل ۵-۸۷ اضافه کردن سرور

در شکل ۵-۸۸ باید در قسمت مورد نظر، نام یا آدرس IP سرور خود که قرار است نقش پشتیبان را بازی کند را وارد کنید و بر روی Next کلیک کنید.



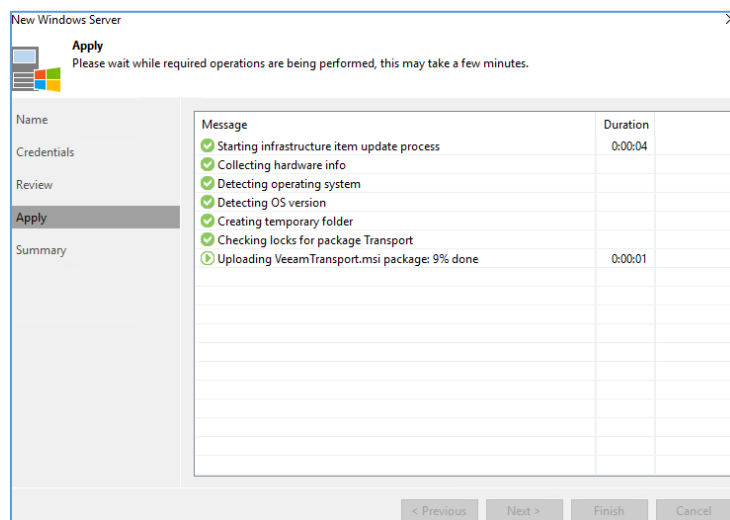
شکل ۵-۸۸ اضافه کردن سرور

در شکل ۵-۸۹ با کلیک بر روی Add باید نام کاربری را وارد کنید که دسترسی کامل به شبکه داشته باشد، اگر از قبل وارد کردید، می‌توانید آن را از لیست انتخاب کنید؛ در قسمت پایین صفحه، گزینه Port وجود دارد که پورت‌هایی را که نرم‌افزار نیاز دارد تا با این سرور در ارتباط باشد را مشخص می‌کند که البته قابل تغییر است.



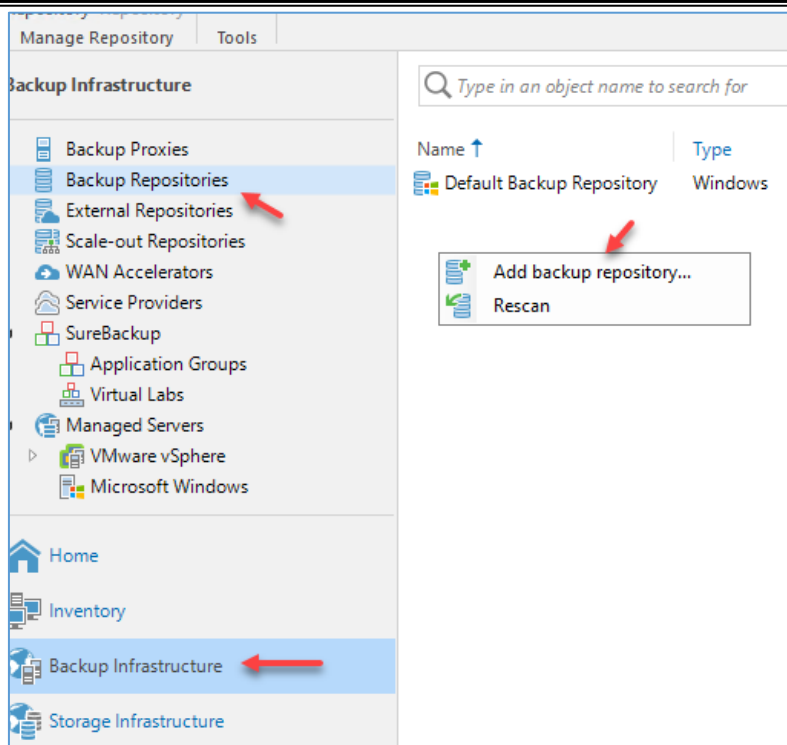
شکل ۵-۸۹ اضافه کردن سرور

همان‌طور که در شکل ۵-۹۰ مشاهده می‌کنید، اطلاعات در سرور FS در حال نصب است، اگر به شکل ۵-۹۰ دقت کنید، یک پکیج با نام Transport Veeam به سیستم FS انتقال داده شد و در حال نصب است که این پکیج، شامل سرویس‌های مربوط به نرم‌افزار Veeam است، بعد از پایان کار بر روی Finish کلیک کنید.



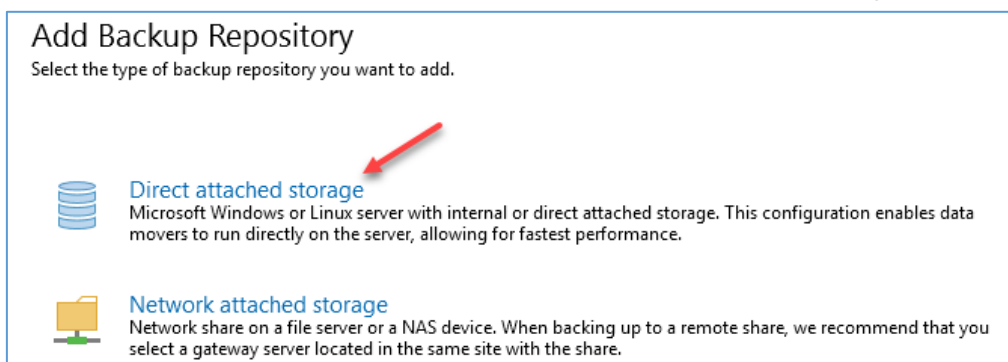
شکل ۵-۹۰ اضافه کردن سرور

در مرحله بعد باید سیستمی را که به عنوان پشتیبان اضافه کردید را در لیست Repositories یا همان، مخزن اضافه کنید تا بتوانید اطلاعات را به آن انتقال دهید، برای این کار به مانند شکل ۵-۹۱ وارد Backup Repositories شوید و در صفحه باز شده کلیک راست و گزینه‌ی Add backup repository را انتخاب کنید.



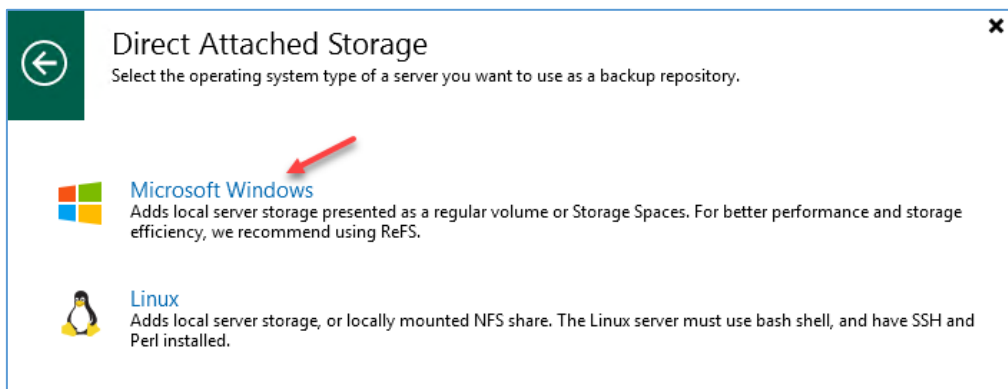
شکل ۹۱-۵ اضافه کردن Repository

در شکل ۹۲-۵ چند نوع مختلف Repository وجود دارد که برای این قسمت باید گزینه‌ی اول را انتخاب کنید.



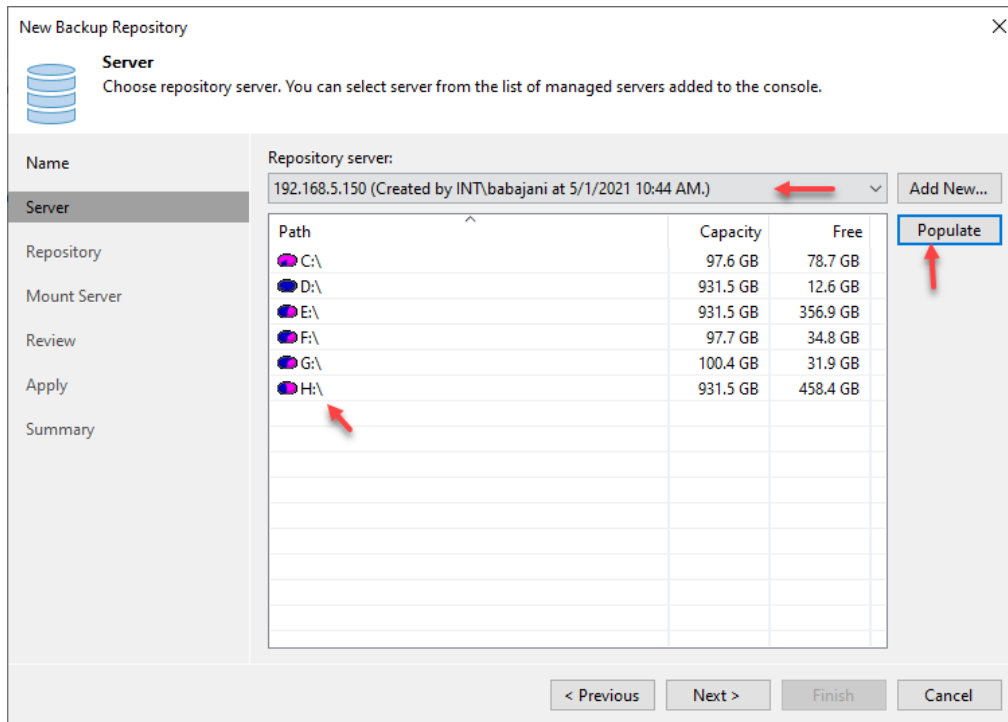
شکل ۹۲-۵ اضافه کردن Repository

در شکل ۹۲-۵ سیستم‌عامل ویندوز را انتخاب و بر روی Next کلیک کنید.



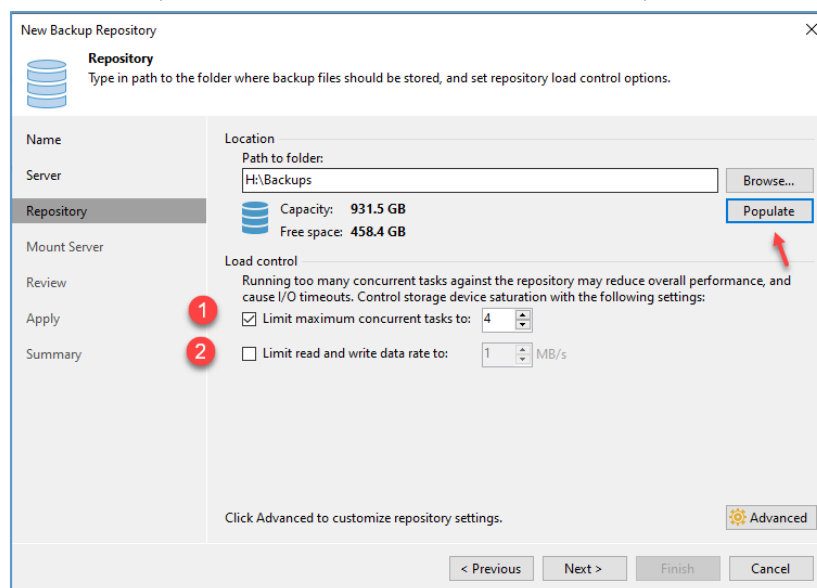
شکل ۹۲-۵ انتخاب سیستم‌عامل

در شکل ۵-۹۳ باید از قسمت Repository Server، نام سرور خود را انتخاب کنید، اگر بر روی Populate کلیک کنید، کلّ درایوهای سیستم مورد نظر را برای شما مشخص می‌کند، بعد از این کار بر روی Next کلیک کنید.



شکل ۵-۹۳ انتخاب Storage

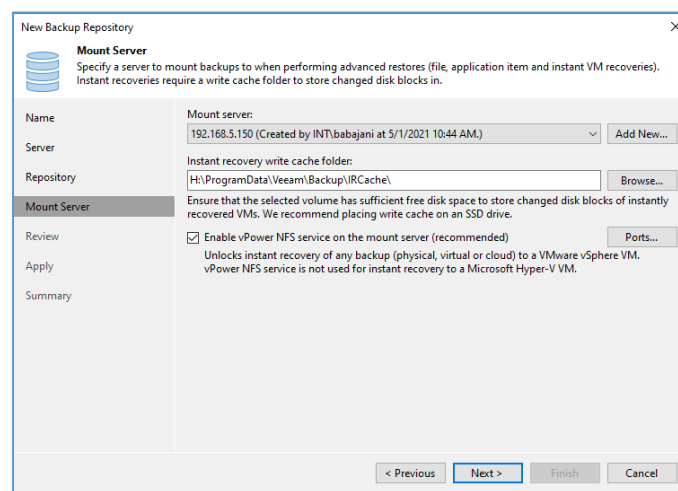
در شکل ۵-۹۴ باید درایو مورد نظر را در سرور انتخاب کنید، اگر بر روی Populate کلیک کنید، فضای خالی آن را مشاهده خواهید کرد، در قسمت شماره یک می‌توانید تعداد وظایف هم‌زمان را مشخص کنید، مثلاً در این شکل، عدد چهار در نظر گرفته شده است که هم‌زمان، چهار وظیفه می‌تواند بر روی این سیستم (FS) اجرا کرد.



شکل ۵-۹۴ اضافه کردن Storage

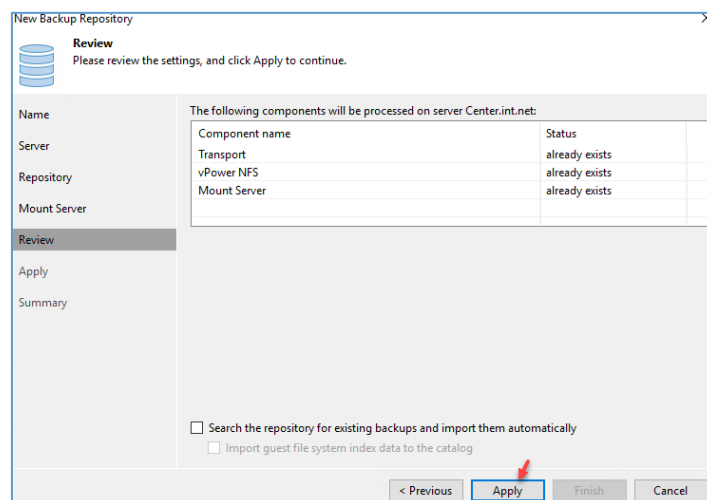
در قسمت شماره دو می‌توانید، مشخص کنید که سرعت خواندن و نوشتن در سیستم چقدر باشد، با این کار می‌توانید سرور را بهتر مدیریت کنید و از اینکه سیستم مورد نظر از شبکه خارج شود، جلوگیری کنید، در قسمت Advanced گزینه‌هایی وجود دارد که در صورت نیاز آنها را بررسی خواهیم کرد.

شکل ۹۵-۵ مربوط به Restore شدن اطلاعات و نمایش آن است که باید Mount Server را مشخص کنید که همان، سیستم FS با همان را انتخاب می‌کنیم، تیک گزینه مورد نظر را انتخاب کنید تا سرویس NFS بر روی سیستم شما فعال شود، توجه داشته باشید، اگر از قبل بر روی سیستم شما این سرویس نصب شده باشد در این قسمت این سرویس نصب نخواهد شد و با مشکل روبرو می‌شوید، در قسمت Folder نیز می‌توانید آدرسی برای ذخیره شدن اطلاعات مشخص کنید و بعد از این کار بر روی Next کلیک کنید.



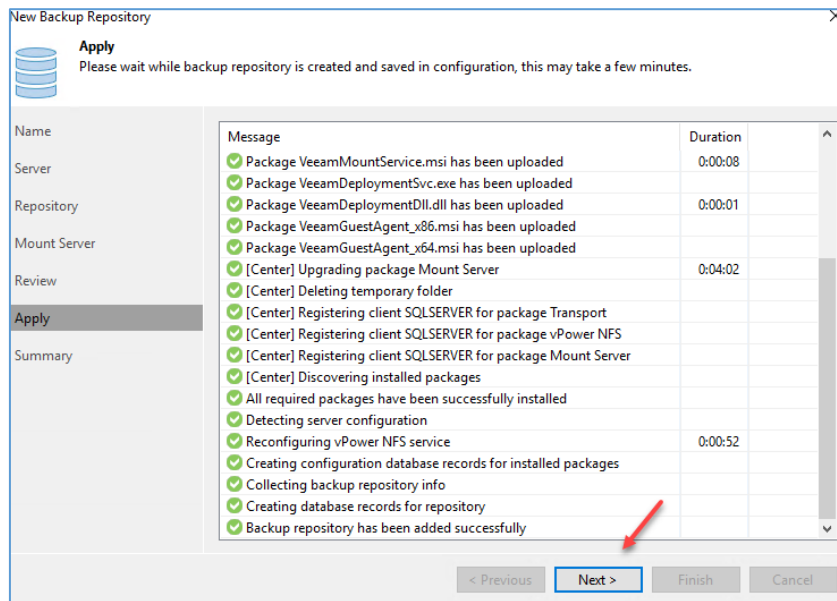
شکل ۹۵-۵ ایجاد Repository

در این صفحه، اطلاعات کلی را مشاهده می‌کنید، اگر تیک گزینه آخر، یعنی Import Ex... را انتخاب کنید، نرم‌افزار به صورت اتوماتیک از بک‌آپ‌های قبلی نیز استفاده خواهد کرد و اگر تیک بعدی انتخاب شود، فایل‌های Index را نیز به بک آپ اضافه می‌کند.



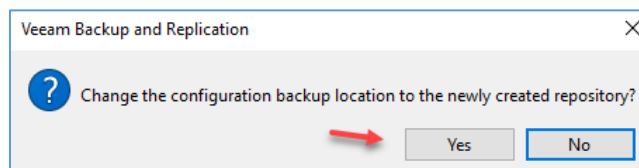
شکل ۹۶-۵ نصب Agent در ویندوز

همان‌طور که در شکل ۵-۹۷ مشاهده می‌کنید، نرم‌افزارهای مورد نظر در سرور نصب شده است.



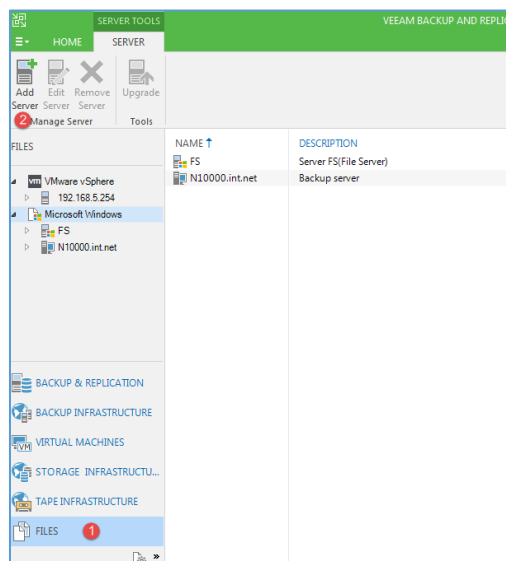
شکل ۵-۹۷ نصب Agent

در شکل ۵-۹۸، سؤال می‌کند که آیا مایل هستید Repository پیش‌فرض را تغییر دهید یا نه که باید بر روی Yes کلیک کنید.



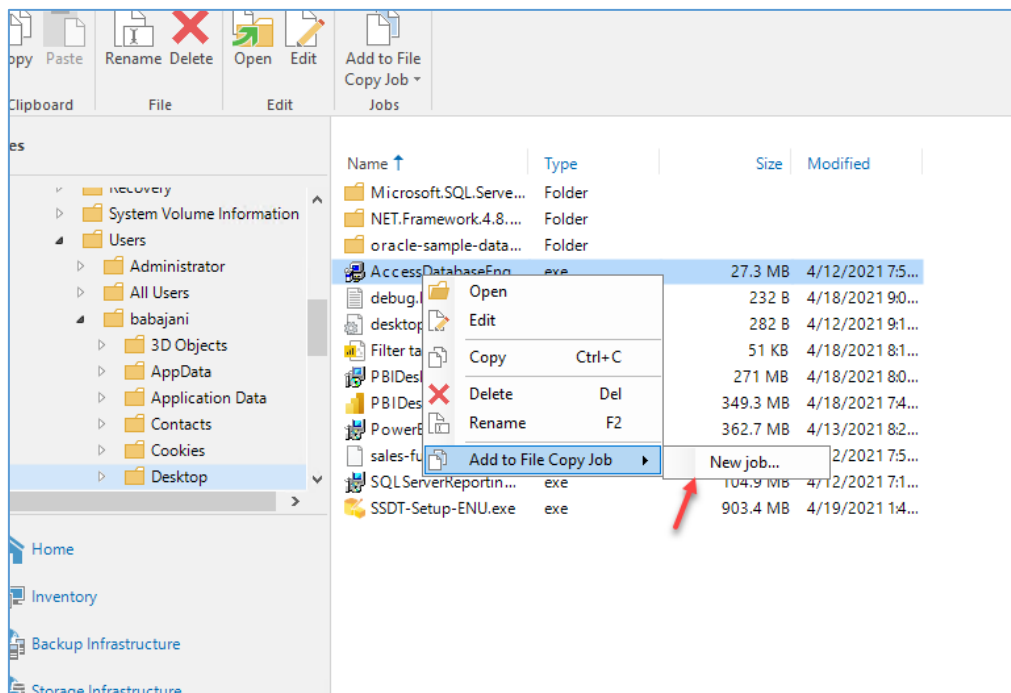
شکل ۵-۹۸ انتخاب Repository

بعد از انجام مراحل بالا برای تست به‌مانند شکل ۵-۹۹ وارد قسمت Files شوید، در این قسمت می‌توانید سرورها و سیستم‌های کلاینتی خود را به لیست اضافه کنید و از آنها پشتیبان تهیه کنید، برای تعریف سرور جدید و اضافه کردن آن به لیست باید بر روی Add Server کلیک کنید.



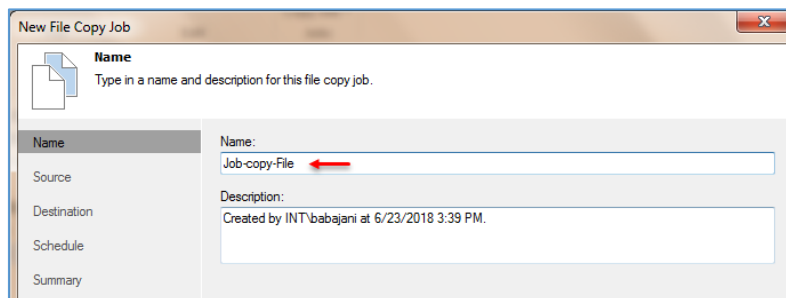
شکل ۵-۹۹ بررسی سرور

در شکل ۵-۱۰۰ وارد یکی از سیستم‌ها که به لیست اضافه کردیم، شدید؛ بر روی یکی از فایل‌ها که داخل درایو آن است، کلیک راست کنید و گزینه Add to File Copy Job و سپس گزینه New job را انتخاب کنید.



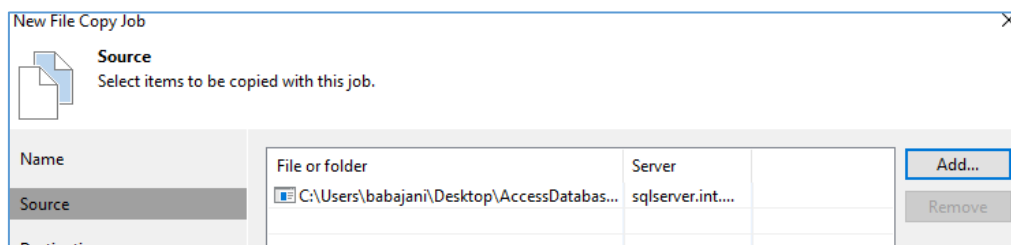
شکل ۵-۱۰۰

در شکل ۵-۱۰۱، نام مورد نظر خود را وارد و بر روی next کلیک کنید.



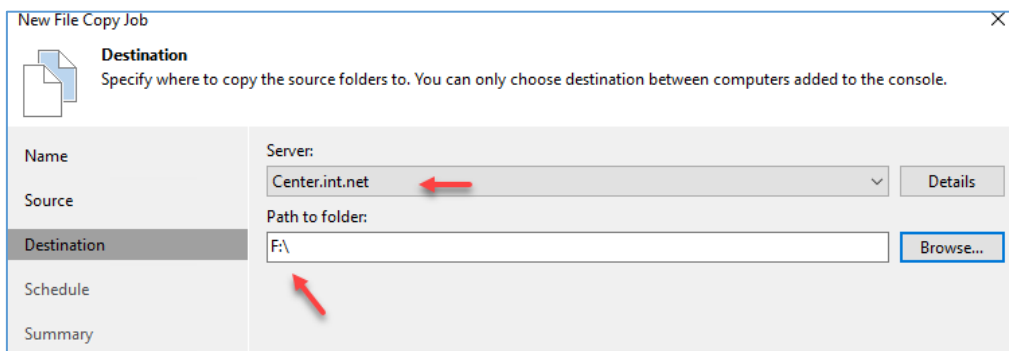
شکل ۵-۱۰۱/ ایجاد job

در شکل ۵-۱۰۲ باید فایل‌های منبع خود را انتخاب کنید تا یک پشتیبان از آن تهیه کنید، برای اضافه کردن فایل جدید می‌توانید بر روی Add کلیک کنید.



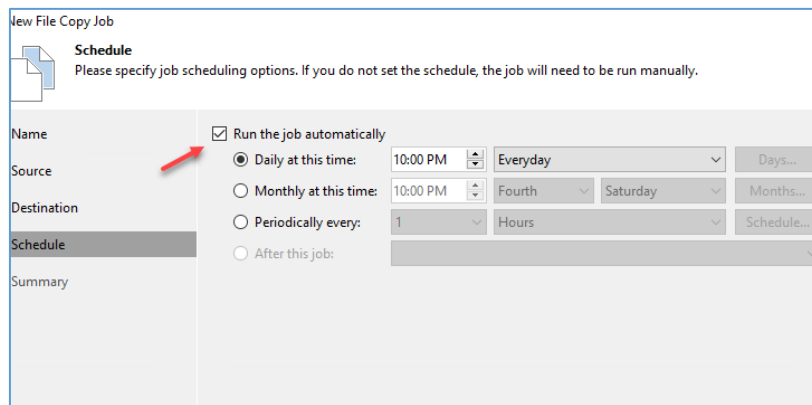
شکل ۵-۱۰۲/ ایجاد job

در شکل ۵-۱۰۳ باید سرور مقصد که در قسمت قبل به لیست اضافه کردیم را انتخاب کنید، سپس در قسمت Path to folder، هارددیسک داخل سرور را انتخاب و بر روی Next کلیک کنید.



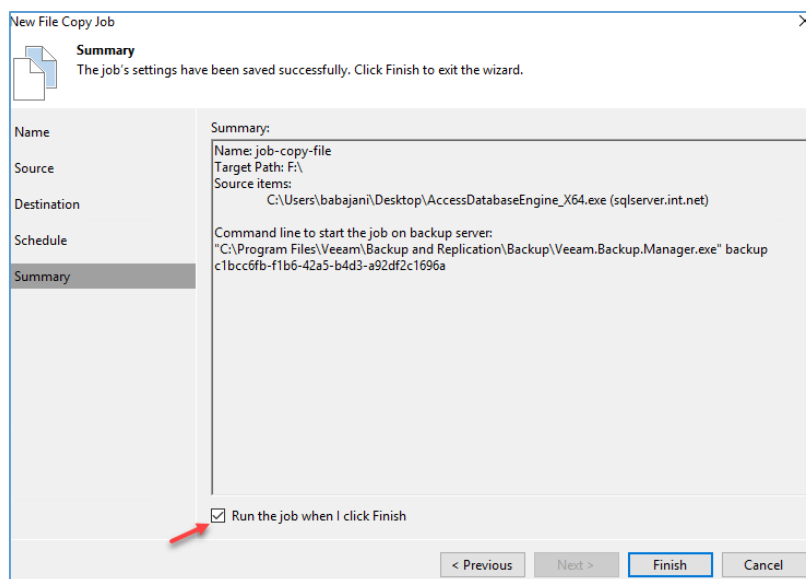
شکل ۵-۱۰۳ ایجاد job

در شکل ۵-۱۰۴ باید زمان انجام این کار را مشخص کنید تا به صورت اتوماتیک در زمان مشخص شده، کار پشتیبان-گیری انجام شود، سعی کنید زمان آن را در ساعات بیکاری سرور در نظر بگیرید تا مشکلی در شبکه پیش نیاید؛ بر روی Save کلیک کنید.



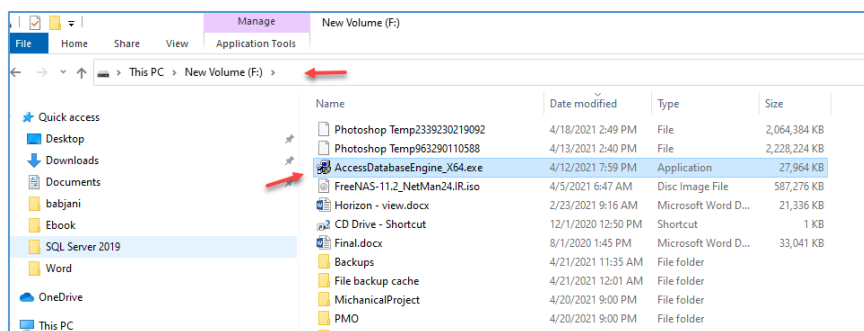
شکل ۵-۱۰۴ ایجاد job

اگر می‌خواهید بعد از بستن این پنجره، کار پشتیبان‌گیری آغاز شود باید تیک گزینه 'Run the job when...' را به مانند شکل ۵-۱۰۵ انتخاب و بر روی Finish کلیک کنید.



شکل ۵-۱۰۵ ایجاد job

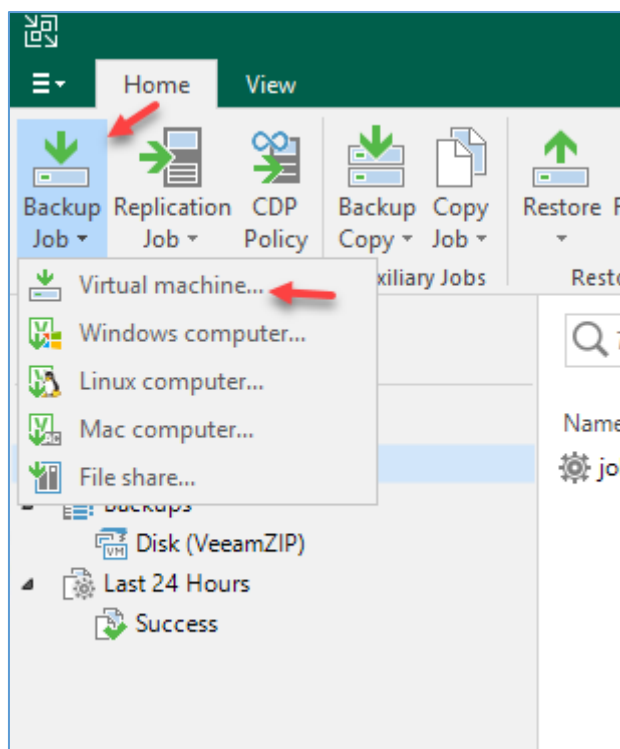
اگر به شکل ۵-۱۰۶ دقت کنید فایل مورد نظر که برای انتقال انتخاب کرده بودیم در مسیر مشخص شده کپی شده است، حالا شما می‌توانید هر فایل را برای این کار انتخاب کنید، فقط توجه داشته باشید که فایل‌هایی که انتخاب می‌کنید توسط پردازش دیگر در حال استفاده نباشد که با خطا روبرو خواهید شد.



شکل ۵-۱۰۶ انتقال فایل

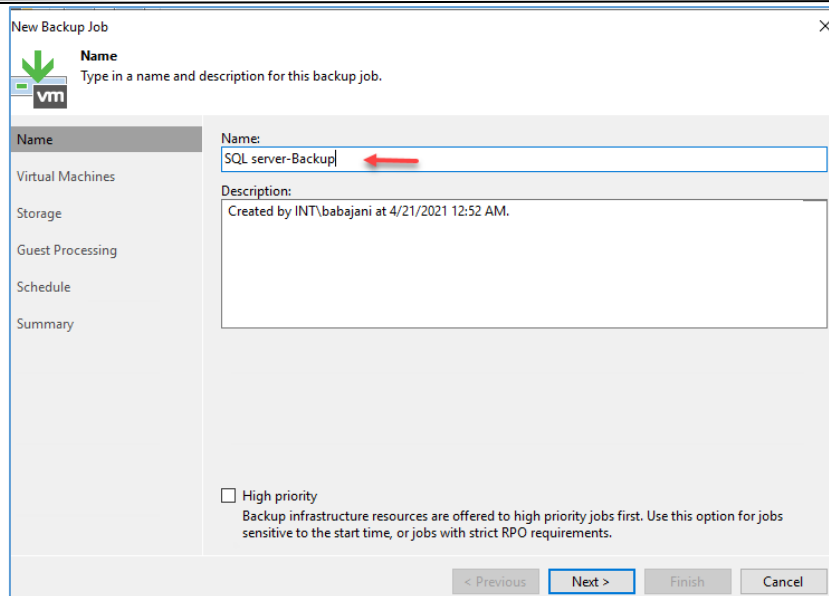
۵-۲-۴ پشتیبان‌گیری از ماشین مجازی

خوب در ادامه می‌خواهیم از ماشین مجازی پشتیبان تهیه کنیم، برای این کار به مانند شکل ۵-۱۰۷ وارد Backup & Replication شوید و از منوی بالایی بر روی Backup Job کلیک کنید و بعد گزینه‌ی Virtual machine را انتخاب کنید.



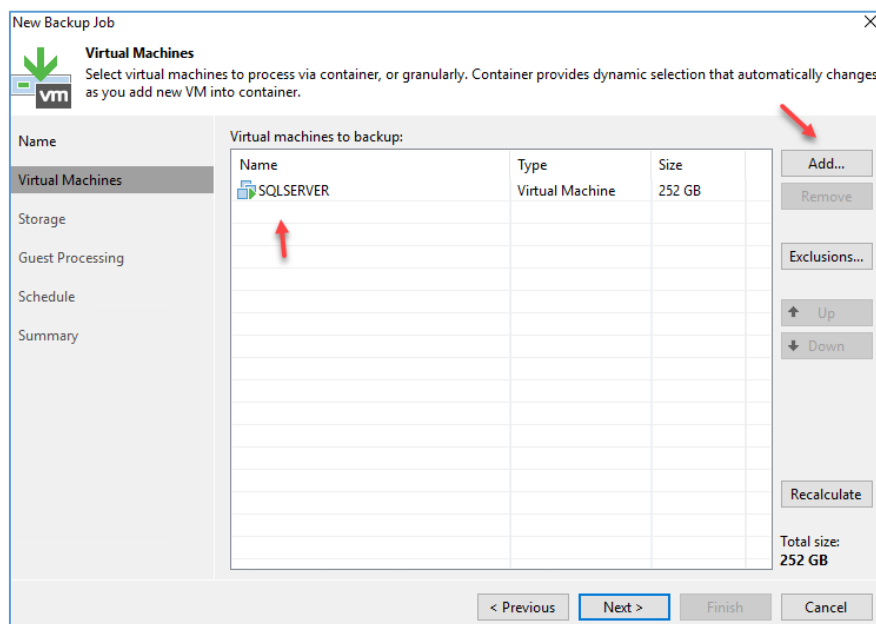
شکل ۵-۱۰۷ اضافه کردن ماشین مجازی

در شکل ۵-۱۰۸ یک نام به دلخواه وارد و بر روی Next کلیک کنید.



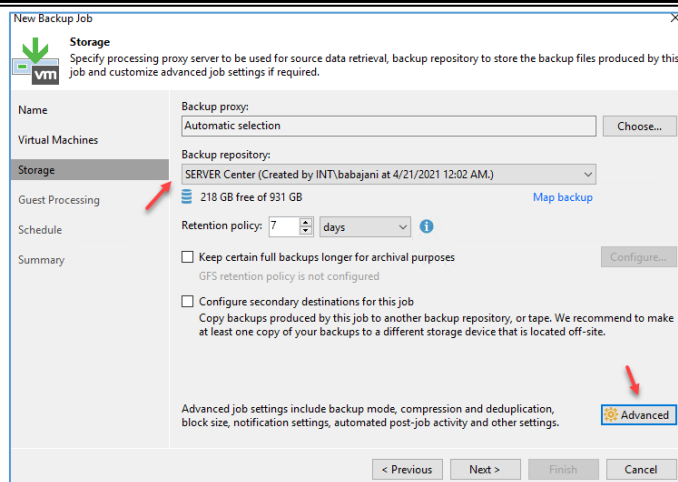
شکل ۵-۱۰۸ اضافه کردن ماشین مجازی

در شکل ۵-۱۰۹ می‌توانید تمام ماشین‌های مجازی یا تنها، یکی را به لیست اضافه کنید، برای این کار باید بر روی Add کلیک کنید.



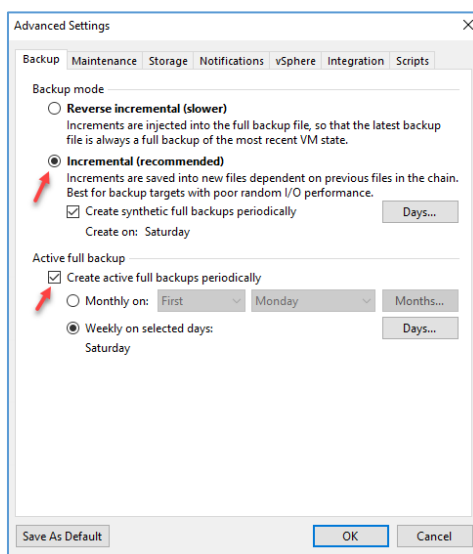
شکل ۵-۱۰۹ اضافه کردن ماشین مجازی

در شکل ۵-۱۱۰ در قسمت Repository Backup باید سرور مورد نظر خود را انتخاب کنید که بعد از انتخاب، مقدار فضای خالی از کل هارد را در زیر آن نمایش می‌دهد، اگر تیک گزینه 'Configure secondary...' را انتخاب کنید، می‌توانید در ادامه یک سرور Repository دیگری برای گرفتن پشتیبان دوم انتخاب کنید.



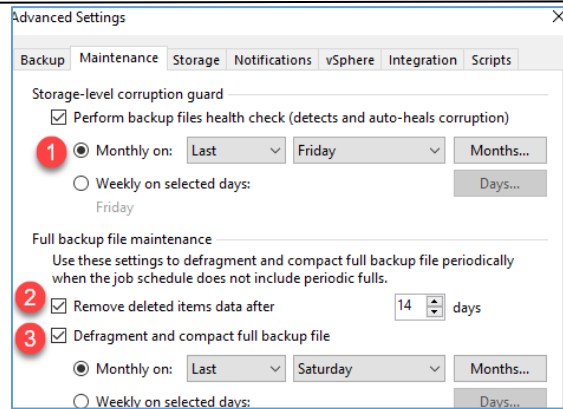
شکل ۱۱۰-۵ اضافه کردن ماشین مجازی

بعد از انجام مراحل بالا بر روی Advanced کلیک کنید تا این گزینه را نیز بررسی کنیم، به مانند شکل ۱۱۱-۵ در قسمت Advanced و در تب Backup، گزینه Incremental را برای انجام پشتیبان گیری از تغییرات انتخاب کنید و بعد، گزینه Create active full backups را انتخاب کنید تا یک هفته، یک بار، یا چند بار از ماشین مورد نظر پشتیبان کامل تهیه کند.



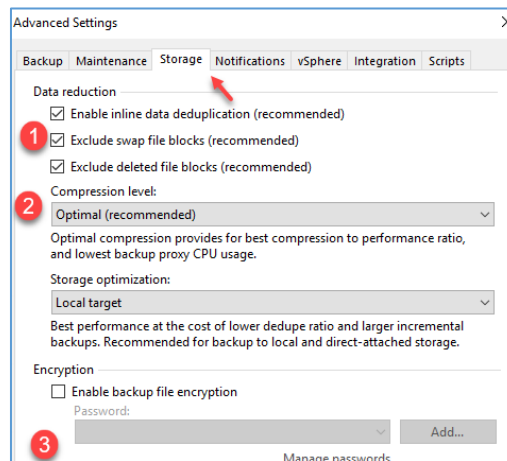
شکل ۱۱۱-۵ بررسی قسمت Advanced

به مانند شکل ۱۱۲-۵ در قسمت Maintenance، شماره ۱، برای اینکه از وضعیت سالم بودن فایل های پشتیبان خود مطلع شویم، این گزینه را فعال می کنیم، در قسمت شماره ۲ دو می توانید بر اساس روزهایی که نیاز به فایل پشتیبان دارید، یک عدد به جای عدد ۱۴ وارد کنید که تعداد روز نگهداری از فایل پشتیبان را به شما اعلام می کند، در قسمت شماره ۳، اگر این گزینه را فعال کنید، فایل های پشتیبان، Defragment خواهند شد تا فضای اضافی بین آنها کم شود.



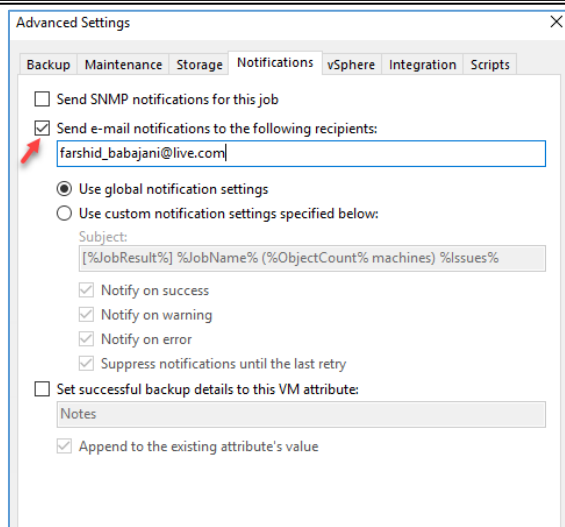
شکل ۱۱۲-۵ بررسی قسمت Advanced

به‌مانند شکل ۱۱۳-۵ در تب Storage و در قسمت اول می‌توانید تیک هر سه گزینه را که یک‌سری دستورالعمل برای کاهش حجم فایل پشتیبان است را انتخاب کنید، در قسمت شماره دو می‌توانید طبق سرعت پردازنده خودتان، مقدار فشرده‌سازی فایل‌های پشتیبان را انتخاب کنید و در گزینه پایین آن می‌توانید به نسبت سرعت شبکه، یکی از گزینه‌ها را انتخاب کنید، مثلاً اگر از شبکه داخلی استفاده می‌کنید، می‌توانید گزینه LAN را انتخاب کنید، در قسمت شماره سه می‌توانید رمز عبور بر روی فایل‌های پشتیبان خود قرار دهید.

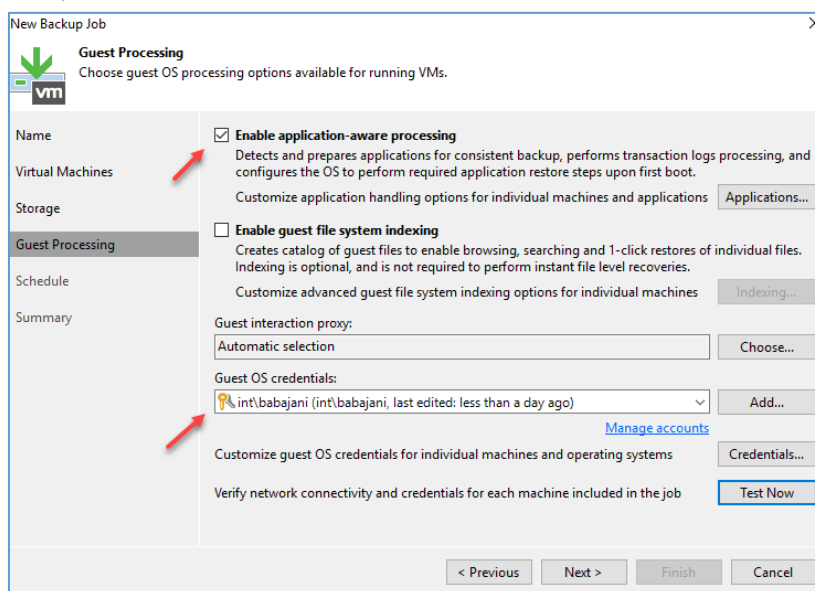


شکل ۱۱۳-۵ بررسی قسمت Advanced

به‌مانند شکل ۱۱۴-۵ در تب Notification می‌توانید یک آدرس ایمیل وارد کنید تا تمام رویدادهایی که در این پشتیبانی انجام می‌شود، برای شما ایمیل شود. بقیه تب‌ها را در صورت نیاز بررسی خواهیم کرد.

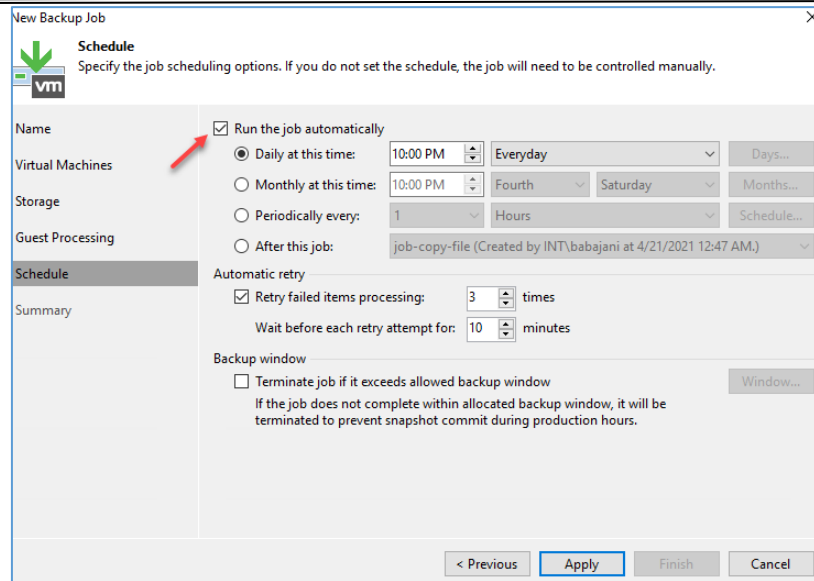
شکل ۱۱۴-۵ بررسی قسمت *Advanced*

در شکل ۱۱۵-۵ با فعال کردن این دو گزینه می‌توانید به نرم‌افزار این امکان را دهید تا بر روی سرورهای سرویس‌ها، مانند: Active Directory یا بر روی جدول‌های SQL تغییرات ایجاد کند که در ادامه این کار را انجام خواهیم داد.



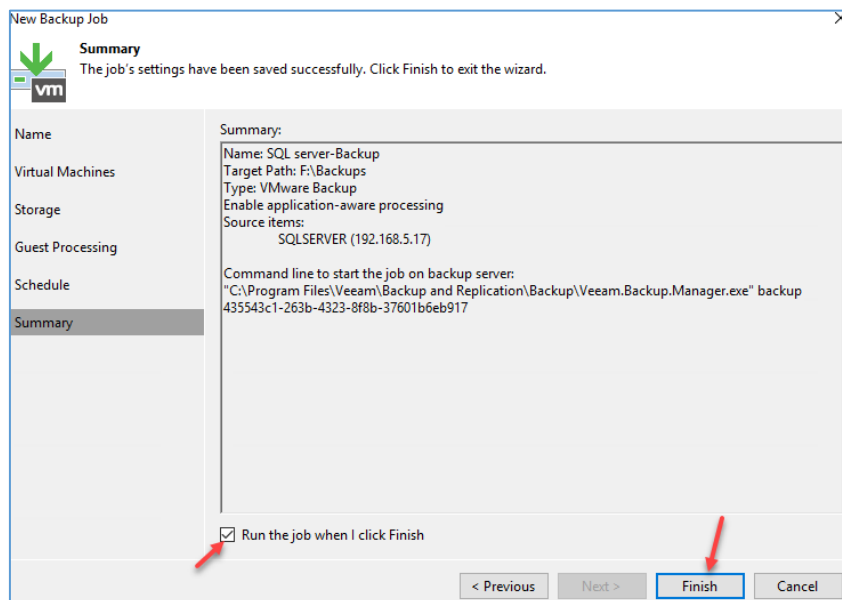
شکل ۱۱۵-۵ اضافه کردن ماشین مجازی

شکل ۱۱۶-۵ در این قسمت باید زمان انجام کار را مشخص کنید، بهتر است زمان مورد نظر در ساعات بیکاری شبکه باشد؛ بر روی Save کلیک کنید.



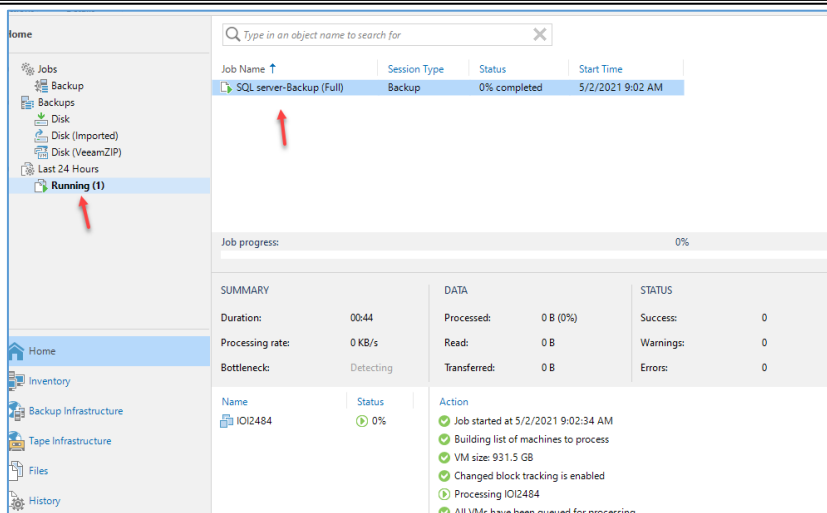
شکل ۵-۱۱۶ اضافه کردن ماشین مجازی

در شکل ۵-۱۱۷، تیک گزینه مورد نظر را انتخاب کنید تا بعد از بسته شدن صفحه، کار پشتیبان‌گیری آغاز شود.



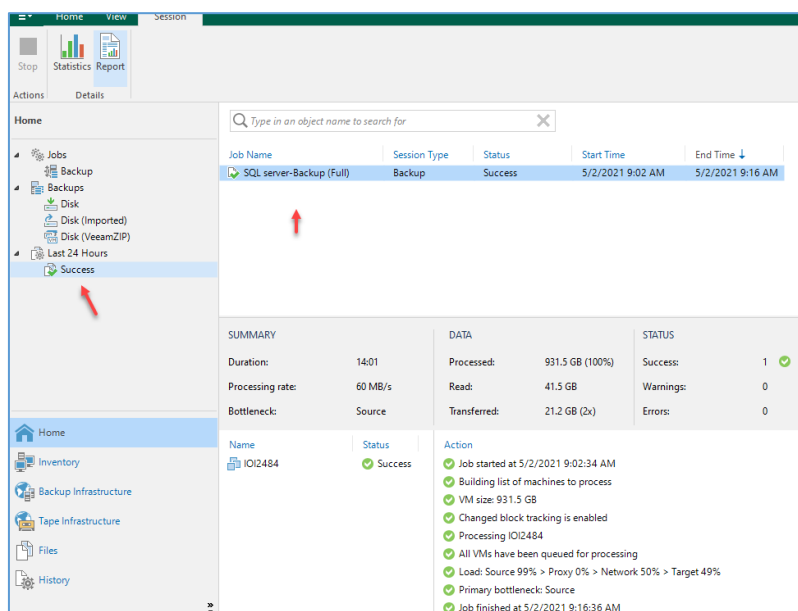
شکل ۵-۱۱۷ اضافه کردن ماشین مجازی

همان‌طور که در شکل ۵-۱۱۸ مشاهده می‌کنید، فایل پشتیبان در حال تکمیل است.



شکل ۵-۱۱۸/ ایجاد پشتیبان از ماشین مجازی

بعد از انجام شدن پشتیبان، به مانند شکل ۵-۱۱۹ در قسمت Success، می توانید موفق بودن کار را مشاهده کنید.



شکل ۵-۱۱۹/ پشتیبان گیری از ماشین مجازی

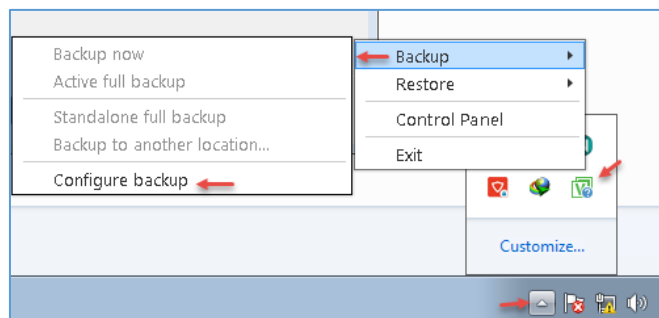
۵-۲-۵ استفاده از Veam Agent

یکی از ابزارهای فوق العاده نرم افزار Veam، استفاده از سرویس Agent است که به شما این امکان را می دهد که از تمام اطلاعات کلاینت، یک پشتیبان تهیه کنید، یعنی اینکه با نصب این نرم افزار از سیستم های کاربران با تنظیمی که بر روی آنها انجام می دهید، پشتیبان تهیه خواهد شد و با ازدست دادن اطلاعات کاربری، مشکلی برای برگشت نخواهید داشت، یعنی یک شبکه کاملاً قابل اعتماد.

برای دانلود Veam Agent به لینک زیر مراجعه کنید:

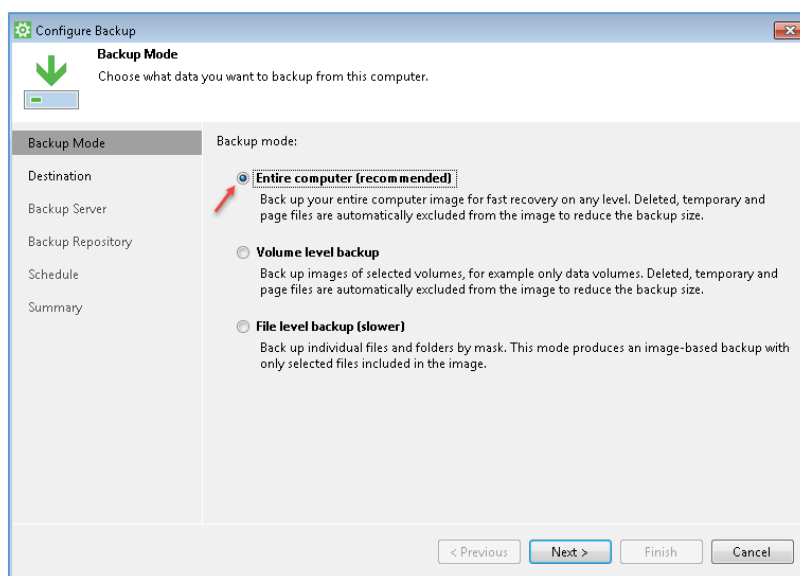
<https://www.veeam.com/downloads.html>

بعد از نصب به‌مانند شکل ۵-۱۲۳ در Taskbar بر روی آیکون آن کلیک راست کنید و از قسمت Backup، گزینه Configure Backup را انتخاب کنید.



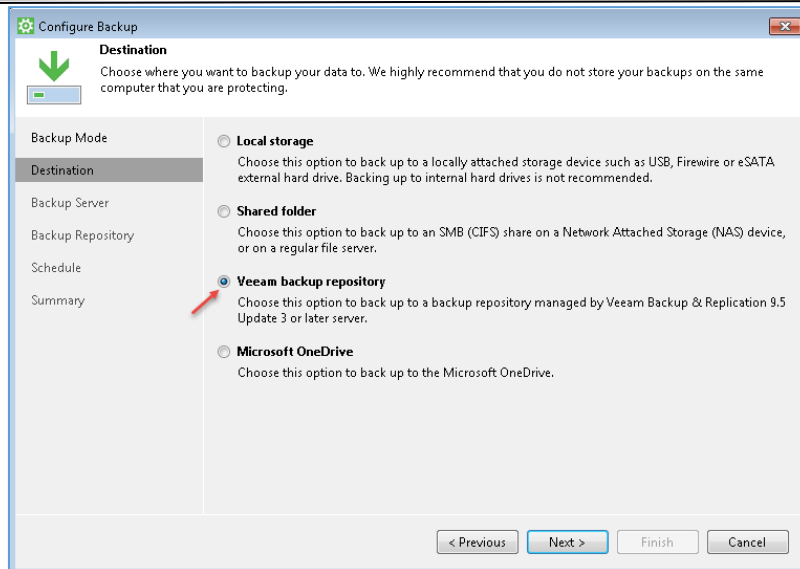
شکل ۵-۱۲۳ تنظیم Veeam Agent

در شکل ۵-۱۲۴، سه گزینه را مشاهده می‌کنید که اگر گزینه اول انتخاب شود، از کل اطلاعات سیستم، پشتیبان تهیه خواهد کرد، یعنی از همه پارتیشن‌های موجود؛ در گزینه دوم می‌توانید پارتیشن مورد نظر خود را به‌صورت دستی انتخاب کنید و در قسمت آخر نیز می‌توانید فایل‌های مشخص خود را در پارتیشن انتخاب کنید؛ در حال حاضر، گزینه اول را انتخاب و بر روی Next کلیک کنید.



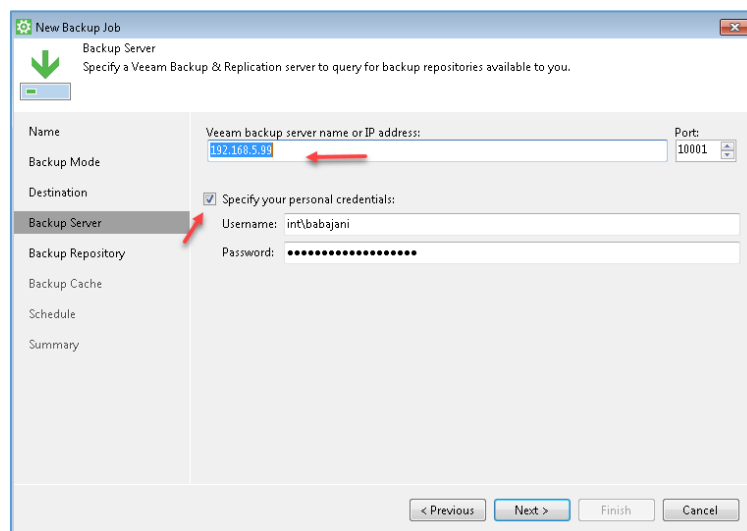
شکل ۵-۱۲۴ تنظیم Veeam agent

در شکل ۵-۱۲۵ باید مسیر ذخیره‌سازی را انتخاب کنید که این محل می‌تواند بر روی هارد دیسک خود سیستم باشد، یا بر روی شبکه یا بر روی مایکروسافت ondrive که البته در اینجا می‌خواهیم از گزینه سوم، یعنی backup Veeam Repository که قبلاً با هم ایجاد کردیم، استفاده کنیم.



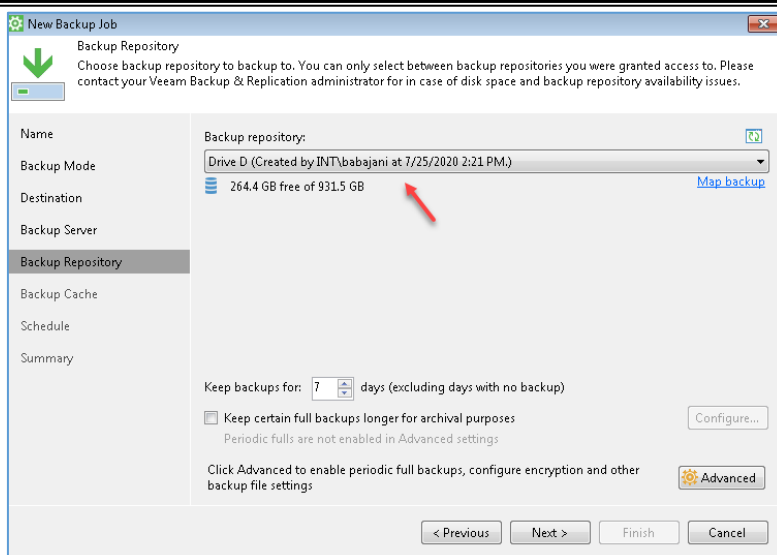
شکل ۵-۱۲۵ تنظیم Veeam agent

در شکل ۵-۱۲۶، باید آدرس سروری که بر روی آن، نرم‌افزار Backup & Repository را نصب کردید را وارد کنید و یک نام کاربری با دسترسی کامل را وارد کنید، اگر در موقع نصب نرم‌افزار، پورت آن را تغییر دادید، باید در این قسمت نیز تغییر دهید و بر روی Next کلیک کنید.



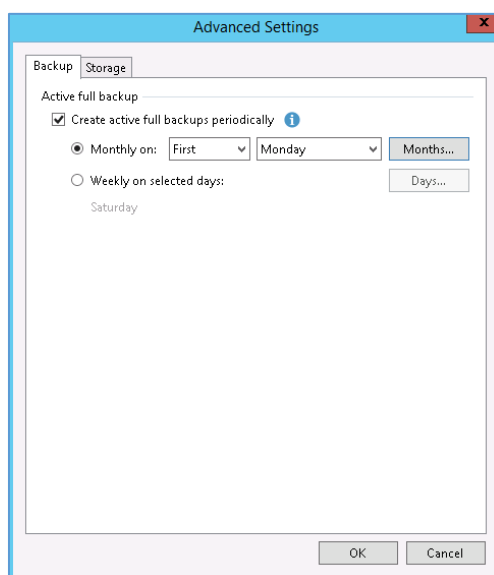
شکل ۵-۱۲۶ تنظیم Veeam agent

در شکل ۵-۱۲۷ باید Repository مورد نظر خود را انتخاب کنید، در قسمت پایین، مدت زمان نگهداری از فایل پشتیبان را که به صورت پیش فرض، ۱۴ روز است را تغییر دهید، بر روی Advanced کلیک کنید.



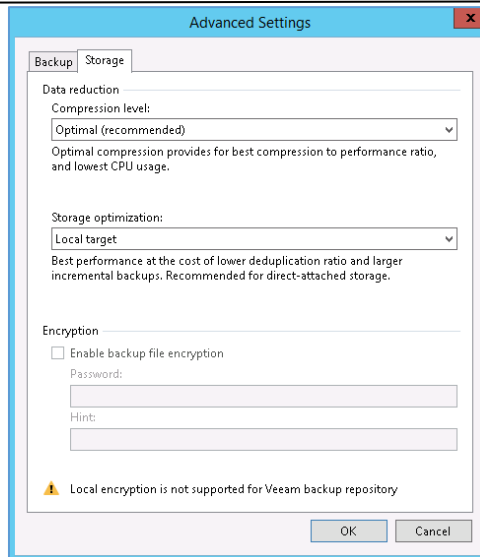
شکل ۵-۱۲۷ تنظیم Veeam agent

به‌مانند شکل ۵-۱۲۸، در تب Backup می‌توانید تعداد دفعات گرفتن Backup Full را در طول ماه یا در طول هفته مشخص کنید که در این قسمت باید تیک گزینه‌ی مورد نظر را فعال کنید.



شکل ۵-۱۲۸ تنظیم Veeam agent

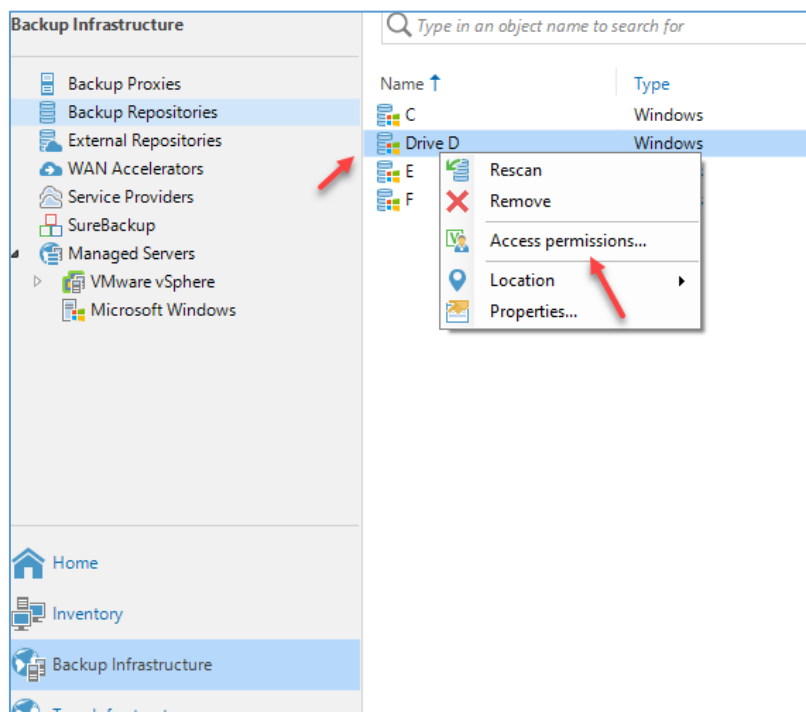
در تب Storage می‌توانید حالت فشرده‌سازی و سرعت ذخیره‌سازی آن را مشخص کنید و بر روی آن، رمز عبور قرار دهید که البته قبلاً این مورد را انجام دادیم.



شکل ۵-۱۲۹ تنظیم Veeam agent

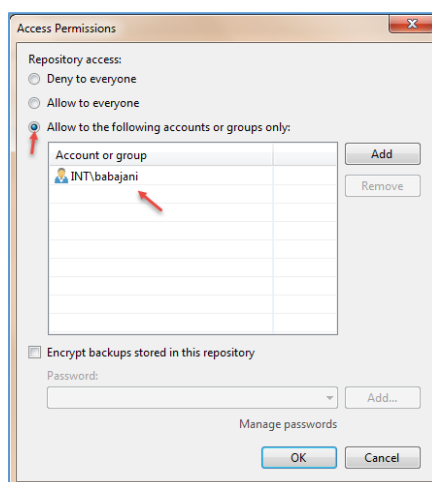
یک نکته‌ی مهم قبل از ادامه‌ی این قسمت، این است که زمانی که به Repository متصل می‌شوید باید یک دسترسی به آن منبع را برای این کاربر یا کاربرانی که در حال متصل شدن به آن هستند، مشخص کنید؛ برای این کار دوباره وارد سرور Veeam شوید.

در شکل ۵-۱۳۰، از سمت چپ بر روی Infrastructure Backup کلیک کنید و در قسمت Repository Backup، گزینه‌ی Permissions Access مورد نظر خود را که از قبل ایجاد کردید را انتخاب کنید و بر روی آن کلیک راست و گزینه‌ی Permissions Access را انتخاب کنید.



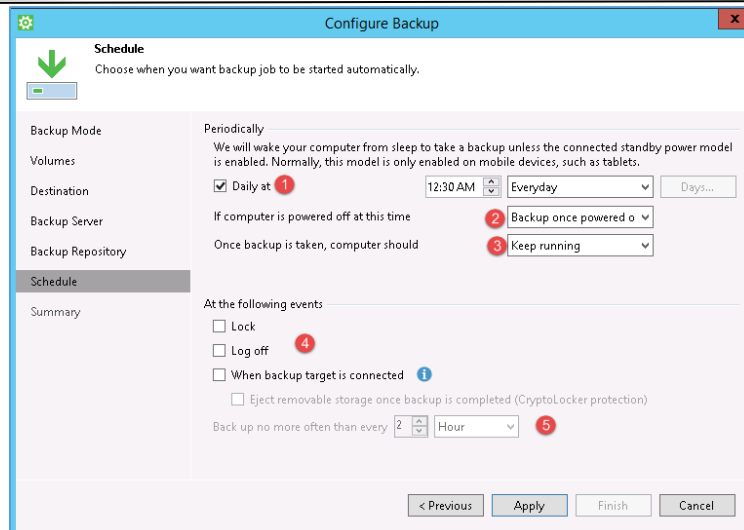
شکل ۵-۱۳۰ تنظیم دسترسی Veeam agent

در شکل ۵-۱۳۱ باید دسترسی لازم را به کاربر مورد نظر دهید، البته می‌توانید گزینه *Allow to everyone* را انتخاب کنید که با فعال کردن آن، همه می‌توانند به آن دسترسی داشته باشند که از نظر امنیتی کار جالبی نیست، بهتر است گزینه *سوّم* را به‌مانند شکل روبرو انتخاب کنید و بر روی *Add* کلیک کنید و کاربر مورد نظر را به لیست اضافه کنید، در قسمت پایین صفحه می‌توانید بر روی فایل‌هایی که در این *Repository* قرار می‌گیرند، رمز قرار دهید.



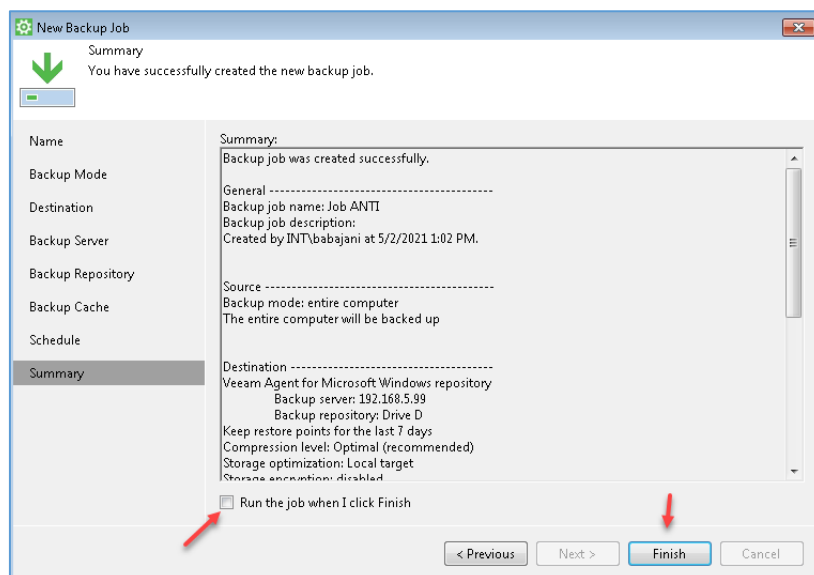
شکل ۵-۱۳۱ دسترسی به کاربر

در شکل ۵-۱۳۲ در قسمت شماره‌ی یک می‌توانید زمان پشتیبان‌گیری را مشخص کنید، در قسمت شماره‌ی دو اگر گزینه‌ی اول را انتخاب کنید، در هنگام شروع پشتیبان‌گیری اگر سیستم خاموش باشد، بعد از روشن شدن آن، پشتیبان‌گیری آغاز خواهد شد و اگر گزینه‌ی دوم را انتخاب کنید، این کار انجام نخواهد شد، در قسمت شماره‌ی سه نیز می‌توانید نوع از دست رفتن سیستم را انتخاب کنید که بهترین گزینه، همان *Keep Running* است، در قسمت شماره‌ی چهار می‌توانید نوع واکنش سیستم را در هنگام شروع به کار پشتیبان‌گیری مشخص کنید که سیستم را *Lock* یا *Log off* کنید و در قسمت شماره‌ی پنج، اگر این گزینه را فعال کنید می‌توانید مقدار زمان انجام پشتیبان‌گیری را مشخص کنید تا زمان زیادی را درگیر کار پشتیبان‌گیری نباشد و سیستم را خسته نکند، بر روی *Apply* و بعد بر روی *Yes* کلیک کنید.



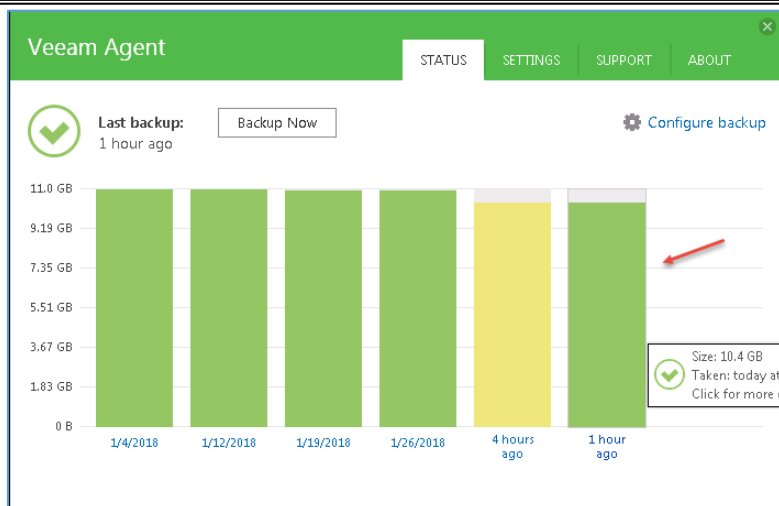
شکل ۵-۱۳۲ تنظیم زمان‌بندی Agent

در شکل ۵-۱۳۳، اطلاعات نهایی را مشاهده کنید و بر روی **Finish** کلیک کنید، البته اگر بخواهید بعد از بستن این پنجره، کار پشتیبان‌گیری آغاز شود، تیک مورد نظر را انتخاب و بر روی **Finish** کلیک کنید.



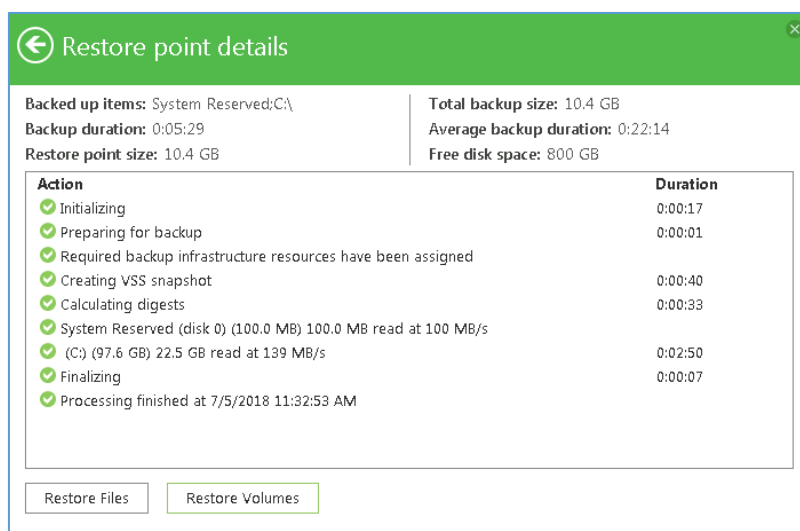
شکل ۵-۱۳۳ تنظیم Veeam Agent

همان‌طور که در شکل ۵-۱۳۴ مشاهده می‌کنید، عملیات پشتیبان‌گیری از ویندوز مورد نظر و ارسال آن به سیستم Veeam با موفقیت انجام شد، اگر بر روی هر یک از ستون‌های مورد نظر کلیک کنید می‌توانید اطلاعات کامل‌تری از پشتیبان‌گیری انجام‌شده به دست آورید.



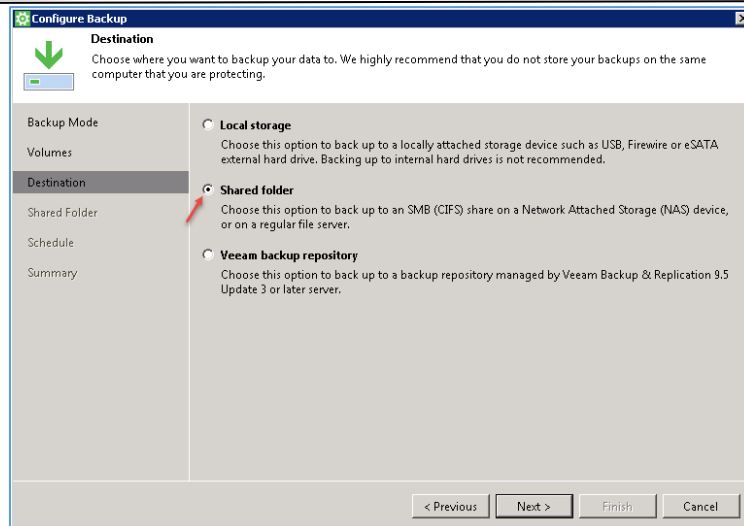
شکل ۱۳۴-۵ اجرا نرم‌افزار Veeam Agent

همان‌طور که در شکل ۱۳۵-۵ مشاهده می‌کنید، مراحل ساخت و ایجاد فایل پشتیبان نمایش داده شده است، اگر بخواهید فقط، فایل خاصی را دوباره به موقعیت خود برگردانید باید بر روی Files Restore کلیک کنید و یا اگر بخواهید کل پارتیشن را برگردانید باید بر روی Restore Volumes کلیک کنید.



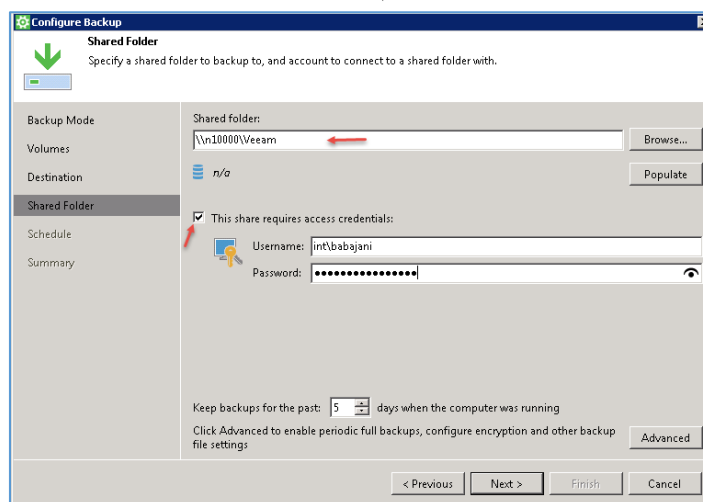
شکل ۱۳۵-۵ تنظیمات Veeam Agent

باتوجه به شکل ۱۳۶-۵، اگر دوباره به Configure backup بازگردید، دو گزینه‌ی دیگر وجود دارد که برای انجام پشتیبان‌گیری مورد استفاده قرار می‌گیرد، گزینه‌ی storage Local، یک محل ذخیره‌سازی را بر روی همان سیستم مشخص می‌کند که کار جالبی نخواهد بود و گزینه‌ی Shared folder برای ذخیره در یک پوشه‌ی اشتراک‌گذاری شده مورد استفاده قرار می‌گیرد.



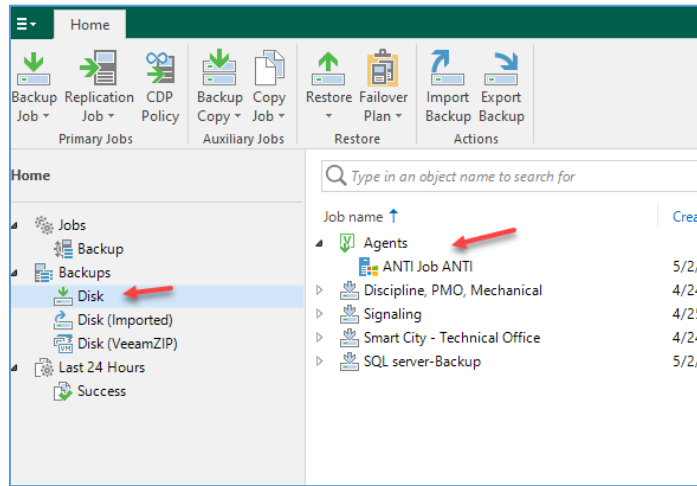
شکل ۵-۱۳۶ تنظیمات Veeam Agent

در شکل ۵-۱۳۷ باید آدرس فولدر به اشتراک گذاری شده را وارد کنید، مطمئن باشید که محل مورد نظر دارای فضای مناسب برای این کار است، در قسمت پایین می‌توانید نام کاربری را که به آن دسترسی دارد را وارد کنید.



شکل ۵-۱۳۷ تنظیمات Veeam Agent

برای به دست آوردن بهترین عملکرد در شبکه بهتر است از هاردهای SSD و شبکه با سرعت گیگابایت استفاده کنید. به مانند شکل ۵-۱۳۸، اگر وارد نرم افزار شوید از سمت چپ و از قسمت Home، قسمت Backups را باز کنید و بر روی Disk کلیک کنید، با این کار سروری که از طریق Agent پشتیبان گرفته شده است را به ما نمایش می‌دهد و می‌توانید آن را دوباره برگردانید و یا اطلاعات آن را مشاهده کنید.



شکل ۱۳۸-۵ تنظیمات Veeam Agent

فصل ششم

SQL Replication

۸-۱ بررسی سرویس Replication

عمل Replication یکی از ویژگی‌های سطح بالای SQL Server محسوب می‌شود. Replication متقابل هنگامی استفاده می‌شود که تغییرات شمای DML یا DDL اجرا شده روی یک شی موجود در دیتابیس یک سرور نیازمند منعکس شدن بر روی پایگاه داده‌ای موجود بر روی سرور دیگر نیز باشد. این تغییرات تقریباً در یک بازه‌ی زمانی Real مثلاً در ثانیه اتفاق می‌افتند. پس از بیان فنی، حال به زبانی ساده و در قالب مثال به توضیح Replication می‌پردازیم:

همان‌طور که می‌دانیم افراد بسیاری وجود دارند که مشترک مجلات الکترونیکی بوده و یا اخبار روز جهان را از طریق ایمیل خود دریافت می‌کنند؛ بنابراین با عضویت و مشترک شدن در این مجلات و mailerها به‌سادگی قادر به دریافت اخبار روز و مورد نظر خود هستند. به طور مشابه قابلیت Replication در MS SQL نیز همین نقش را ایفا کرده و داده‌ها را به‌عنوان مثال از یک سرور ریموت به Boxهای سرورهای لوکال از طریق مکانیزم نشر و اشتراک‌گذاری (publications and subscriptions) انتقال می‌دهد.

Replication در واقع به مجموعه‌ای از توپولوژی‌ها برای کپی و توزیع داده‌ها و اشیاء و یا Objectهای پایگاه‌داده، از یک پایگاه‌داده به دیگری و نیز به هماهنگ‌سازی بین پایگاه‌های داده برای حفظ انسجام اطلاق می‌شود. با استفاده از Replication می‌توانیم داده‌ها را به مکان‌های مختلف و میان کاربران از راه دور یا ریموت در سراسر شبکه‌های محلی و نیز گسترده و ارتباطات dial-up و wireless و همچنین در اینترنت، توزیع کنیم. دلایل و سناریوهای مختلفی موجود است که موجب می‌شود که Replication به‌عنوان ابزاری قدرتمند برای پخش کردن و انتشار داده‌ها در نظر گرفته شود. در اینجا به دلایلی برای عمل Replication اشاره خواهیم کرد:

در نظر گرفتن replication برای از بین بردن اثرات عملیات متمرکز و فشرده‌ی Read مثل تولید گزارش و... در واقع replication گزینه‌ی مناسبی است برای زمانی که اطلاعات مورد نظر read only بوده و نیازی به آپدیت کردن source نداریم.

در دسترس قراردادن داده‌ها برای کاربر، به‌عنوان مثال سرور A1 در تهران است و از آنجایی که سازماندهی بخش توزیع ویدئوهای آموزشی آن در شهر اهواز صورت می‌گیرد، برای چنین سازماندهی، نیازمند اطلاعات کاملاً مرتب هستیم. حال هر بار که ما نیاز به استفاده از داده‌های مربوط به محصولات و یا جداول فروش داشته باشیم از یک Linked Server برای برقراری ارتباط با سرور تهران و دریافت داده‌های مورد نظر استفاده خواهیم کرد. چنین شرایطی اثراتی را نیز به دنبال دارد:

قطعاً برای دریافت داده‌ها در هر زمان شدیداً به ارتباطات شبکه‌ای متکی خواهیم بود. سرور منبع یا همان Source Server نیز Load کاری بالایی را برای خواندن داده‌ها متحمل خواهد شد.

پیش از پرداختن به انواع replication و جزئیات مربوط به نصب آن ابتدا باید با مفهوم چند واژه آشنا شویم. عمل replication به طور سنتی از ساختار publisher/subscriber تبعیت می‌کند. همانند رابطه‌ی مشترکین و ناشران مجلات. برای هر مجله یک ناشر (publisher) وجود دارد که اطلاعات را در قالب مقالات (articles) منتشر می‌کند.

حال این مجله‌ای که دربردارنده‌ی مجموعه‌ای از مقالات است برای انتشار و در اختیار عموم و نیز مشترکان قرارگرفتن نیاز به یک توزیع‌کننده (distributer) دارد و این یک ساختار استاندارد برای چرخه‌ی publisher // subscriber محسوب می‌شود. اما گاهی با تغییراتی در این روند نیز مواجه می‌شویم به‌عنوان مثال ممکن است که یک publisher به‌عنوان یک distributer یا توزیع‌کننده ایفای نقش کند و یا اینکه یک distributer یا توزیع‌کننده ایفاگر نقش یک subscriber نیز باشد. حال به توضیح هر یک از این واژه‌های کلیدی می‌پردازیم:

Article یا مقاله: به اطلاعاتی اطلاق می‌شود که قصد replicate آنها را داریم. این اطلاعات می‌تواند یک جدول یا یک روال یا procedure یا یک جدول فیلتر شده و ... را در برگیرد. Publication یا نشریه: به گروهی از article‌ها اطلاق می‌شود. یک article به‌تنهایی قادر به انتشار نیست از این رو نیاز به ایجاد publication داریم. به بیان دیگر انتشار یا publication به معنی مجموعه‌ای از Article‌ها است (اشیاء مختلف پایگاه‌داده) که توسط ناشر منتشر شده است.

Publisher یا ناشر: به database روی Source سرور گفته می‌شود که در واقع قصد replicate داده‌های آن را داریم. به‌عبارت‌دیگر یک Publisher در واقع یک database Instance است که داده‌های موجود در طی فرایند Replication به مکان‌های دیگر را در دسترس قرار می‌دهد. یک ناشر می‌تواند یک یا چندین نشریه و هر تعریف منطقی مرتبط با مجموعه‌ی اشیاء و داده‌هایی برای Replicate را در برداشته باشد.

Distributor یا توزیع‌کننده: یک توزیع‌کننده را می‌توان همانند پسر بچه‌ای در نظر گرفت که مسئولیت تحویل نشریات به مشترکین را به عهده دارد. یک توزیع‌کننده می‌تواند ایفاگر نقش ناشر یا مشترک نیز باشد.

Distribution Database یا پایگاه داده‌ی توزیع شده: این پایگاه‌داده دربردارنده‌ی تمامی خط فرمان‌های replication است. زمانی که هرگونه از تغییرات شمای DML یا DDL در Publisher اجرا شود، دستورات مربوط به این اعمال که توسط SQL سرور تولید می‌شود، در این بخش ذخیره خواهد شد. این پایگاه‌داده می‌تواند روی همان سرور Publisher موجود باشد اما معمولاً برای عملکرد بهتر توصیه می‌شود که آن را بر روی یک سرور مجزا قرار دهند. به طور معمول مشاهده شده که اگر distribution database روی همان ماشینی باشد که پایگاه‌داده‌ی publisher روی آن قرار دارد،

در صورتی که تعداد زیادی Publisher موجود باشد، این موضوع عملکرد سیستم را تحت تأثیر قرار می دهد و این دلیلی است که برای هر Publisher یک فایل distrib.exe ایجاد می شود.

Subscriber یا مشترک: پرخه‌ی انتشار با دریافت داده‌ها توسط یک مشترک پایان می یابد. تغییرات منتشر شده میان تمامی مشترکین یک فرایند انتشار، از طریق یک توزیع کننده تکثیر می شود؛ بنابراین یک مشترک با اشتراک و عضویت در یک فرایند انتشار در نهایت قادر به دریافت داده‌ها خواهد بود. به عبارت دیگر Subscriber یک Database Instance است که داده‌های replicate شده را دریافت می کند و نیز می تواند داده‌ها را از چندین ناشر (publisher) یا نشریه (publishers) بگیرد. بسته به نوع Replication ای که انتخاب می شود، یک Subscriber می تواند داده‌ها را متقابلاً به ناشر انتقال داده و یا آنها را برای سایر مشترکین منتشر کند.

Subscription یا اشتراک: به درخواست یک مشترک یا subscriber برای دریافت نشریه یا publication اطلاق می شود و بر دو نوع است: Pull و Push که در واقع دو روش برای انتقال داده‌ها از distributor یا توزیع کننده به subscriberها یا مشترکین محسوب می شوند. در ذیل به شرح آنها می پردازیم:

Push Subscription: در روش push subscription یک distributor یا توزیع کننده مسئول داده‌های بر صف از یک publisher بوده و در نهایت آنها را در اختیار subscriberها قرار می دهد. این نوع اشتراک مدیریت را ساده و متمرکز می کند چون سناریوی replication معمولی، یک publisher و چندین subscriber را شامل می شود. مزیت این اشتراک امنیت بالای آن است چراکه فرایند آغازین در یک مکان مدیریت می شود. به عبارت دیگر کارایی distributorها کاهش می یابد زیرا کل توزیع subscriberها یک دفعه اجرا می شود.

Pull subscription: همانند روش push یک distributor یا توزیع کننده داده‌های بر صف یک publisher بوده و این وظیفه‌ی subscriberها است که با distributor ارتباط برقرار کرده و داده‌های صف بندی شده‌ی آماده برای عمل replication را به تصرف خود در آورند. در مقایسه با روش push از روش pull برای نشریه‌هایی با امنیت پایین و تعداد بالای مشترک ها استفاده می شود. این نوع اشتراک رایج تر بوده چرا که یک مشترک می تواند publication و یا نشریه‌هایی را انتخاب کند تا در آن شرکت کند.

شرح replication Agent ها در SQL سرور

حال به توضیح عواملی که برای انجام replication در پشت صحنه فعالیت می کنند می پردازیم که Agent نامیده می شوند. این agentها در فایل های مربوطه با پسوند exe در مسیر \COM folder قرار دارند. همچنین تمامی اطلاعات مربوط به agentها در جداول dbo.MSxxx_agents و dbo.MSxxx_history موجود در Distribution Database ثبت شده اند. حال به شرح انواع agentها و اینکه در کدام نوع از انواع replication کاربرد دارند می پردازیم برای روشن شدن این مبحث در اینجا اشاره می کنیم که replication در SQL سرور به سه نوع Snapshot و transactional

و merge تقسیم می‌شود که نیازمند Agentها برای فعالیت خود هستند. در ادامه عناوین ذکر شده شفاف‌سازی خواهند شد.

Snapshot Agent: یک فایل اجرایی است که snapshot فایل‌های دربردارنده‌ی شماتیک یا ساختار و داده‌های جداول و اشیاء پایگاه‌داده را ایجاد کرده و آنها را در Distributor ذخیره می‌کند همچنین اطلاعات مربوط به وضعیت synchronization را در distribution database ثبت می‌کند.

Distribution Agent: برای انواع snapshot و transactional مورد استفاده قرار می‌گیرد. و فایل‌های snapshot را از distribution db به مشترکین انتقال می‌دهد همچنین تمامی تراکنش‌های منتظر برای انتشار را نیز به subscriberها منتقل می‌کند و برای هر دو push subscription و pull subscription قابل اجرا است.

Log Reader Agent: برای transactional replication استفاده می‌شود تراکنش‌های مشخص شده برای replication را از transaction log به distribution db روی publisher انتقال می‌دهد. هر پایگاه داده‌ی log reader مختص به خود را دارد که روی distributor اجرا شده و با publisher ارتباط برقرار می‌کند.

Merge Agent: در merge replication کاربرد دارد و snapshot اولیه را برای subscriberها اجرا کرده و تغییرات تدریجی به وجود آمده روی داده‌ها را نیز ادغام کرده و به subscriberها انتقال می‌دهد. هر اشتراک ادغامی merge agent خود را دارد که قادر به برقراری ارتباط با publisher و subscriber و به‌روزرسانی هر دوی آنها می‌باشد.

Queue Reader Agent: برای transactional replication کاربرد دارد و بر روی Distributor اجرا شده و تغییرات صورت گرفته در سمت subscriber را به publisher بازمی‌گرداند. برخلاف merge agent و distribution agent تنها یک نمونه از Queue Reader Agent برای سرویس‌دهی به تمامی publisherها و publicationها و برای یک distribution db معین وجود دارد.

فیلتر کردن داده‌های منتشر شده:

فیلتر کردن جداول موجود در articleها ما را قادر می‌سازد تا پارتیشن‌هایی از داده‌ها را برای انتشار ایجاد کنیم. به کمک فیلتر کردن داده‌های منتشر شده می‌توان:

- داده‌های ارسال شده روی شبکه را به حداقل رساند.
- مقدار فضای ذخیره‌سازی مورد نیاز برای یک مشترک را کاهش داد.
- نشریات یا publicationها را سفارشی‌سازی کرده و نیز شرایطی را فراهم کرد که applicationها بر اساس نیازهای اختصاصی یک مشترک باشند.

اگر مشترکین داده‌ها را به‌روزرسانی کنند، می‌توان از بروز conflictها جلوگیری کرده و یا آنها را کاهش داد زیرا پارتیشن‌های داده‌های مختلف می‌توانند برای مشترکین مختلف ارسال شوند. (در واقع در این شرایط هیچ دو مشترکی قادر به به‌روزرسانی داده‌های مشابه نخواهند بود)

می‌توان از انتقال اطلاعات مهم و حساس جلوگیری کرد. فیلترهای ردیف و فیلترهای ستون‌ها می‌توانند برای محدود کردن دسترسی مشترک به داده‌ها مورد استفاده قرار گیرند.

Replication چهار نوع فیلتر را ارائه می‌دهد که در ذیل به معرفی آنها می‌پردازیم:

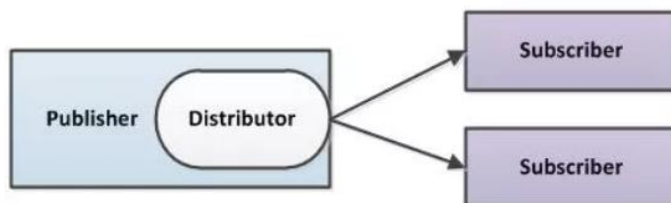
Static row filterها یا فیلتر کردن ردیف‌ها به‌صورت استاتیک که برای تمامی انواع replication از قبیل snapshot و transactional و merge کاربرد دارد. با استفاده از آن قادر خواهیم بود زیر مجموعه‌ای از ردیف‌هایی را که منتشر می‌شوند، انتخاب کنیم و یا در واقع برحسب نیازمان آنها را محدود کنیم.

Column filter یا فیلتر کردن ستون که این نوع فیلتر نیز برای تمامی انواع replication به کار می‌رود و به کمک آن می‌توان زیرمجموعه‌ای از ستون‌هایی را که منتشر می‌شوند، انتخاب کرد.

Parameterized row filtered یا فیلترهای پارامتری ردیف‌ها تنها برای merge replication به کار گرفته می‌شود. این نوع از فیلتر از نظر مفهوم همانند static filter است اما در اجرا تفاوت قابل توجهی با یکدیگر دارند. هدف parameterized filter ایجاد چندین پارتیشن از داده‌ها است که بدون ایجاد چندین نشریه یا publication بتوانند replicate داشته باشند. به‌عنوان مثال اگر ما از جدول پایه‌ی یکسانی استفاده کنیم و دو مشترک مختلف به نام‌های A و B نیز داشته باشیم که هر کدام به زیرمجموعه‌هایی متفاوتی از آن جدول نیاز داشته باشند، هنگام استفاده از فیلترهای ردیفی استاندارد نیازمند ایجاد دو نشریه یکی برای مشترک A و دیگری برای مشترک B خواهیم بود. اما با استفاده از فیلترهای پارامتری می‌توانیم برای مشترکین A و B به‌صورت مجزا مقادیر مورد نظر متفاوتی را در نظر بگیریم. اما در نهایت هر یک از این مقادیر و مجموعه داده‌ها به‌عنوان بخشی از یک نشریه محسوب می‌شوند.

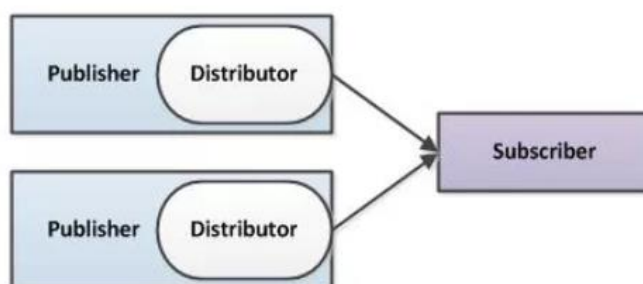
Join filter یا فیلتر الحاق که تنها برای merge replication کاربرد دارد و این نوع فیلتر به طور معمول همراه با parameterized filterها برای بسط و گسترش فیلترینگ به دیگر جداول مربوطه به کار می‌رود. مثلاً یک نمایندگی فروش معمولاً داده‌های جداول دیگر از قبیل مشتریان و دیگر جداول را دارد. این داده‌ها می‌توانند فیلتر شده باشند به‌طوری‌که نماینده فروش تنها داده‌های مشتریان خود و سفارش‌ها آنها را دریافت کند. این نوع فیلتر همچنین می‌تواند همراه با static filterها نیز به کار گرفته شود.

توپولوژی‌هایی که MS SQL سرور برای انجام Replication آنها را پشتیبانی می‌کند: Central Publisher: این یکی از پرکاربردترین نوع توپولوژی‌ها است که در replication مورد استفاده قرار می‌گیرد. در این سناریو یک سرور به‌عنوان publisher و distributor منظور شده و سرور و یا سرورهای دیگر به‌عنوان subscriberها در نظر گرفته می‌شوند.



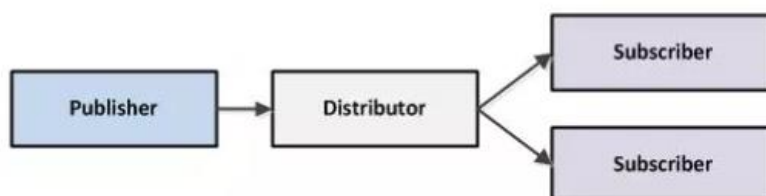
شکل ۶-۱ Central Publisher

Central subscriber: این یک توپولوژی رایج در انبار داده‌ها است. سرورها یا پایگاه داده‌های متعددی داده‌های خود را با یک سرور مرکزی replicate می‌کنند.



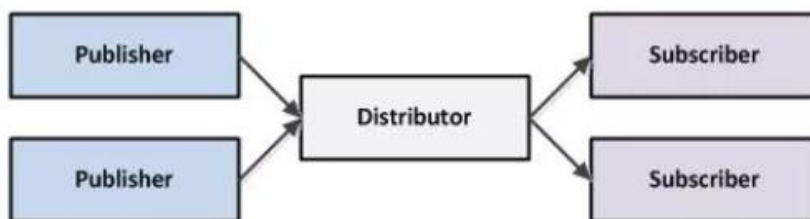
شکل ۶-۲ Central subscriber

Central publisher with remote distributor: در این توپولوژی Distribution database روی یک سرور مجزا از Distributor در نظر گرفته می‌شود از این ساختار زمانی استفاده می‌شود که سطح فعالیت‌های replication افزایش پیدا می‌کند و یا اینکه سرور و یا منابع شبکه محدود شده‌اند. این توپولوژی زمان load را برای publisher کاهش داده اما به‌طور کلی موجب افزایش ترافیک شبکه می‌شود.



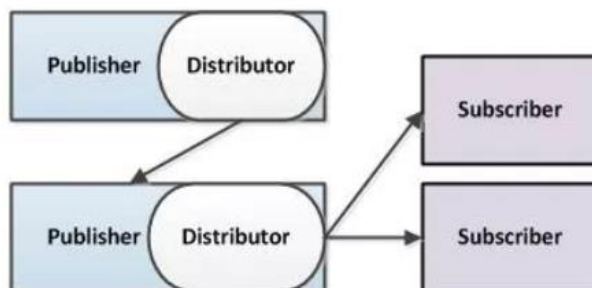
شکل ۶-۳ Central publisher with remote distributor

Central distributor: در این توپولوژی publisherهای متعدد تنها از یک distributor استفاده می‌کنند که روی سروری مجزا پیاده‌سازی شده است. این نوع توپولوژی عملاً چندان مورد استفاده قرار نمی‌گیرد. چراکه تنها دارای یک point of failure می‌باشد (یعنی همان سروری که به‌عنوان distributor مرکزی پیکربندی شده است) و اگر سرور distributor دچار اختلال شده و یا از کار بیفتد، کل عملیات replication ای که بر اساس این سناریو صورت می‌گیرد، نابود خواهد شد.



شکل ۴-۶ Central distributor

Publishing Subscriber: این توپولوژی دارای نقشی دوگانه است. در این ساختار دو سرور داده‌های یکسان را منتشر می‌کنند. یکی از سرورهایی که عهده‌دار نقش publisher است داده‌ها را برای subscriber ارسال کرده و سپس subscriber داده‌ها را برای هر تعداد از مشترکین موجود منتشر می‌کند. این توپولوژی هنگامی کاربرد دارد که یک publisher قصد انتقال داده‌ها را از طریق لینک‌های ارتباطی کند و یا گران‌قیمت به subscriber داشته باشد.



شکل ۴-۵ Publishing Subscriber

انواع Replication

انواع replication ای که در SQL Server 2019 صورت می‌گیرند به‌قرار زیر است:

Transactional Replication یا رونوشت تراکنش

Merge Replication یا رونوشت ادغامی

Snapshot replication یا رونوشت ثبت لحظه‌ای

نوع replication ای که ما انتخاب می‌کنیم به فاکتورهای مختلفی وابسته است این فاکتورها محیط فیزیکی replication نوع و کمیت داده‌هایی که قصد replicate آنها را داریم و اینکه آیا داده‌ها باید برای subscriber به‌روزرسانی شوند یا خیر را شامل می‌شود. محیط فیزیکی به تعداد و مکان کامپیوترهای درگیر در عمل replication اطلاق می‌شود. حال این کامپیوترها می‌توانند clientهایی همچون workstationها، لپ‌تاپ‌ها و یا device های handle بوده و یا سرورها را در برگیرند.

هر نوع از replication به طور معمول با همگام‌سازی اولیه‌ی objectهای publish شده میان publisher و subscriber شروع می‌شود. این همگام‌سازی اولیه می‌تواند توسط replication و با یک snapshot ایجاد شود. Snapshot یک کپی از تمامی objectها و داده‌های مشخص شده توسط یک نشریه یا publication تهیه می‌کند. پس از اینکه snapshot ایجاد شد، به مشترکین یا subscriberها تحویل داده می‌شود.

برای برخی از نرم‌افزارهای کاربردی snapshot replication موردنیاز می‌باشد و برای دیگر applicationها این مسئله مهم است که تغییرات ایجاد شده روی داده‌ها به صورت تدریجی در طول زمان در اختیار Subscriber قرار بگیرد. برخی از Applicationها نیازمند بازگشت تغییرات صورت گرفته روی داده از subscriber به publisher نیز هستند. transactional replication و merge replication آپشن‌هایی هستند که این امکان را برای این گونه applicationها فراهم می‌آورند. تغییرات صورت گرفته روی داده‌ها، از طریق snapshot قابل پیگیری نیست و هر زمان که یک snapshot گرفته شده تأیید شود، روی داده‌ی موجود overwrite خواهد شد. در transnational replication پیگیری تغییرات از طریق transaction logهای SQL Server و در merge replication نیز از طریق triggerها و metadata table میسر خواهد بود. حال به شرح هر کدام از انواع replication می‌پردازیم:

Snapshot Replication یا رونوشت ثبت لحظه‌ای

داده‌ها را دقیقاً همانگونه که در یک لحظه‌ی خاص زمانی ظاهر می‌شوند توزیع می‌کند و نمی‌توان بواسطه‌ی آن آپدیت و به‌روزرسانی داده‌ها را مانیتور کرد. وقتی که عمل synchronization اتفاق می‌افتد Snapshot تولید شده و برای subscriberها ارسال می‌شود. باید این نکته را در نظر داشت که Snapshot Replication به‌خودی‌خود مورد استفاده قرار می‌گیرد اما فرایند پردازش snapshotها که در واقع مراحل کپی‌برداری از objectها و داده‌های تعیین شده توسط یک نشریه را نیز شامل می‌شود و معمولاً برای فراهم‌آوردن مجموعه‌ی اولیه‌ی از داده‌ها و objectهای پایگاه‌داده برای انواع transactional /merge replication کاربرد دارد. استفاده از snapshot replication به‌خودی‌خود هنگامی مناسب است که موارد عنوان شده در ذیل محقق باشند:

- داده‌ها به‌ندرت تغییر داشته باشند
- قصد replicate کردن حجم کمی از داده‌ها را داشته باشیم.
- اگر که حجم زیادی از داده‌ها در طی یک دوره کوتاه زمانی تغییر کند.

مناسب‌ترین زمان برای استفاده از snapshot replication هنگامی است که تغییراتی اساسی و قابل‌توجه اما نادر برای داده‌ها اتفاق بیفتد. برای مثال اگر لیست قیمت محصولات موجود در فروشگاه ITPro ثابت بوده و یک یا دو بار در سال در یک‌زمان مشخص آپدیت و به‌روزرسانی می‌شوند، در چنین وضعیتی استفاده از snapshot replication بعد از

اعمال تغییرات روی داده‌ها توصیه می‌شود با توجه به انواع خاصی از داده‌ها، ممکن است که Snapshot های مکرری نیاز باشد.

مثلاً اگر یک جدول نسبتاً کوچک در سرور publisher در طی روز آپدیت شده اما اندکی تاخیر نیز جایز باشد، این تغییرات می‌توانند به عنوان یک Snapshot شبانه تحویل داده شوند. Snapshot replication دارای سربار مداوم کمتری نسبت به transactional replication بر روی publisher می‌باشد چراکه تغییرات تدریجی را دنبال نمی‌کند. با این حال اگر مجموعه داده‌ای که در حال replicate است بسیار بزرگ باشد به منابع قابل توجهی برای ساخت و به کار بردن snapshot نیاز دارد. هنگام ارزیابی شرایط برای استفاده از snapshot replication باید اندازه‌ی کل مجموعه داده‌ها و فراوانی تغییرات ایجاد شده روی آنها را در نظر گرفت.

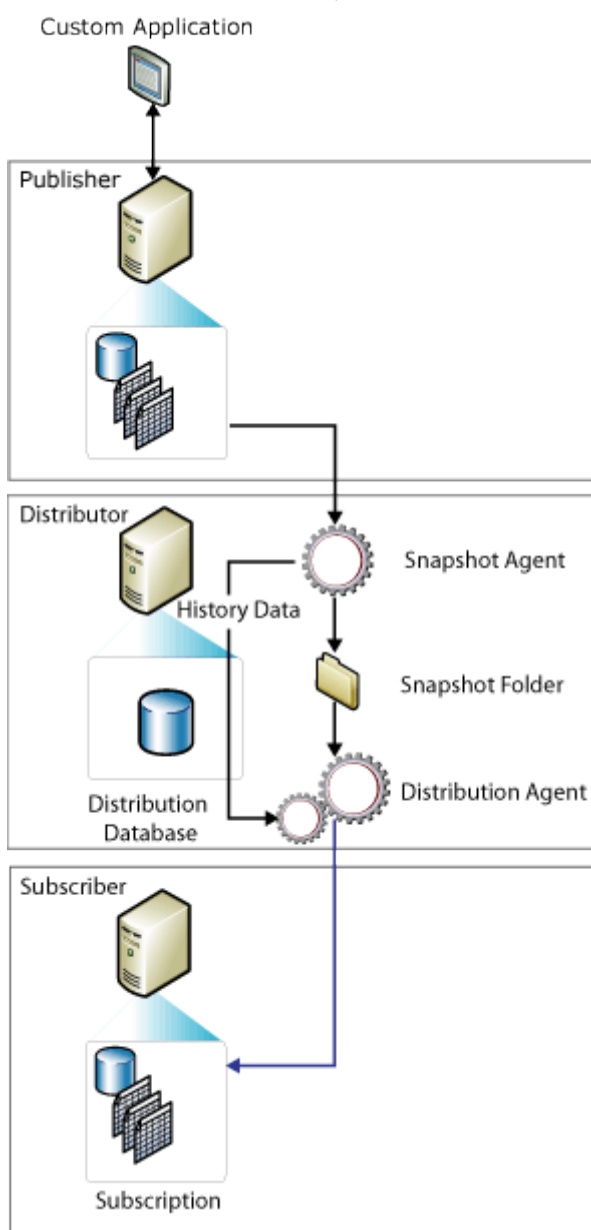
Snapshot replication چگونه کار می‌کند؟

به طور پیش فرض هر سه نوع replication از یک snapshot برای مقاردهی اولیه به subscriberها استفاده می‌کنند. همیشه Snapshot agent موجود در SQL Server وظیفه تولید فایل‌های snapshot را به عهده دارد اما agent مربوط به ارائه و تحویل این فایل‌ها بسته به نوع replication انتخاب شده، متفاوت است. Snapshot replication و transactional replication برای ارائه فایل‌ها از distribution agent استفاده می‌کنند در حالی که merge replication برای این منظور از merge agent بهره می‌گیرد. Snapshot agent روی distributor اجرا می‌شود. distribution agent و Merge Agent روی یک Distributor برای push subscription اجرا شده و همچنین بر روی subscriptionها برای pull subscriberها اجرا می‌شوند.

Snapshot می‌تواند بلافاصله بعد از ایجاد یک subscription، تولید و اعمال شود و یا اینکه بر اساس یک برنامه در زمان انتشار ساخته شود. Snapshot Agent فایل‌های snapshot حاوی ساختار و داده‌های جداول منتشر شده (published tables) و نیز اشیاء پایگاه داده را آماده کرده و این فایل‌ها را برای ناشر یا publisher در Snapshot folder ذخیره می‌کند و مسیر ردیابی این اطلاعات را در Distribution database موجود در Distributor ثبت می‌کند. می‌توان Snapshot folder پیش فرض را به هنگام پیاده‌سازی و پیکربندی یک Distributor مشخص کنیم اما می‌شود که محل دیگری نیز برای نشریه، به جای آن فولدر یا علاوه بر آن فولدر پیش فرض در نظر گرفت.

علاوه بر فرایند snapshot استاندارد که به توضیح آن پرداختیم، یک فرایند دوبخشی snapshot نیز وجود دارد که در انتشار به صورت ادغام کاربرد دارد که از parameterized filter بهره می‌برد. همان گونه که قبلاً گفتیم یک filter فرایندی است که اطلاعات را محدود و یک زیرمجموعه را تولید می‌کند. استفاده از parameterized filter به ما این اجازه را می‌دهد که پارتیشن‌های مختلفی از داده‌ها را برای subscriberهای متعدد ارسال کنیم بدون اینکه نیاز به ایجاد

publication ها و یا همان نشریه های متعدد باشد. در تصویر ۶-۶ اجزای اصلی Snapshot replication نمایش داده شده که به درک بهتر مفاهیم گفته شده کمک خواهد کرد:



شکل ۶-۶ Snapshot replication

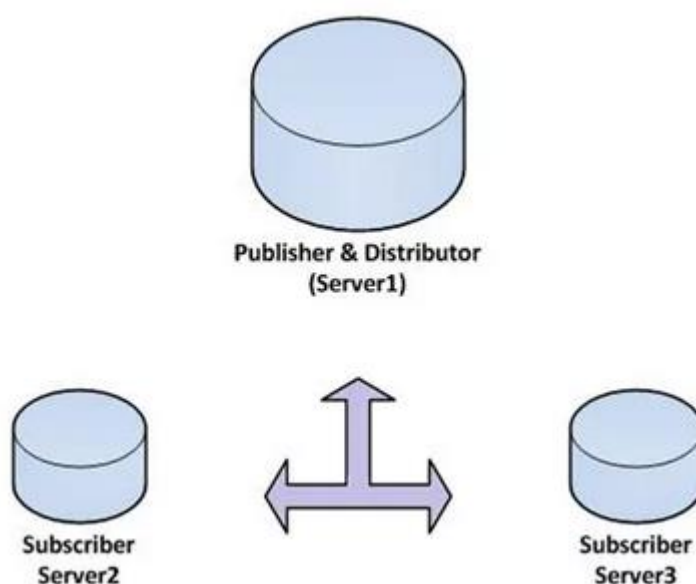
Transactional Replication یا رونوشت تراکنشی

عملیات transactional replication به طور معمول با گرفتن یک snapshot از اشیا و داده‌های پایگاه‌داده‌ی یک publication شروع می‌شود. به محض اینکه Snapshot اولیه گرفته شد، تغییرات بعدی که روی داده‌ها و شماهای موجود در سمت ناشر یا publisher ایجاد شده به subscriberها تحویل داده می‌شوند. به طور جزئی‌تر می‌توان گفت که تغییرات و تراکنش‌های که روی articleهای منتشر شده رخ داده‌اند از سمت publisher برای distributor ارسال می‌شود تا آنها

را برای subscriberها و یا همان مشترکین بفرستد. subscriberها تنها می‌توانند از این داده‌ها به صورت read only استفاده کنند چراکه در این نوع replication، تغییراتی که صورت می‌گیرد مجدداً برای publisher بازگردانده نمی‌شود. باین حال transactional replication آپشن‌هایی ارائه می‌دهد که شرایط به‌روزرسانی اطلاعات را برای Subscriberها فراهم می‌آورند.

قطعاً عنوان کردن یک مثال مسئله را روشن‌تر می‌کند: یک وب‌سایت رزرو بلیط را در نظر می‌گیریم تمامی بلیط‌های رزرو شده به صورت متمرکز در پایگاه‌داده‌ی واقع در شهر تهران ذخیره می‌شوند و در هر شهر از کشور نیز یک مرکز توزیع یا distribution center قرار دارد که رزروها را گرفته و بلیط‌های رزرو شده را برای آدرس‌های موجود ارسال می‌کند. لازم است که تمامی بلیط‌های رزرو شده از اهواز برای مشتریان مربوطه ارسال شوند. پایگاه توزیع مرکزی موجود در اهواز می‌تواند که یک transnational replication فیلتر شده را راه‌اندازی کند برای اینکه هر رزرو جدیدی (تراکنشی) که صورت گرفت با سایر شعب مربوط به این مرکز، در حداقل زمان ممکن replicate شود. (فیلتر موجب می‌شود که این مرکز تنها رزروها را برای اهواز دریافت کند). این شعب نیازمند دسترسی read only برای استفاده از داده‌های replicate شده هستند که transnational replication این امکان را برای آنها فراهم می‌آورد؛ بنابراین مواردی را می‌شود برای transactional replication به‌خاطر سپرد:

از آنجایی که عمل replication در طی یک تراکنش اتفاق می‌افتد، تأخیر تکرار و replicate داده‌ها بسیار ناچیز است. مشترکین دسترسی read only به داده‌ها دارند، از این رو تقریباً هیچ‌گونه استقلالی برای مشترکین یا subscribe وجود ندارد.



شکل ۶-۷ Transactional Replication

Transactional replication به طور معمول برای محیط‌های سرور به سرور (server-to-server) استفاده می‌شود و برای هر یک از موارد زیر مناسب است:

زمانی که بخواهیم تغییرات تدریجی داده‌های مورد نظر را پس از وقوع برای مشترکین منتشر کنیم. برای application‌هایی که به تأخیر کمی بین زمانی که یک تغییر در سمت ناشر ایجاد می‌شود تا زمانی که تغییر به یک مشترک می‌رسد نیاز دارند.

هنگامی که publisher دارای حجم بسیار بالایی از فعالیت‌ها از قبیل درج، به‌روزرسانی و یا حذف باشد. برای زمانی که Publisher یا Subscriber پایگاه‌داده ای غیر از SQL سرور باشد. مثلاً Oracle باشد. یک Application نیاز به دسترسی برای مداخله در وضعیت داده‌ها داشته باشد برای مثال اگر ردیفی از یک جدول پنج بار تغییر کند، Transactional Replication به هر Application این امکان را می‌دهد که قادر باشد برای هر تغییر مداخله کند (مثلاً استفاده از triggerها) و تغییرات داده‌ها به‌سادگی به‌ردیف‌ها اعمال نشوند.

Transactional Replication چگونه کار می‌کند؟

برای پیاده‌سازی transactional replication عواملی مانند Snapshot Agent و Log Reader Agent و Distribution Agent ها دخیل هستند. Snapshot Agent در واقع snapshot فایل‌های دربردارنده ساختارها (schema) و داده‌های جداول منتشر شده و اشیاء پایگاه‌داده را مهیا کرده و این فایل‌ها را در فولدر Snapshot ذخیره می‌کند و نتیجه‌ی عمل synchronization را در Distribution database موجود روی Distributor ثبت می‌کند.

Log Reader Agent وظیفه‌ی مانیتور کردن transaction log های هر پایگاه داده‌ای که برای عمل transactional replication بیکربندی شده را به عهده داشته و تراکنش‌های مشخص شده برای عمل replication را از transaction log به distribution database کپی می‌کند. Distribution Agent نیز Snapshot فایل‌های اولیه از پوشه‌ی snapshot و همچنین تراکنش‌های نگه‌داشته شده در جداول distribution database را برای مشترکین یا subscriber کپی می‌کند.

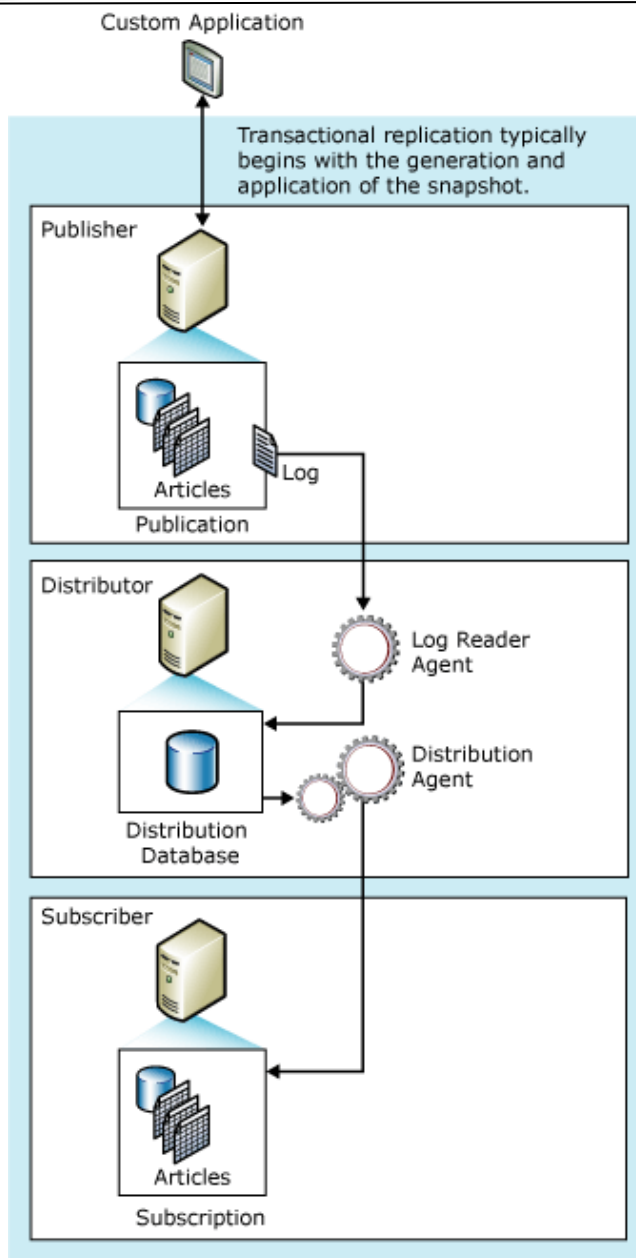
تغییرات تدریجی ایجاد شده سمت ناشر مطابق برنامه زمان‌بندی Distribution Agent در دسترس مشترکین قرار می‌گیرند که این جریان انتقال اطلاعات می‌تواند به طور مداوم با حداقل زمان تأخیر و یا در فواصل زمانی برنامه‌ریزی شده اجرا شود. از آنجاکه تغییرات داده‌ها باید در سمت ناشر ایجاد می‌شوند. (زمانی که از transactional replication بدون آپشن‌های immediate updating و queued updating بهره می‌گیریم. البته در ادامه به شرح کارایی این دو آپشن می‌پردازیم) باید از ایجاد conflict در به‌روزرسانی‌ها جلوگیری کرد. در نهایت تمامی مشترکین به مقادیر مشابه و یکسان از یک ناشر دست خواهند یافت. همچنین اگر از آپشن‌های immediate updating یا queued updating برای transactional replication بهره بگیریم، به‌روزرسانی‌ها می‌توانند سمت مشترکین ایجاد شوند و با queued updating، ممکن است تضاد و conflict اتفاق بیفتد.

Immediate Updating به مشترکین Snapshot // Transactional Replication اجازه می‌دهد تا داده‌های replicate شده را در سمت مشترک به‌روزرسانی کرده و تغییرات به وقوع پیوسته را به ناشر و دیگر مشترکین بازگردانی کند. این آپشن برای زمانی مفید است که قصد استفاده از Snapshot یا transactional replication را داشته باشیم اما نیازمند شرایطی باشیم که به‌روزرسانی‌های گاه‌به‌گاه در سمت مشترک صورت گیرد. برای استفاده از این آپشن، ناشر و مشترک باید در دسترس بوده و ارتباط بین آن دو برقرار باشد.

Queued Updating به مشترکین Snapshot // Transactional Replication نیز اجازه می‌دهد داده‌های منتشر شده را بدون نیاز به یک اتصال شبکه‌ای اکتیو به publisher، تغییر دهند. وقتی که یک نشریه یا publication را با استفاده از این آپشن ایجاد می‌کنیم و یکی از مشترکین عملیاتی چون Insert یا Update یا Delete را روی داده‌های منتشر شده پیاده‌سازی می‌کند، این تغییرات در یک صف ذخیره می‌شوند. هنگامی که ارتباطات شبکه‌ای مجدداً برقرار شود، این تراکنش‌های بر صف به صورت غیر هم‌زمان به publisher اعمال می‌شوند.

از آنجایی که به‌روزرسانی داده‌ها به طور غیر هم‌زمان برای publisher منتشر می‌شوند، همان داده‌ها ممکن است توسط خود ناشر یا publisher و یا مشترک دیگری نیز آپدیت شده باشند در نتیجه به هنگام اعمال آپدیت‌های برصف ممکن است conflict یا ناسازگاری به وجود بیاید. این conflict‌ها به کمک conflict resolution policy که به هنگام ایجاد یک publication تنظیم می‌شود، شناسایی و حل و فصل می‌شوند. Queued updating برای Application‌هایی مناسب است که کاربران آنها اغلب داده‌ها را خوانده و گاه‌به‌گاه به‌روزرسانی داده‌ها می‌پردازند. ارتباطات مشترکین نیز باید اکثر اوقات برقرار باشد و اگر که مشترکین آفلاین باشند نیز عملیات به‌روزرسانی بدون وقفه ادامه خواهد یافت.

نکته: این آپشن‌ها قابلیت سوئیچ شدن روی یکدیگر را دارند به‌عنوان مثال برای Subscription مشخص کرده‌ایم که از حالت immediate updating استفاده کند، اما فرضاً تحت شرایطی ارتباطات شبکه دچار مشکل شده بنابراین می‌توانیم برای دریافت آپدیت‌ها روی queued updating سوئیچ کنیم. باید در نظر داشت که replication به صورت اتوماتیک نمی‌تواند بین حالات آپدیت سوئیچ کند و برای این منظور باید تنظیمات مربوط به update mode از طریق SQL Server Management Studio انجام شود و یا application ما sp_setrepfailovermode را برای سوئیچ بین دو حالت فراخوانی کند. (این روال به ما اجازه می‌دهد تا تنظیمات مربوط به failover را برای مشترکین یا subscriberها انجام دهیم تا قادر به تعویض وضعیت از حالت immediate updating به Queued updating باشند.) نحوه عملکرد transactional replication در تصویر ۸-۶ نمایش داده شده است:



شکل ۸-۶ transactional replication

Merge Replication یا رونوشت ادغامی:

در snapshot/transactional replication ناشران داده‌ها را ارسال کرده و مشترکین آنها را دریافت می‌کنند و احتمال اینکه یک مشترک داده‌هایش کپی شده را به ناشران ارسال کند، وجود ندارد اما این امکان به کمک merge replication میسر شده است. Merge replication نیز همانند transactional replication فعالیت خود را به طور معمول با گرفتن snapshot از اشیا و داده‌های موجود در پایگاه داده‌ی یک publication آغاز می‌کند. تغییرات ثانویه داده‌ها و نیز تغییرات طرح یا schema در سمت ناشر صورت می‌گیرد و مشترکین نیز به واسطه‌ی triggerها این تغییرات را دنبال می‌کنند.

یک مشترک یا subscriber به هنگام اتصال به شبکه داده‌های خود را با ناشر تطبیق داده و synchronize می‌کند و در واقع به مبادله‌ی تمامی ردیف‌هایی که از آخرین عمل synchronization بین ناشر و مشترک دچار تغییر شده‌اند، می‌پردازد. Merge replication معمولاً برای محیط‌های سرور به سرویس‌گیرنده یا Server-to-Client کاربرد دارد و به‌کارگیری آن در هریک از شرایط عنوان شده در زیر مناسب است:

در شرایطی که چندین مشترک ممکن است که داده‌های مشابه را در زمان‌های مختلفی به‌روزرسانی کرده و این تغییرات را برای ناشر و سایر مشترکین منتشر کنند.

زمانی که مشترکین نیاز به دریافت داده‌ها و ایجاد تغییرات به‌صورت آفلاین داشته باشند و بعداً این تغییرات را با ناشر و دیگر مشترکین synchronize کنند.

هر مشترک به یک پارتیشن متفاوت از داده‌ها نیاز داشته باشد.

هنگامی که ممکن است conflict اتفاق بیفتد و در چنین شرایطی باید قادر به تشخیص و رفع این مشکل باشیم.

Merge replication به سایت‌های مختلف اجازه می‌دهد تا به‌صورت مستقل کار کنند و در نهایت آپدیت‌ها را با یکدیگر ادغام کرده تا به نتیجه‌ای یکسان و یکنواخت دست یابند. از آنجایی که به‌روزرسانی در بیشتر از یک گره صورت می‌گیرد، داده‌های مشابه ممکن است توسط ناشر و نیز بیش از یک مشترک آپدیت شوند بنابراین ممکن است به هنگام ادغام این آپدیت‌ها تضاد یا conflict اتفاق بیفتد و merge replication راه‌هایی را برای رسیدگی به مشکل conflict فراهم می‌آورد.

Merge Replication چگونه کار می‌کند؟

Merge replication توسط عامل‌هایی چون snapshot agent و merge agent اجرا می‌شود. اگر publication یا نشریه فیلتر نشده باشد و یا از فیلترهای استاتیک استفاده می‌کند، snapshot agent تنها یک snapshot ایجاد می‌کند و اگر که publication از فیلترهای پارامتری یا parameterized filter بهره می‌گیرد، snapshot agent از هر پارتیشن داده‌ها یک snapshot تهیه می‌کند. merge agent نیز snapshot‌های اولیه‌ی تهیه شده را به مشترکین اعمال می‌کند. این عامل همچنین تغییرات تدریجی که روی داده‌ها در سمت ناشر و یا مشترکین پس از ایجاد snapshot اولیه صورت گرفته را ادغام کرده و با توجه به قوانینی که برای آن پیکربندی می‌کنیم به تشخیص و حل و فصل conflict‌های ایجاد شده می‌پردازد. به هنگام استفاده از Merge replication سه تغییر برای ساختار پایگاه داده‌ی نشریه اتفاق می‌افتد:

یک ستون منحصر به فرد را برای هر سطر کپی شده تعیین می‌کند

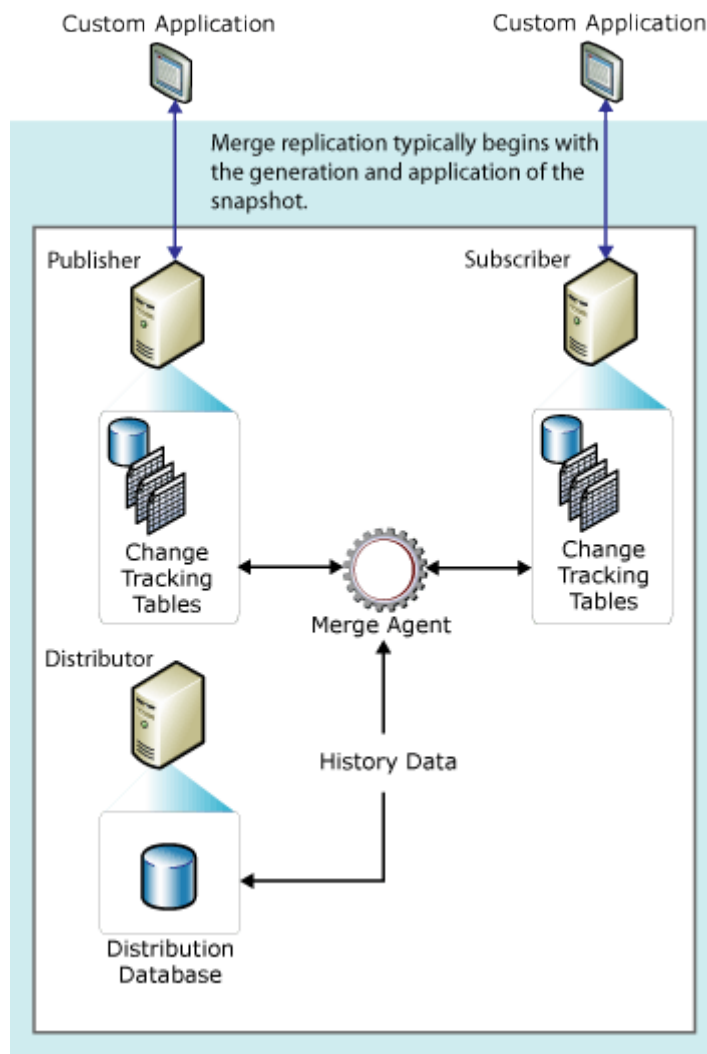
چندین جدول سیستمی اضافه می‌شود

Triggerهایی برای جداولی که داده‌های آنها کپی شده‌اند، ایجاد می‌کند.

برای ردیابی تغییرات، merge replication و نیز transactional replication ای که از قابلیت queued updating استفاده می‌کند باید قادر به شناسایی هر ردیف از جدول منتشر شده به گونه‌ای منحصر به فرد باشند. برای انجام این عمل merge replication ستون rowguid را به هر جدول اضافه می‌کند مگر اینکه جدول در حال حاضر دارای ستونی با نوع داده‌ی uniqueidentifier و ویژگی ROWGUIDCOL باشد که در این صورت از همین ستون برای شناسایی هر سطر و تغییرات آنها استفاده می‌شود.

SQL سرور چندین جدول سیستمی برای پشتیبانی از ردیابی داده‌ها، انجام عمل synchronization به صورت کارآمد و برای تشخیص Conflictها و نیز برای گزارش‌دهی، را به پایگاه داده اضافه می‌کند. تمامی تغییرات مربوط به داده‌ها در جداول سیستمی msmerge__contents و msmerge__tombstone ذخیره می‌شوند. این نوع از replicatinها triggerهایی را نصب می‌کند که تغییرات داده‌های هر ستون و یا هر ردیف از جدول را دنبال می‌کنند.

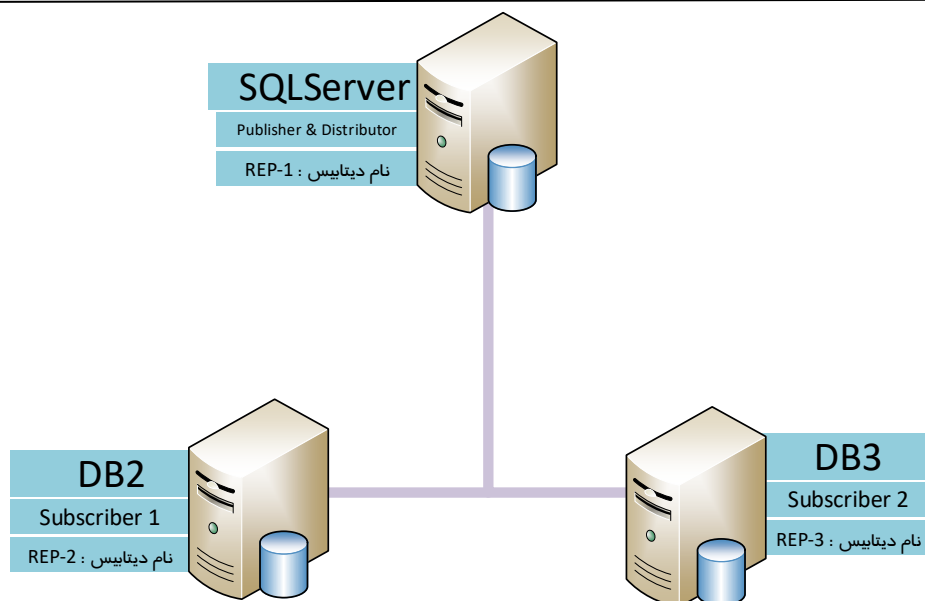
Triggerها تغییرات ایجاد شده روی جدول منتشر شده را ضبط کرده و این تغییرات را در جدول‌های سیستمی msmerge__contents و msmerge__tombstone ثبت می‌کنند. این triggerها هنگامی که snapshot agent برای اولین بار برای یک نشریه یا publication اجرا می‌شود، ایجاد می‌شوند. Triggerها در سمت یک مشترک نیز هنگامی به وجود می‌آیند که snapshot برای مشترک استفاده شود. در تصویر زیر اجزایی که در merge replication به کار گرفته می‌شوند نمایش داده شده است:



شکل ۶-۹ merge replication

۸-۲ نصب و راه‌اندازی سرویس Replication

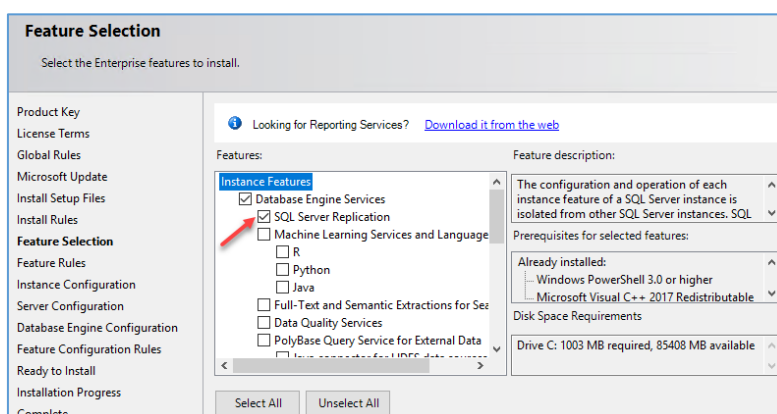
برای راه‌اندازی این سرویس نیاز به سه سرور داریم تا بتوانیم این سرویس را پیاده‌سازی کنیم، در شکل 44 سه سرور را برای این کار مشاهده می‌کنید، سرور SQLServer که سرور اصلی می‌باشد و نقش Publisher & Distributor را بازی می‌کند، در سرور DB-2 و DB-3 نقش Subscriber را دارند، برای اینکه سرورها را ایجاد کنید پیشنهاد من سرور مجازی است تا هم نحوه‌ی ایجاد و کار با آن آسان باشد و هم نحوه‌ی پشتیبان‌گیری آن سریع و بدون مشکل انجام شود.



شکل ۴۴ سناریو Replication

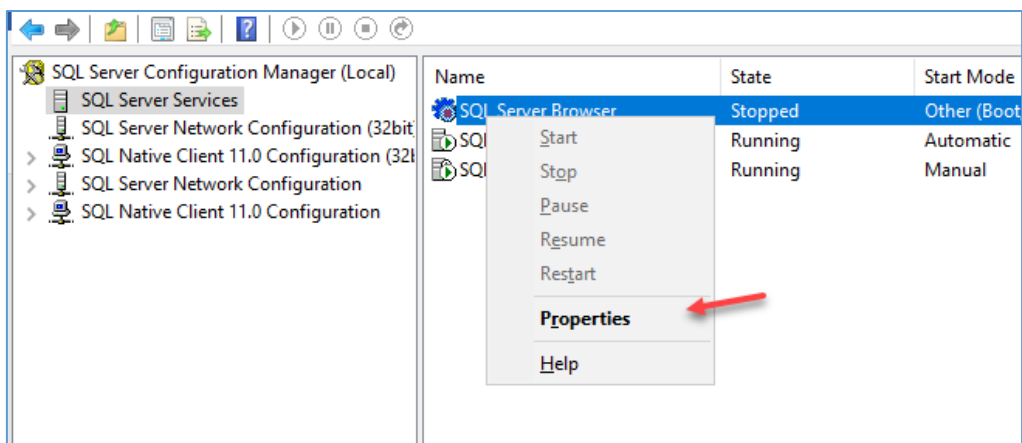
در ادامه و در سناریو شکل ۴۴ می‌خواهیم که سرور SQLServer هرگونه تغییری که در جدول مرتبط با لیست کاربران برتر ایجاد می‌شود را با جداول موجود روی سرورهای DB2 و DB3 Replicate کند؛ بنابراین باید تنظیمات لازم برای توزیع و انتشار (Publication و Distribution) را روی سرور SQLServer و تنظیمات مربوط به اشتراک (Subscription) را روی سرورهای DB2 و DB3 انجام دهیم.

اول از هر چیز باید در نظر داشته باشیم که موقع نصب SQL سرور، قابلیت Replication را نیز فعال کنیم، برای همین به‌مانند شکل ۴۵ باید تیک گزینه‌ی SQL Server Replication را انتخاب و آن را نصب کنید، در هر سه سرور باید این کار را انجام دهید.



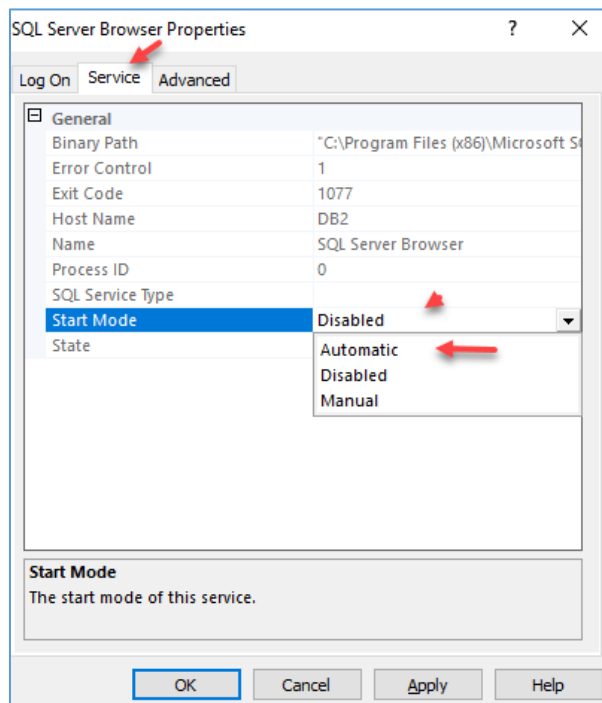
شکل ۴۵

بعد از فعال‌کردن سرویس SQL Server Replication باید به‌مانند شکل ۴۷ وارد SQL Server Configuration Manager شوید و سرویس SQL Server Browser را فعال کنید، این کار را باید در همه‌ی سرورها انجام دهید، البته برای اینکه این سرویس را فعال کنید باید بر روی آن کلیک راست کنید و گزینه‌ی Properties را انتخاب کنید.



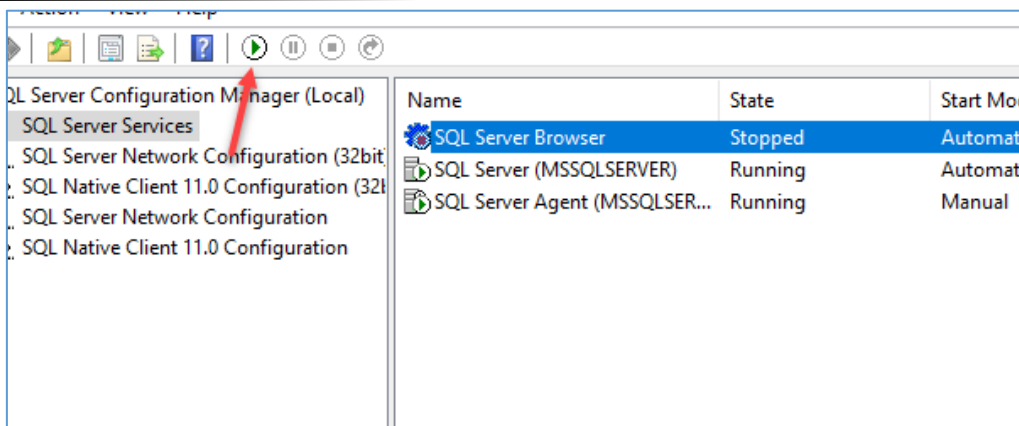
شکل ۴۷

در شکل ۴۸ باید وارد تب Service شوید و گزینه‌ی Start Mode را در حالت Automatic قرار دهید و ok کنید.



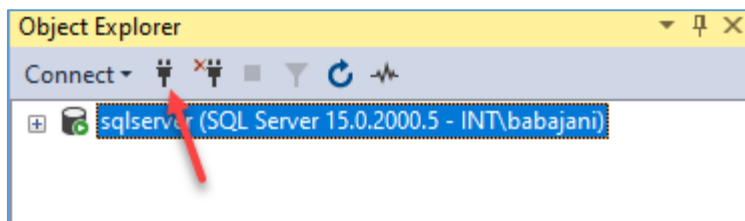
شکل ۴۸

در ادامه به‌مانند شکل ۴۹ باید سرویس را Start کنید.



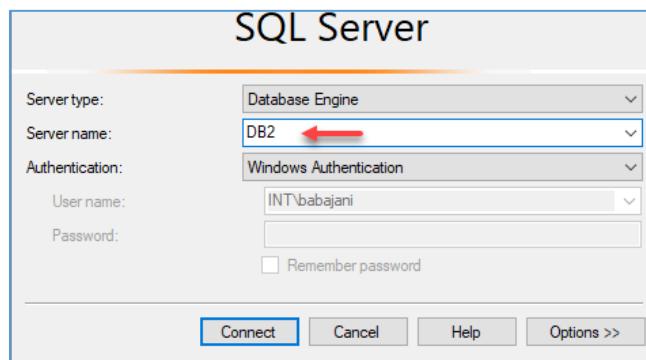
شکل ۴۹

برای اینکه هم‌زمان به هر سه سرور دسترسی داشته باشید باید SQL Server Management Studio را اجرا کنید و به‌مانند شکل ۵۰ دو سرور DB2 و DB3 را با کلیک بر روی Connect اضافه کنید.



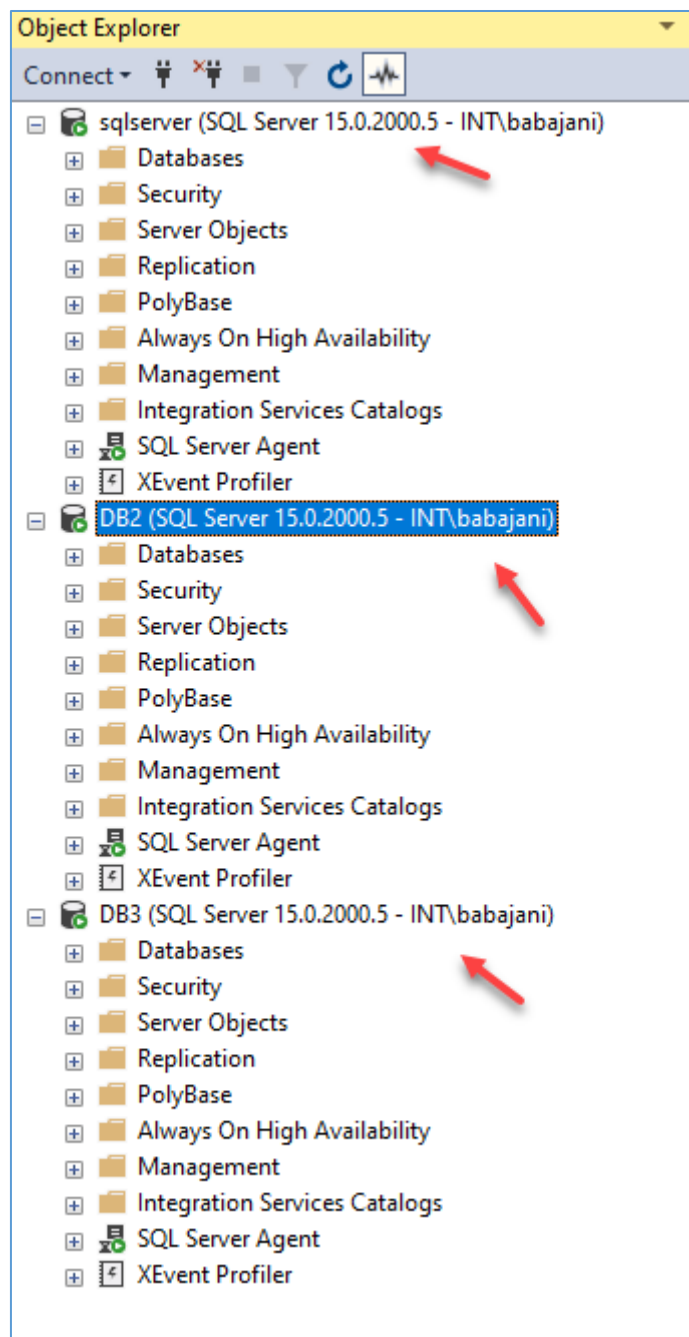
شکل ۵۰

در شکل ۵۱ باید سرور DB2 را وارد کنید و بر روی Connect کلیک کنید، توجه داشته باشید دقیقاً همین کار را برای سرور DB3 انجام دهید.



شکل ۵۱

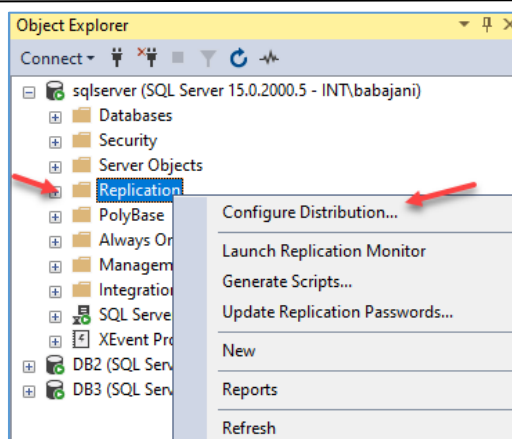
همان‌طور که در شکل ۵۲ مشاهده می‌کنید هر سه سرور به لیست اضافه شده است و حالا می‌توانیم عملیات Replication را آغاز کنیم.



شکل ۵۲

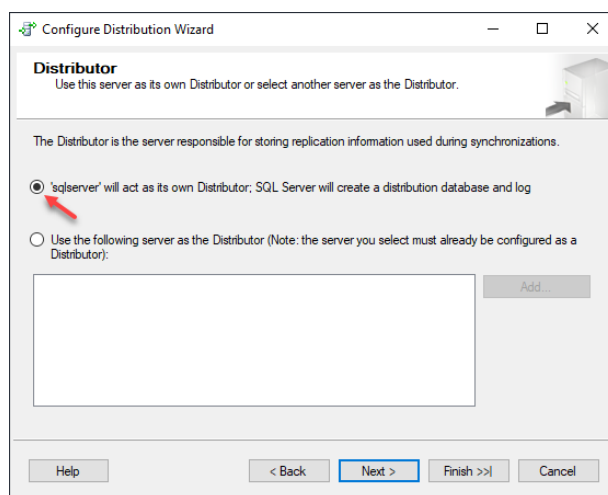
پیکربندی بخش توزیع یا Distribution

در ابتدای کار به سراغ Distribution بر روی سرور SQLServer می‌رویم. برای این منظور روی گزینه‌ی Replication در کنسول مدیریتی راست کلیک کرده و عبارت Configure Distribution را انتخاب کنید.



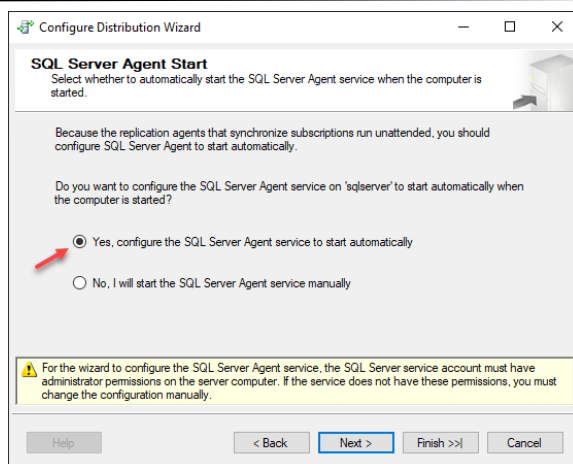
شکل ۵۳

به‌مانند شکل ۵۴ سرور توزیع‌کننده یا Distributor را انتخاب می‌کنیم که می‌توان خود سرور فعلی را انتخاب کرده و یا سرور دیگری را به‌عنوان توزیع‌کننده در نظر بگیریم. فقط باید در نظر داشته باشیم که پیکربندی Distribution روی سرور دیگر که آن را به‌عنوان توزیع‌کننده انتخاب می‌کنیم، از قبل انجام شده باشد. در اینجا ما سرور فعلی را انتخاب و بر روی Next کلیک کنید.



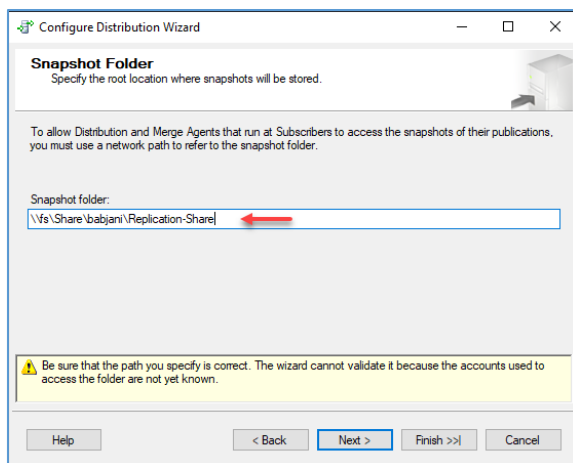
شکل ۵۴

در شکل ۵۵ گزینه‌ی اول را برای سرویس Agent انتخاب کنید تا سرویس به‌صورت اتوماتیکالی اجرا شود.



شکل ۵۵

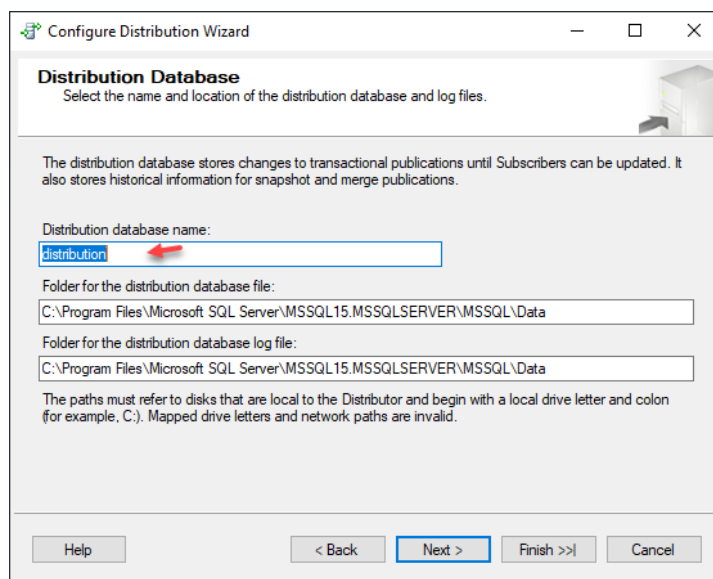
شکل ۵۶ مربوط به تعیین مسیر فولدري است که Snapshot های گرفته شده از تغييرات در آن کپی می شوند. برای اینکه Distribution agent و Merge agent های اجرا شده روی Subscriber ها بتوانند به Snapshot های publication ها دسترسی داشته باشند، در این قسمت باید مسیر فولدر در نظر گرفته شده برای ذخیره ی Snapshot ها را معرفی کنیم. در واقع باید آدرس شبکه ی پوشه ی نگهدارنده ی snapshot ها را مشخص کنیم. مسیر لوکال و پیش فرضی که در این قسمت قرار دارد، pull Subscription های ایجاد شده سمت مشترکین را پشتیبانی نمی کند چراکه این مسیر یک مسیر تحت شبکه نمی باشد و برای ایجاد pull Subscription باید یک مسیر شبکه ای را در اینجا معرفی کنیم که به پوشه ی Snapshot ها اشاره کند، توجه داشته باشید که مسیری که ایجاد می کنید باید



شکل ۵۶

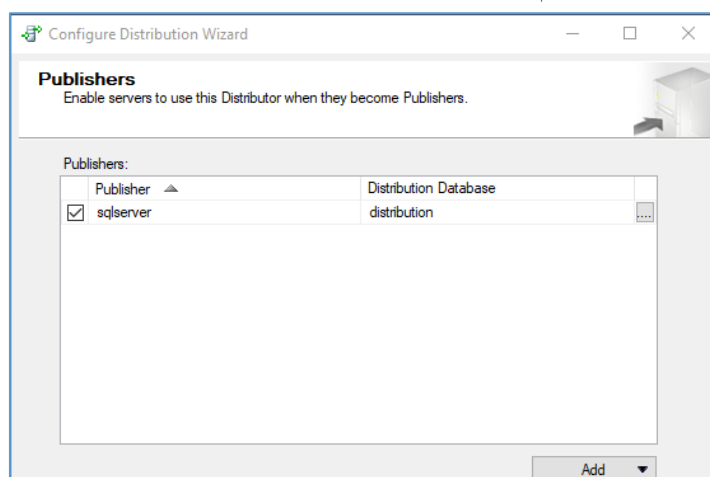
در شکل ۵۸، نام پایگاه داده ای Distribution و نیز محل ذخیره سازی آن را مشخص می کنیم. این پایگاه داده برای دادهایی با پسوند (MDF.) و برای Log فایل ها با پسوند (ldf.) پیکربندی می شود. پایگاه داده ای مربوط به Distribution تغییرات ایجاد شده در داده ها را در transactional publication ذخیره می کند تا زمانی که مشترکین یا همان Subscriber ها بتوانند آپدیت شوند. همچنین این دیتابیس اطلاعات را برحسب تاریخ برای Snapshot publication و merge publication ذخیره می کند. مسیری که در اینجا در نظر می گیریم حتماً باید یک آدرس لوکال باشد و با یک drive

letter و علامت دو نقطه شروع شود (به عنوان مثال C:) و در اینجا استفاده از Drive letter های map شده و مسیرهای تحت شبکه غیر مجاز می باشد.



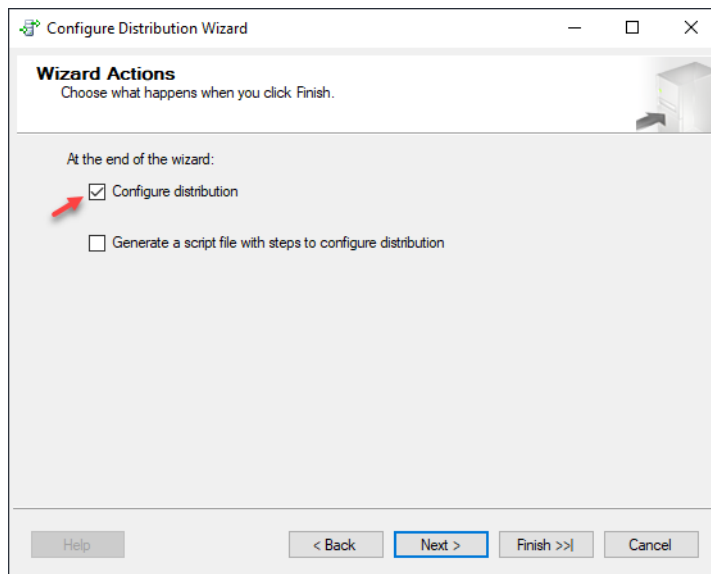
شکل ۵۸

سرور و یا سرورهایی که قرار است به عنوان ناشر و یا همان publisher فعالیت کنند را در شکل ۵۹ مشخص می کنیم. این سرور و یا سرورها از Distributor برای انجام توزیع انتشارات و یا publication های خود استفاده می کند. به صورت پیش فرض سرور فعلی به عنوان publisher در نظر گرفته می شود که بسته به ساختار پایگاه های داده و سناریوی مورد نظر با کلیک بر روی گزینه Add می توانیم سرورهای دیگری نیز برای این بخش در نظر بگیریم.



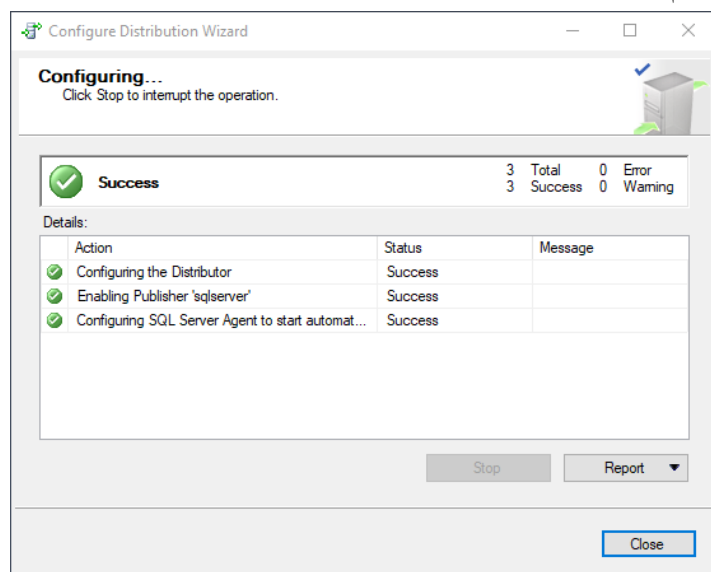
شکل ۵۹

در این قسمت چگونگی خاتمه یافتن تنظیمات انجام شده را مشخص می کنیم. به صورت پیش فرض گزینه ی Configure distribution انتخاب شده است.



شکل ۶۰

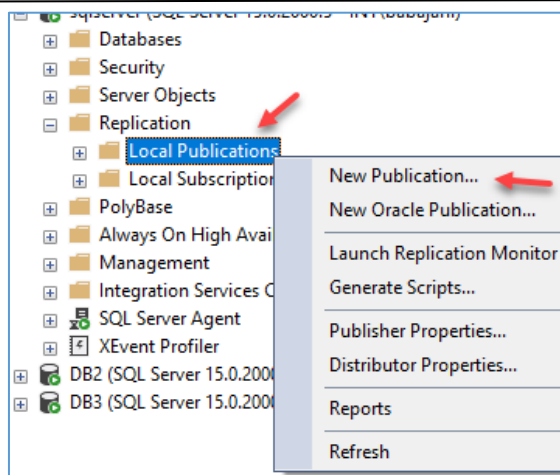
در نهایت در صورت عدم بروز مشکل، با نمایش عبارت success در قسمت وضعیت، انجام موفقیت‌آمیز تنظیمات را در شکل ۶۱ مشاهده خواهیم کرد.



شکل ۶۱

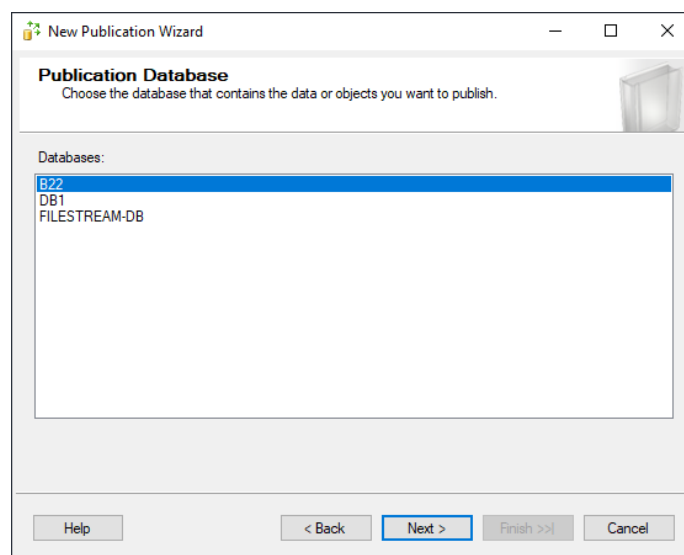
پیگر بندی نشریه یا Publication

انجام تنظیمات این بخش نیز روی سرور اول (SQLServer) صورت خواهد گرفت چراکه طبق سناریو این سرور هر دو نقش ناشر و توزیع‌کننده را ایفا می‌کند؛ بنابراین به‌مانند شکل ۶۲ روی گزینه Local Publication در کنسول مدیریتی راست کلیک کرده و گزینه New Publication را انتخاب می‌کنیم.



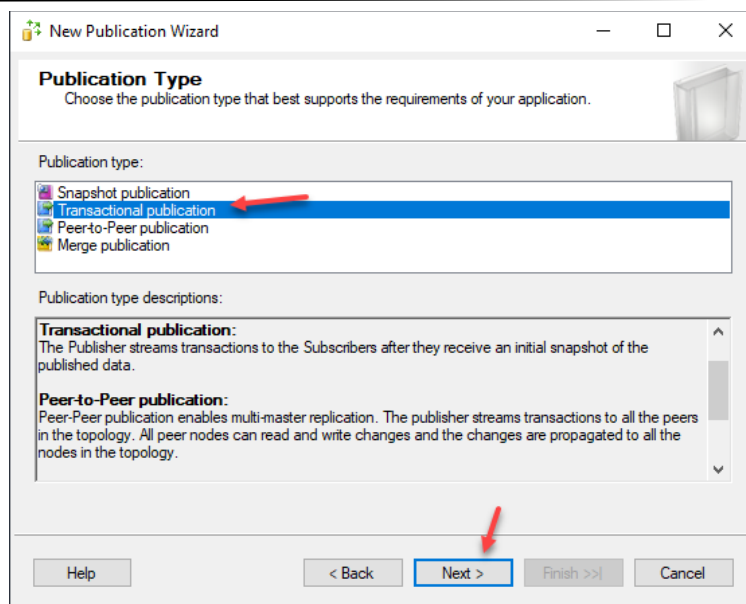
شکل ۶۲

در شکل ۶۳، لیستی از پایگاه‌های داده‌ی دربردارنده‌ی داده‌ها و یا اشیایی که قصد انتشار آنها را داریم نمایش داده می‌شود. در اینجا لیست همه‌ی دیتابیس‌های موجود را مشاهده می‌کنید، بر روی Next کلیک کنید.



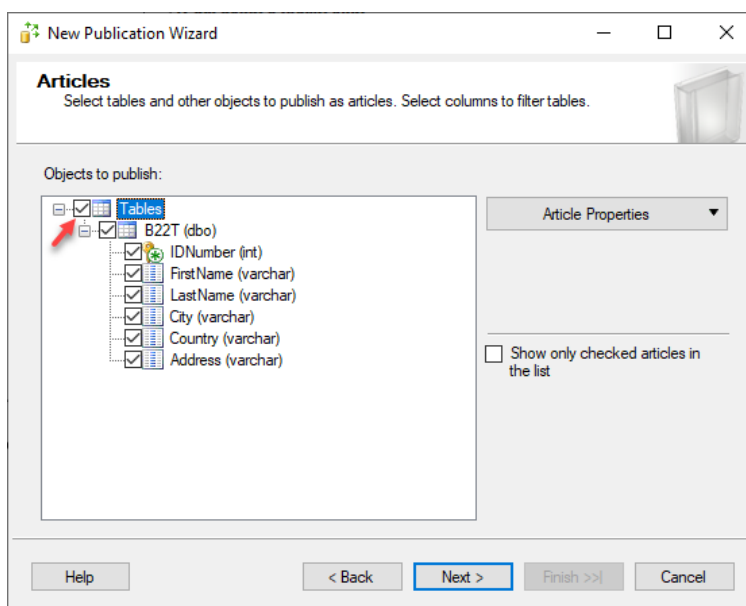
شکل ۶۳

در شکل ۶۴ انواع نشریه و یا همان publicationها نمایش داده می‌شود و در واقع بخشی است که در آن نوع replication مشخص می‌شود، در این قسمت گزینه‌ی Transactional publication را انتخاب کرده و روی Next کلیک می‌کنیم.



شکل ۶۴

در شکل ۶۵ نیز باید جداول و داده‌های مورد نظر که قرار است Replication شود را از لیست انتخاب کنید.

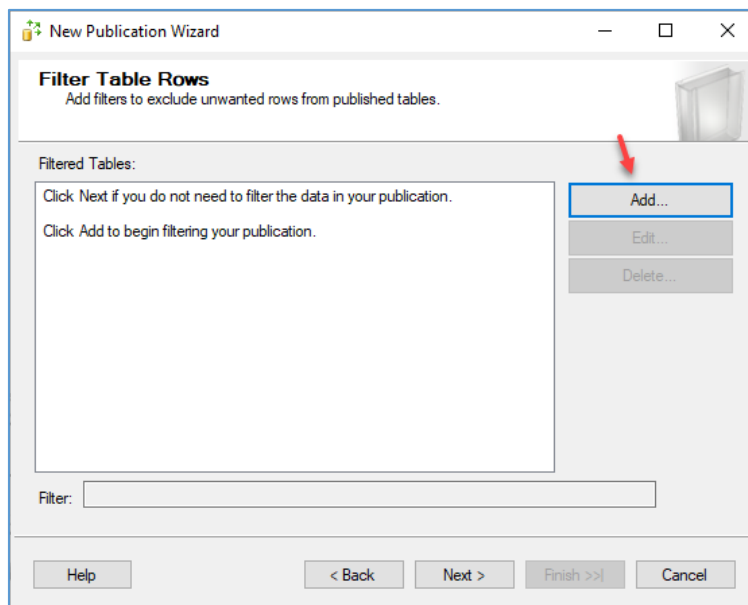


شکل ۶۵

در شکل ۶۵ با کلیک روی گزینه‌ی Article properties می‌توانیم جزئیات توضیحات مرتبط با جدولی را که انتخاب کردیم و یا توضیحات مربوط به تمامی اشیاء موجود در Article را مشاهده کرده و در صورت نیاز تغییراتی را در آن ایجاد کنیم. در این جا ما همان تنظیمات پیش فرض را دست‌نخورده باقی می‌گذاریم.

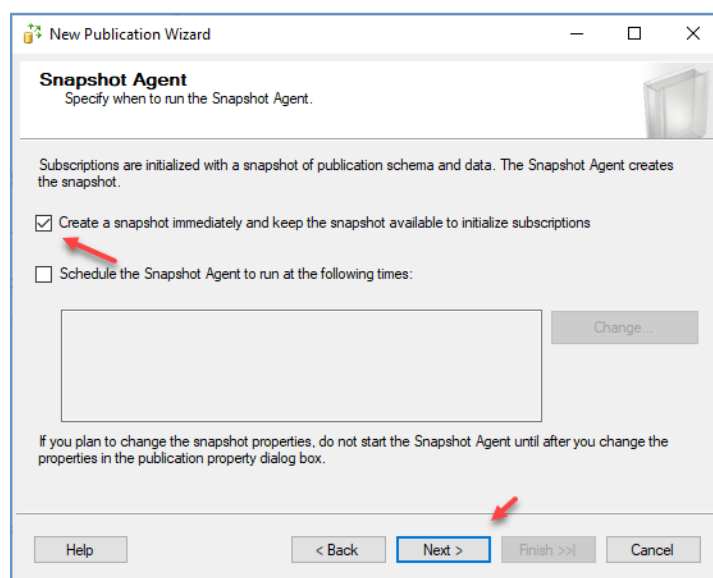
در شکل ۶۶ باید از عملیات فیلترینگ استفاده کنیم، از جمله دلایل استفاده از فیلتر می‌توان به، به‌حداقل‌رساندن داده‌های ارسالی روی شبکه و کاهش مقدار فضای ذخیره‌سازی برای یک مشترک را یادآور شد. در این بخش از تنظیمات با ایجاد فیلترینگ می‌توانیم به حذف ردیف‌هایی که به آنها نیاز نداریم از جدول منتشر شده بپردازیم. یک مثال ساده برای درک فیلترکردن این است که مثلاً نام و نام خانوادگی کاربران برتر تغییر نخواهد کرد بنابراین نیاز به

انتشار مجدد آن‌ها نداریم. در این تصویر با کلیک روی گزینه Add صفحه‌ی مربوط به ایجاد فیلترینگ نمایش داده می‌شود. در این سناریو ما هیچ‌گونه فیلتری را در نظر نگرفته و با کلیک روی Next به مرحله‌ی بعدی می‌رویم.



شکل ۶۶

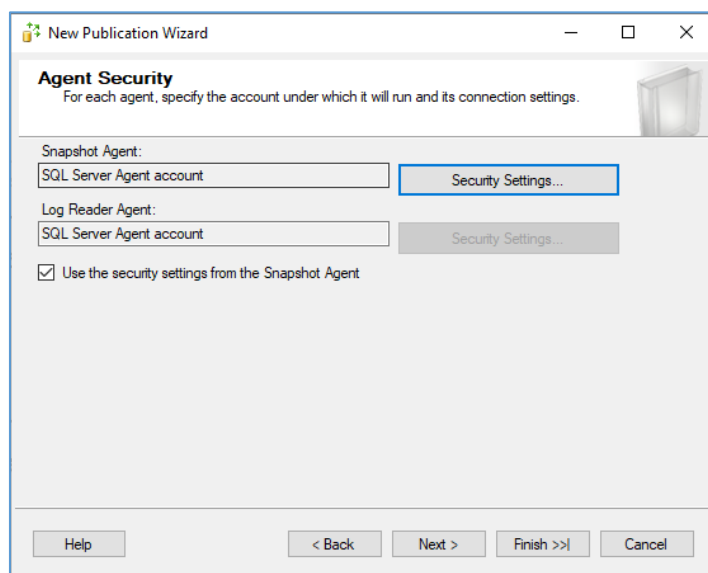
در شکل ۶۷ تعیین می‌کنیم که چه زمانی Snapshot agent اجرا شود. مشترکین به واسطه‌ی snapshotهای داده و طرح‌های یک نشریه مقداردهی می‌شوند و این Snapshot agent است که snapshot را ایجاد می‌کند. در این تصویر دو گزینه وجود دارد اولین گزینه بیانگر ایجاد یک Snapshot به صورت فوری و در دسترس نگه‌داشتن آن برای مقداردهی اولیه به مشترکین است و گزینه‌ی دوم شرایط زمان‌بندی را برای فعالیت Snapshot agent فراهم می‌آورد. گزینه‌ی اول را مارک‌دار می‌کنیم.



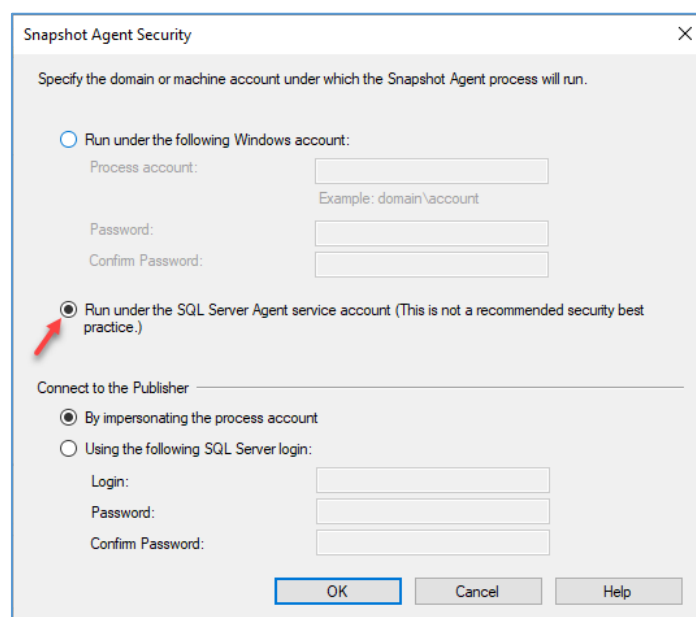
شکل ۶۷

شکل ۶۸ مربوط به مشخص کردن حساب کاربری برای هر Agent می‌باشد که اجرا و تنظیمات ارتباطات آن تحت این Account انجام خواهد شد. برای انجام این تغییرات روی گزینه‌ی Security Settings کلیک می‌کنیم. در شکل ۶۹،

گزینه‌ی Run under the SQL Server Agent account را مارک‌دار می‌کنیم. اگر سرویس SQL Server Agent سطح دسترسی و مجوزهای لازم را برای دسترسی به پوشه‌ای که برای نگهداری Snapshotها انتخاب کرده‌ایم نداشته باشد، باید روشی دیگر از احراز هویت حساب کاربری را برای فراهم‌آوردن این دسترسی در نظر بگیریم. در نهایت تنظیمات صورت گرفته همانند کادر کوچک مشخص شده در صفحه‌نمایش داده خواهند شد.

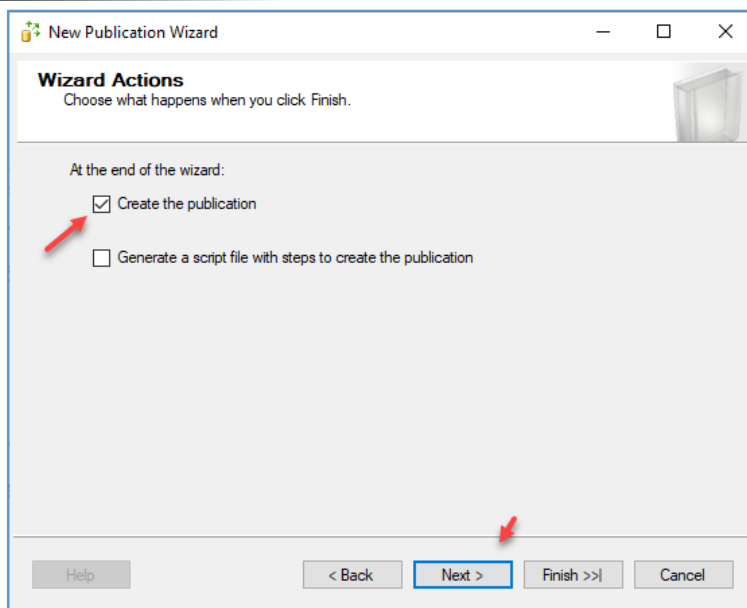


شکل ۶۸



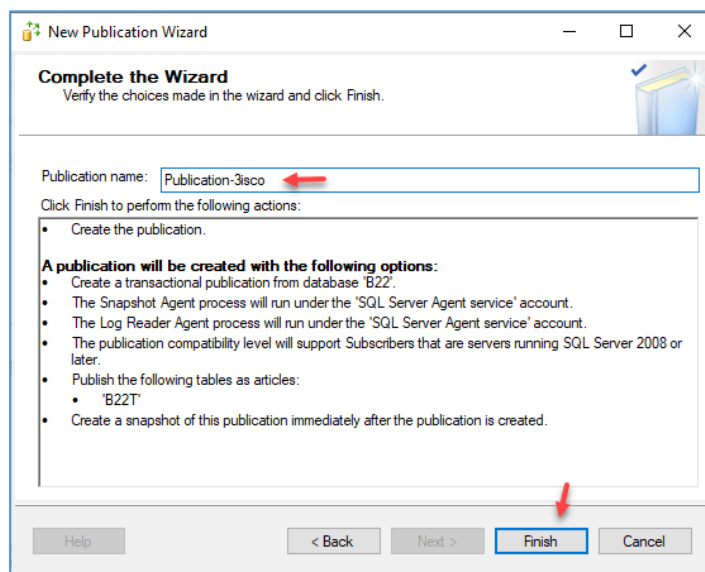
شکل ۶۹

در شکل ۷۰ چگونگی پایان یافتن تنظیمات انجام شده را مشخص می‌کنیم. به صورت پیش فرض گزینه‌ی Create the publication مارک‌دار شده است.



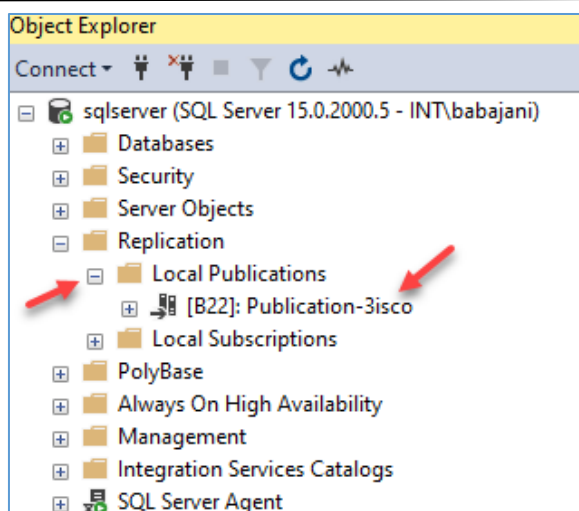
شکل ۷۰

در شکل ۷۱، خلاصه‌ای از تنظیماتی که انجام دادیم، نمایش داده می‌شود. نام مورد نظر برای نشریه یا publication را که ایجاد کرده‌ایم در این بخش مشخص وارد و در نهایت بر روی گزینه‌ی Finish کلیک می‌کنیم.



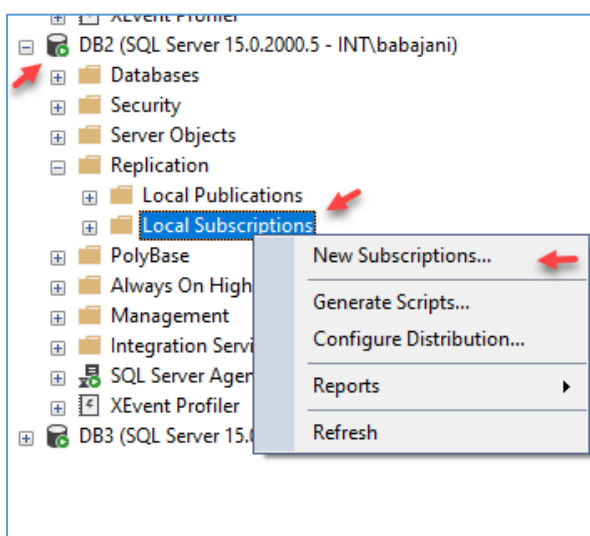
شکل ۷۱

پس از پیکربندی موفقیت‌آمیز Publication، می‌توانیم همانند تصویر Publication ایجاد شده را در بخش Replication کنسول مدیریتی مشاهده کنیم.



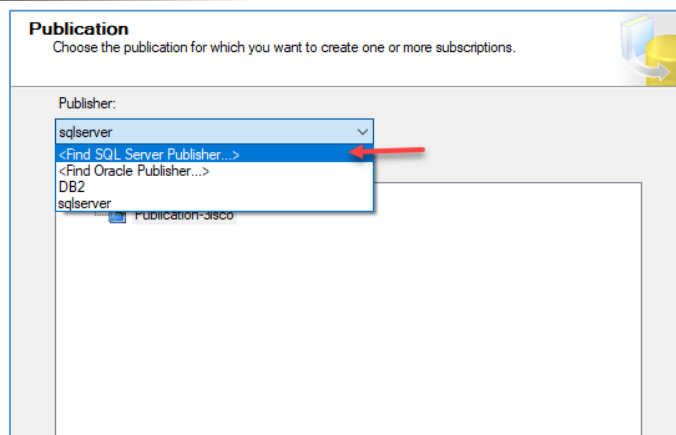
شکل ۷۲

همان‌طور که قبلاً گفتیم در اینجا هر سه سرور موجود در سناریو را از طریق کنسول Management Studio مدیریت می‌کنیم. در سرور دوم یعنی DB2، به‌مانند شکل ۷۲، بر روی Local Subscription زیرمجموعه‌ی Replication این سرور در کنسول مدیریتی راست کلیک کرده و گزینه‌ی New Subscription را انتخاب می‌کنیم.



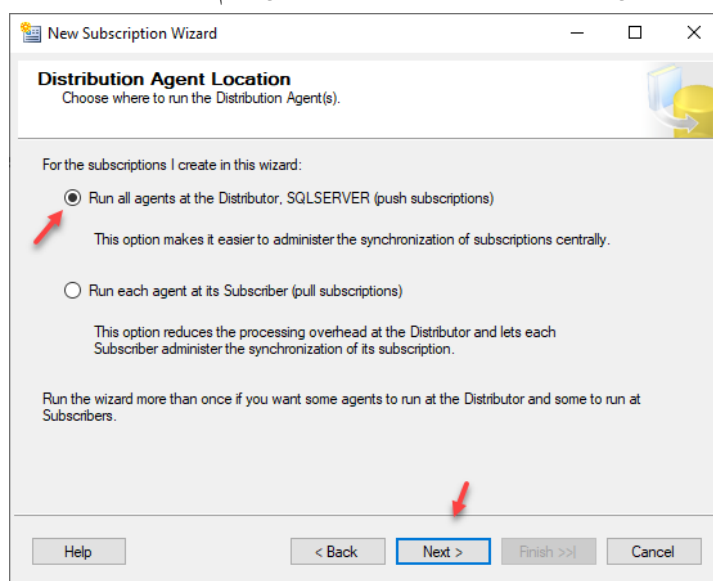
شکل ۷۳

همان‌طور که در شکل ۷۴، نشان‌داده شده است باید سرور publisher و یا نشریه‌ای را که می‌خواهیم Subscriptionها را برای آن ایجاد کرده، تعیین کنیم. پس در پنجره‌ی publisher از قسمت نوار کشویی گزینه‌ی Find SQL Server را باز کرده و پس از انتخاب سرور مورد نظر (در اینجا SQLServer) نام publication ایجاد شده در لیست نمایان خواهد شد.



شکل ۷۴

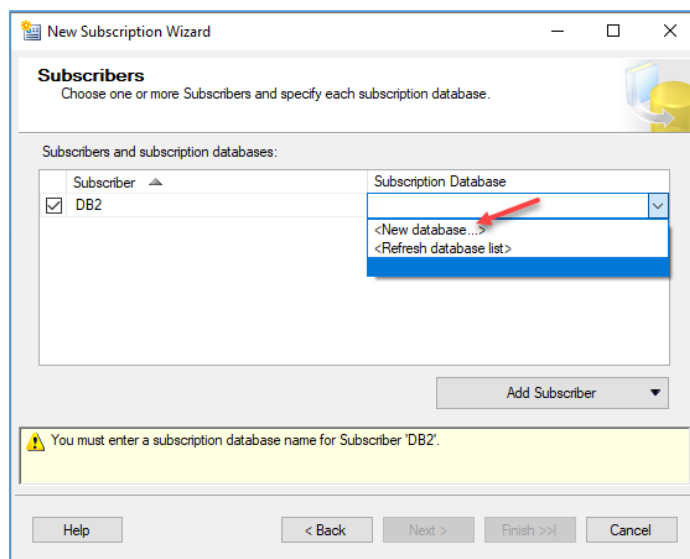
در شکل ۷۵، باید محلی که Distribution Agentها روی آن اجرا می‌شوند را مشخص کنیم. برای این منظور دو گزینه پیشروی ماست: اجرای همه‌ی Agentها روی Distributor یا Push subscription و اجرای هر Agent روی subscriber خود یا Pull subscription. در اوایل این فصل در مورد replication به توضیح Pull Subscription\pushها پرداختیم برحسب شرایط می‌توان یکی از این دو نوع را برای Agentها در نظر گرفت. در قسمت قبل ما یک مسیر لوکال را برای دسترسی به پوشه‌ی Snapshotها تعیین کردیم و می‌دانیم آدرس لوکال Pull Subscription را پشتیبانی نمی‌کند، بنابراین در اینجا گزینه‌ی اول یعنی Push Subscription را انتخاب می‌کنیم.



شکل ۷۵

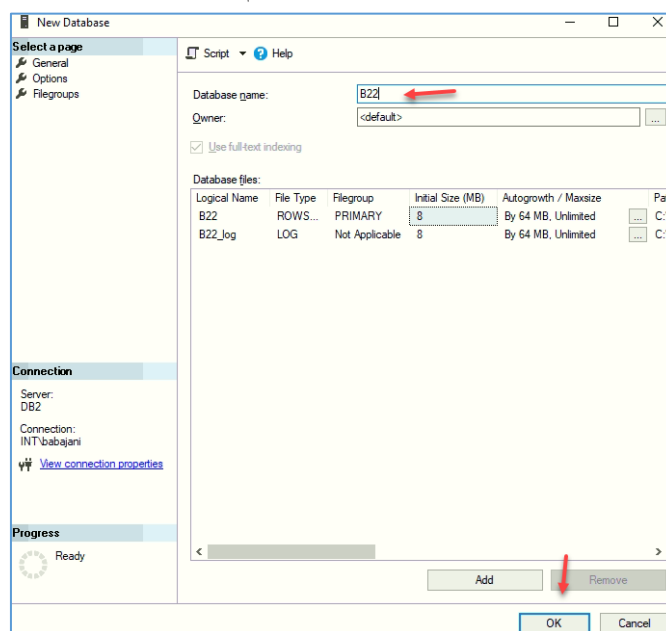
در این قسمت باید subscriberها یا مشترکین و دیتابیس آنها را مشخص کنیم. به صورت پیش فرض ویزارد سروری را که تنظیمات در حال حاضر روی آن صورت می‌گیرد به عنوان یکی از مشترکین انتخاب می‌کند. با کلیک روی گزینه‌ی Add SQL Server Subscriber می‌توانیم سرورهای دیگری را به عنوان مشخص کنیم. در سناریو ما دو Subscriber وجود دارد: سرورهای DB2 و DB3. پس از انتخاب این سرورها حال نوبت به تعیین دیتابیس‌ی که قرار است عمل Replicate با آنها صورت گیرد، می‌باشد. بدین منظور به مانند شکل ۷۶ با کلیک روی نوار کشویی مقابل سرورهای انتخابی،

دیتابیس مورد نظر را برمی‌گزینیم. اگر دیتابیس را روی سرور برای Replication در نظر نگرفته باشیم، با توجه به شکل ۷۶ و کلیک روی گزینه‌ی New Data base نیز می‌توان دیتابیس مورد نظر را در لحظه ایجاد کرد.



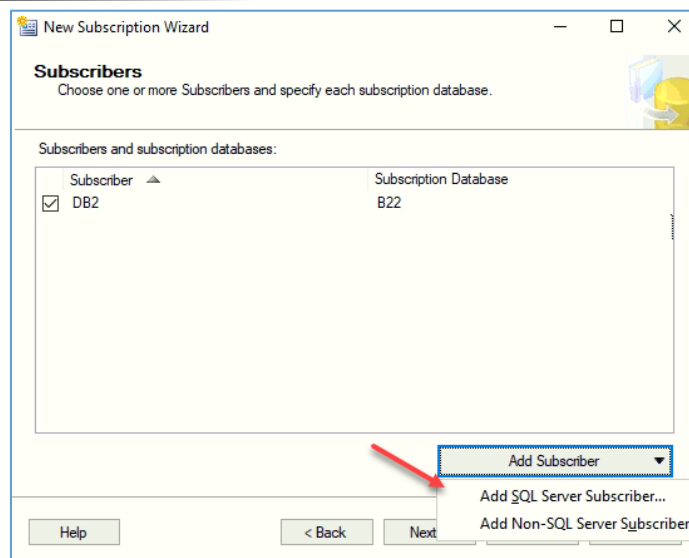
شکل ۷۶

در شکل ۷۷ باید دیتابیس مربوطه که در قسمت قبلی انتخاب کردیم را ایجاد کنید و بر روی OK کلیک کنید.



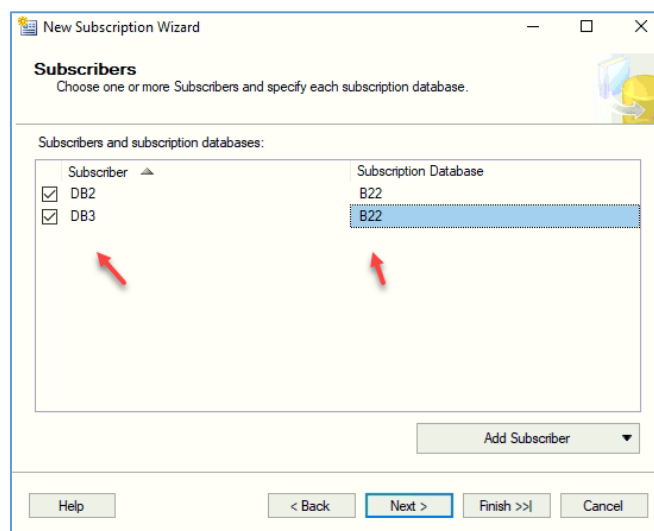
شکل ۷۷

همان‌طور که در شکل ۷۸ مشاهده می‌کنید دیتابیس مورد نظر ایجاد شده است، و در ادامه باید سرور سوم یعنی FB3 را هم به لیست اضافه کنید و برای آن هم دیتابیس DB22 را ایجاد کنید، برای این کار بر روی Add Subscriber کلیک کنید و گزینه‌ی Add SQL Server Subscriber را انتخاب کنید.



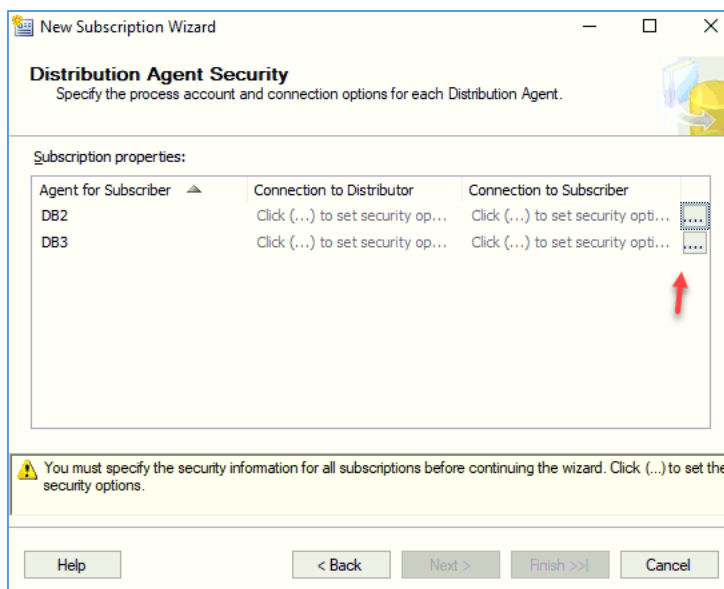
شکل ۷۸

همان‌طور که در شکل ۷۹ مشاهده می‌کنید سرور DB3 هم به لیست اضافه شده و دیتابیس مورد نظر را هم برای آن ایجاد کردیم.



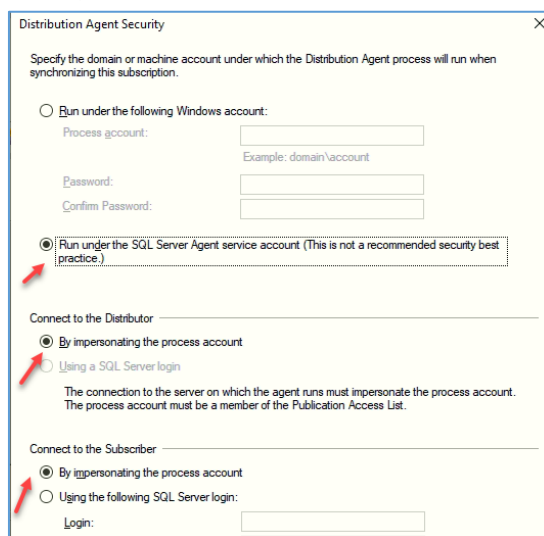
شکل ۷۹

در شکل ۸۰ باید آپشن‌های مرتبط با ارتباطات و پردازش‌ها را برای Distribution Agent مشخص کنیم، بر روی گزینه‌ی مورد نظر کلیک کنید.



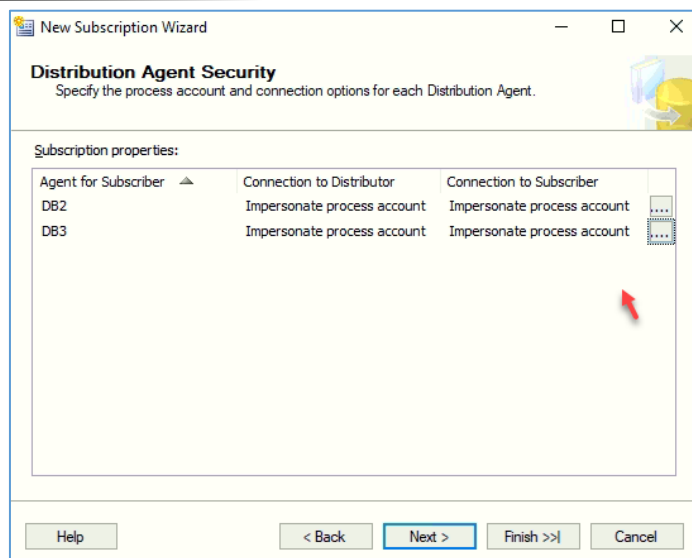
شکل ۸۰

برای تعیین حساب کاربری که Distribution Agent تحت آن اجرا شده و عمل همگام‌سازی با مشترکین را انجام خواهد داد به‌مانند شکل ۸۱ گزینه‌ی Run the SQL Server Agent Service Account را انتخاب کرده و برای برقراری ارتباط با Distributor و Subscriber نیز گزینه‌ی By impersonation the process account را برمی‌گزینیم. این تنظیمات باید برای هر دو سرور انتخاب شده به‌عنوان Subscriber، به‌صورت مجزا انجام شود.



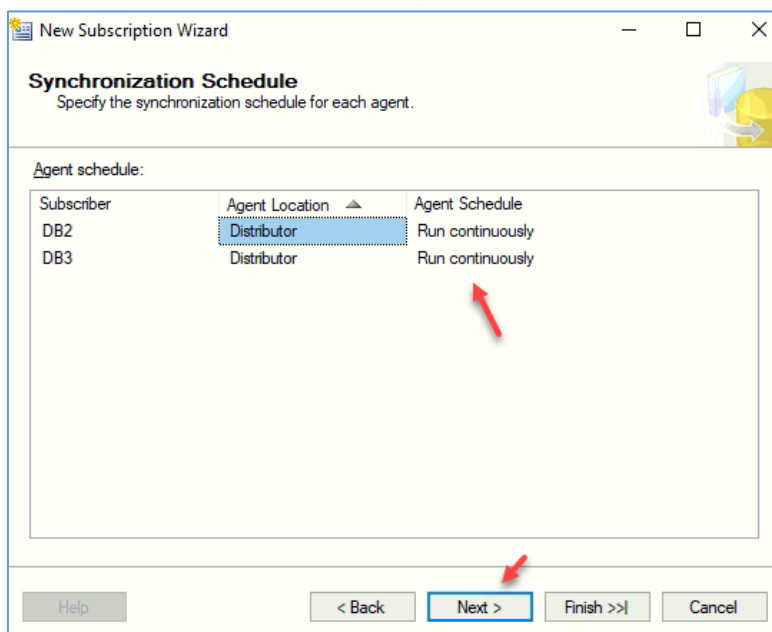
شکل ۸۱

پس از انجام تنظیمات بالا نتیجه‌ی آن همانند شکل ۸۲ خواهد بود، بر روی Next کلیک کنید.



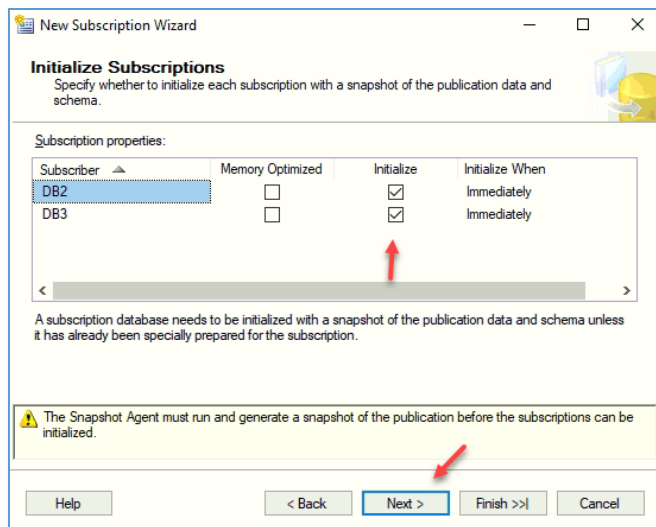
شکل ۸۲

در شکل ۸۳، باید برنامه‌ی زمان‌بندی همگام‌سازی را برای هر Agent مشخص کنیم. منوی کشویی در اینجا سه گزینه در اختیار ما قرار می‌دهد که بواسطه‌ی آنها Distribution agent به ترتیب می‌تواند در حالت اجرا به صورت مداوم، اجرا بر حسب تقاضا و یا اجرا بر اساس برنامه‌ی زمان‌بندی تعیین شده، قرار گیرد. پس از اطمینان از قرارگرفتن agent در وضعیت Run continuously روی Next کلیک می‌کنیم.



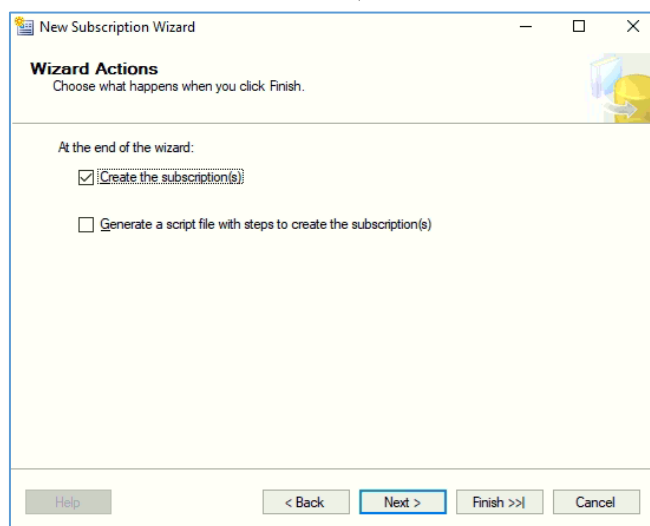
شکل ۸۳

نحوه‌ی مقداردهی اولیه هر اشتراک به‌واسطه‌ی یک Snapshot از داده‌ها و طرح‌های Publication یا نشریه در این قسمت مشخص می‌شود. دو روش برای مقداردهی اولیه‌ی Subscriptionها وجود دارد: به صورت فوری و در اولین همگام‌سازی یا Synchronization. از منوی کشویی موجود گزینه‌ی Immediately یا بلافاصله را برای هر دو سرور موجود انتخاب کرده و روی Next کلیک می‌کنیم.



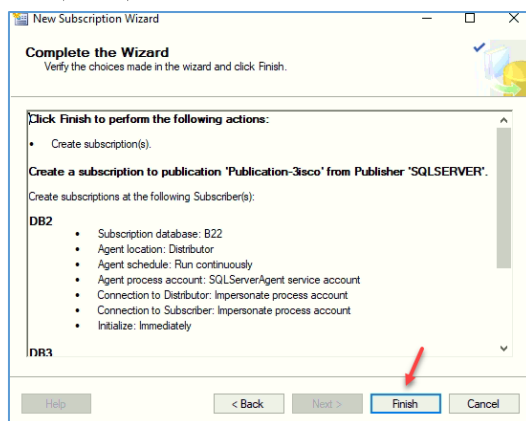
شکل ۱۴

در شکل ۸۳ می‌توانیم تعیین کنیم که تنها Subscription ایجاد شده و یا فایل اسکریپت پیکربندی Subscription نیز ایجاد شود. در این بخش گزینه‌ی اول را انتخاب می‌کنیم.



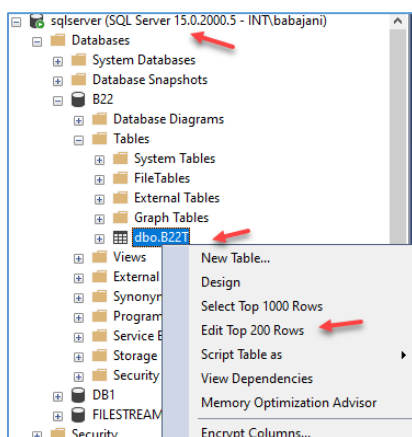
شکل ۱۳

در شکل ۸۴، خلاصه‌ای از تنظیماتی که برای ایجاد Subscriptionها انجام دادیم نمایش داده شده است.



شکل ۱۴

با کلیک روی گزینه‌ی Finish در شکل ۸۴، فرایند ایجاد subscriptionها آغاز خواهد شد و در نهایت با نمایش عبارت success در قسمت Status، انجام موفقیت‌آمیز تنظیمات را مشاهده خواهیم کرد. خوب بعد از ایجاد Replication حالا هر تغییری در سرور اصلی (SQLServer) ایجاد شود در سرورهای دیگر (DB2,DB3) هم انجام خواهد شد که این موضوع را می‌توانیم بررسی کنیم، طبق شکل ۸۵ در سرور اصلی یعنی SQLServer بر روی جدول مورد نظر خود کلیک راست کنید و گزینه‌ی Edit Top را انتخاب کنید.



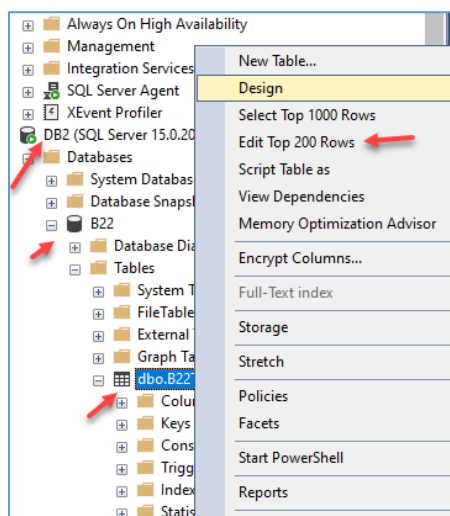
شکل ۸۵

یک سطر جدید به‌مانند شکل ۸۶ به جدول مورد نظر اضافه می‌کنیم، بعد از این کار این تغییرات به‌سرعت و ظرف مدت چند ثانیه به سرورهای دیگر هم ارسال می‌شود.

IDNumber	FirstName	LastName	City	Country	Address
821620324	Negar	Sistani	Karaj	IR	MohamadShahr
848910078	Azadeh	Mohebi	Shiraz	IR	Shiraz1
856520103	Elham	bozorgi	ardebil	IR	Azarbajan
884320101	ahmad	Mohamad	babol	IR	Kordmahaleh
892612101	alireza	nafeei	khozestan	IR	khozestan
884320102	ali	mohebi	babol	IR	Mazandaran
NULL	NULL	NULL	NULL	NULL	NULL

شکل ۸۶

برای اینکه تغییرات را مشاهده کنیم به‌مانند شکل ۸۷ وارد سرور DB2 شوید و بر روی جدول مورد نظر کلیک راست و گزینه‌ی Edit Top را انتخاب کنید.



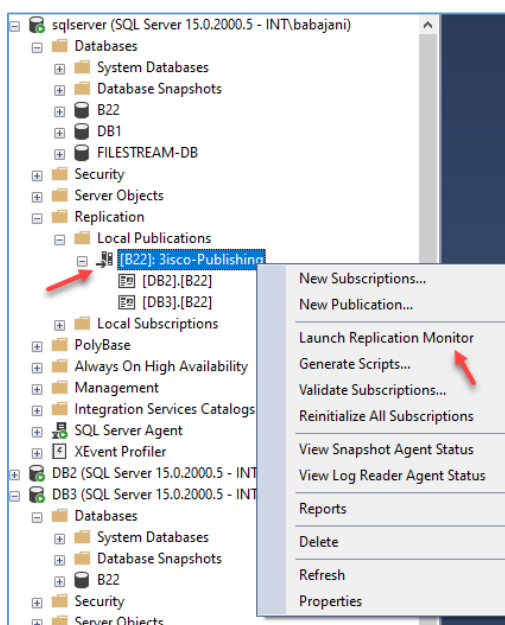
شکل ۸۷

همانطور که مشاهده می‌کنید در شکل ۸۸ تغییرات به درستی انجام شده است.

IDNumber	FirstName	LastName	City	Country	Address
821620324	Negar	Sistani	Karaj	IR	MohamadShahr
848910078	Azadeh	Mohebi	Shiraz	IR	Shiraz1
856520103	Elham	bozorgi	ardebil	IR	Azarbajjan
884320101	ahmad	Mohamad	babol	IR	Kordmahaleh
884320102	ali	mohebi	babol	IR	Mazandaran
892612101	alireza	nafeei	khozestan	IR	khozestan
NULL	NULL	NULL	NULL	NULL	NULL

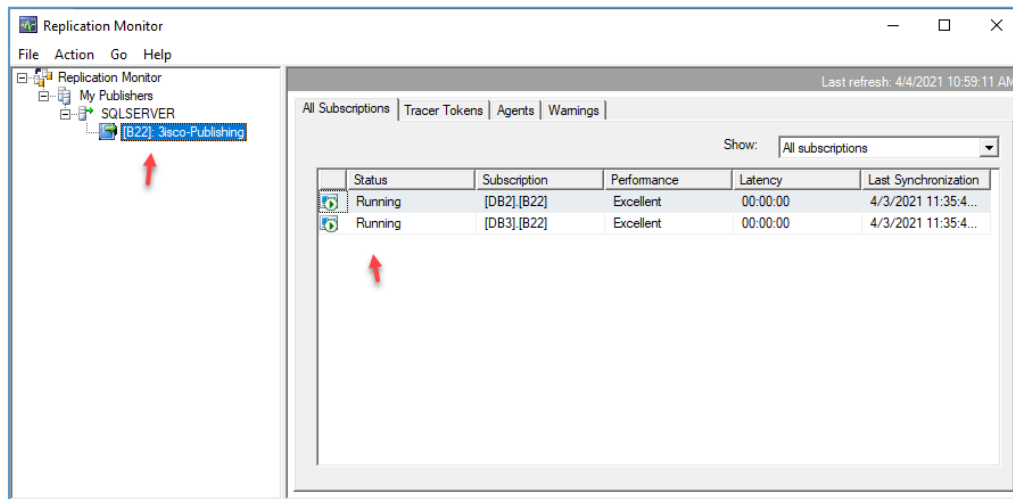
شکل ۸۸

برای این که سرویس Replication را مانیتور کنیم، باید به مانند شکل ۸۹ بر روی Publications مورد نظر که با هم ایجاد کردیم کلیک راست کنید و گزینه Lunch Replication Monitor انتخاب کنید.



شکل ۸۹

همانطور که در شکل ۹۰ مشاهده می‌کنید هر دو سرور DB2 و DB3 در حال کار با بهترین Performance هستند، اگر اطلاعات بیشتری از هر سرور بخواهید می‌توانید بر روی آنها دو بار کلیک کنید.



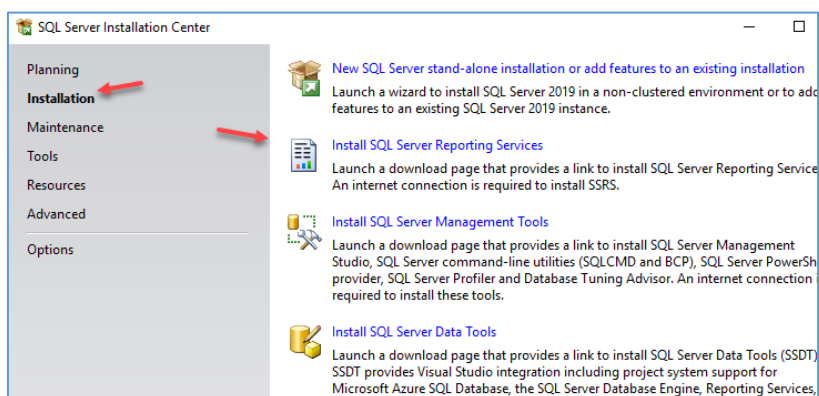
شکل ۹۰

فصل هفتم

SQL Reporting Service

سرویس SSRS یا همان SQL Server Reporting Service یک سرویس برای گزارش‌گیری از عملکرد شما در SQL است و حتی سازمان‌هایی که از SQL استفاده نمی‌کنند این قابلیت وجود دارد که سرویس Reporting به پایگاه داده‌های دیگر متصل شود و یک گزارش کامل از آنها تهیه کند، در نظر داشته باشید این سرویس بسیار کارآمد است و بهتر است در سازمان خود برای ارائه گزارش کامل فعال کنید.

برای راه‌اندازی این سرویس در ورژن‌های جدید SQL باید فایل آن را از سایت مایکروسافت دریافت و در سرور SQL Server که با هم راه‌اندازیم نصب کنید، برای دانلود این سرویس به شکل ۱-۱۰ توجه کنید، در این صفحه بر روی **Install SQL Server Reporting Services** کلیک کنید، بعد از کلیک وارد سایت مایکروسافت خواهید شد.



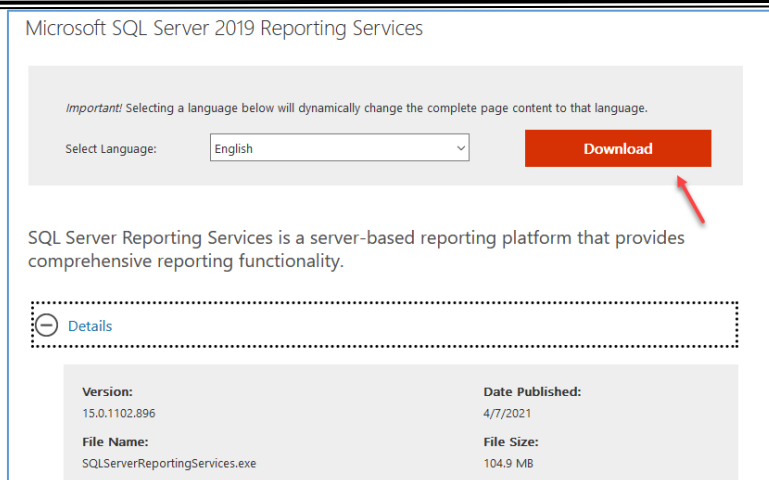
شکل ۱-۱۰

در شکل ۱۰-۲ بر روی لینک دانلود مورد نظر کلیک کنید.



شکل ۱۰-۲

در شکل ۱۰-۳ بر روی **Download** کلیک کنید و در ادامه فایل مورد نظر را نصب کنید.



شکل ۱۰-۳

بعد از دانلود فایل شکل ۱۰-۴ ظاهر خواهد شد که بر روی **Install Reporting Services** کلیک کنید.



شکل ۱۰-۴

در شکل ۱۰-۵ باید لایسنس نرم افزار را وارد و بر روی **Next** کلیک کنید و در ادامه هم بر روی **Next** و **Install** کلیک کنید.



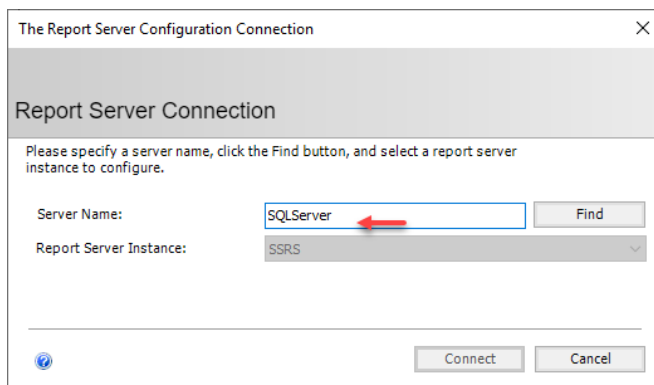
شکل ۱۰-۵

در ادامه کار و بعد از نصب شکل ۱۰-۶ را مشاهده خواهید کرد که برای انجام تنظیمات تکمیلی بر روی **Configure Report Server** کلیک کنید.



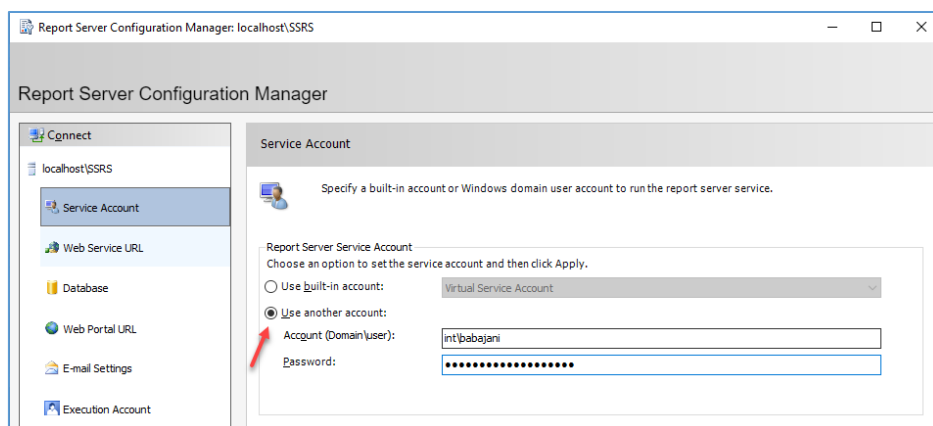
شکل ۱۰-۶

در شکل ۱۰-۷ باید نام سرور SQL خود را وارد و بر روی **Connect** کلیک کنید.



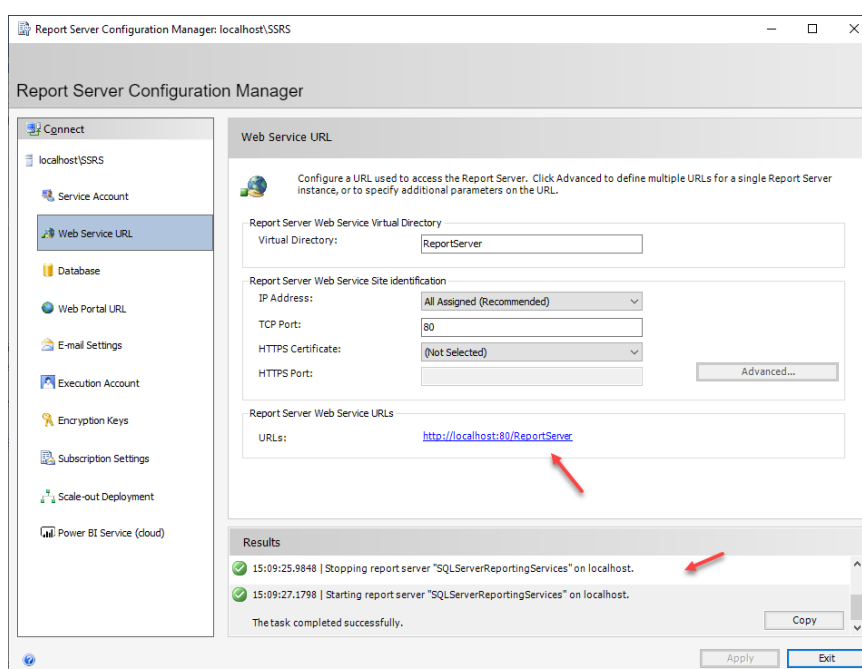
شکل ۱۰-۷

در شکل ۸-۱۰ باید تنظیمات مربوط به Reporting Service را انجام دهید برای شروع وارد Service Account شوید و یک کاربر تحت دومین را به‌مانند شکل وارد کنید، بعد از وارد کردن کاربر مورد نظر بر روی Apply کلیک کنید تا اطلاعات ثبت شود.



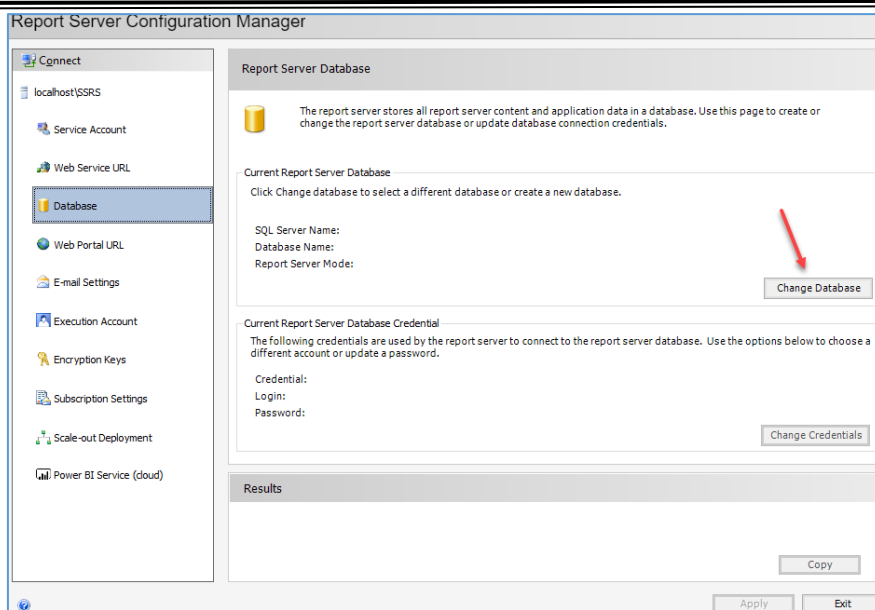
شکل ۸-۱۰

در قسمت Web Service URL که در شکل ۹-۱۰ می‌توانید سرویس تحت وب Reporting را فعال کنید شما فقط کافی است بر روی Apply کلیک کنید تا تنظیمات مورد نظر بر روی سرویس IIS انجام شود، همان‌طور که مشاهده می‌کنید بعد از کلیک بر روی Apply تنظیمات مورد نظر انجام شده و آدرس دسترسی به وب سرویس Reporting مشخص شده است.



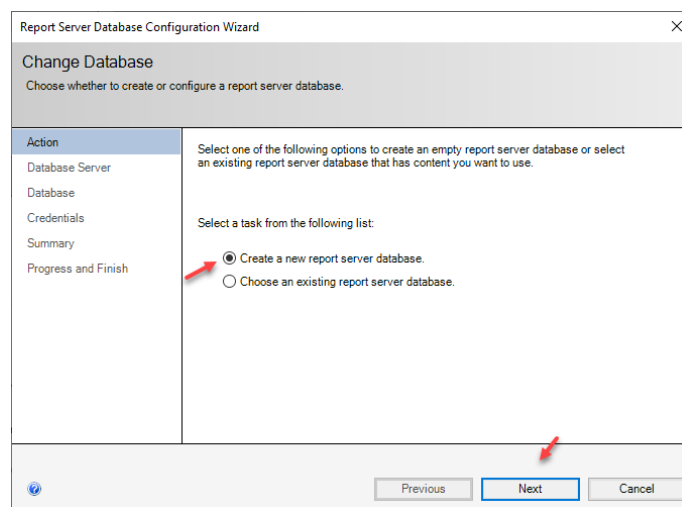
شکل ۹-۱۰

در شکل ۱۰-۱۰ باید برای سرویس Reporting دیتابیس ایجاد کنیم برای این کار به‌مانند شکل ۱۰-۱۰ بر روی Change Database کلیک کنید.



شکل ۱۰-۱۰

در ادامه باید یک دیتابیس جدید ایجاد کنیم برای این کار به مانند شکل ۱۰-۱۱ گزینه‌ی **Create a new report server database** را انتخاب کنید تا دیتابیس جدید را ایجاد کنیم، اگر هم از قبل ایجاد کردید باید گزینه‌ی دوم را انتخاب کنید.



شکل ۱۰-۱۱

در شکل ۱۰-۱۲ باید نام سرور SQL که قرار است بر روی آن دیتابیس را ایجاد کنیم را در قسمت اول وارد کنید، اگر توجه کنید ما localhost وارد کردیم چون در حال کار بر روی سرور SQL هستیم و می‌توانیم localhost هم وارد کنیم و یا اینکه نام آن سرور را وارد کنید، در قسمت بعد باید نوع احراز هویت را مشخص کنید و کاربر مورد نظر خود را وارد کنید، می‌توانید با کلیک بر روی **Test Connection** از ارتباط با سرور مطمئن شوید.

Report Server Database Configuration Wizard

Change Database

Choose whether to create or configure a report server database.

Action

Database Server

Database

Credentials

Summary

Progress and Finish

Choose a local or remote instance of a SQL Server Database Engine and specify credentials that have permission to connect to that server.

Connect to the Database Server:

Server Name: localhost

Authentication Type: SQL Server Account

Username: nt\babajani

Password:

Test Connection

Previous Next Cancel

شکل ۱۰-۱۲

در شکل ۱۰-۱۳ باید نام دیتابیس مربوط به Report Service را وارد و زبان مورد استفاده آن را انتخاب کنید.

Report Server Database Configuration Wizard

Change Database

Choose whether to create or configure a report server database.

Action

Database Server

Database

Credentials

Summary

Progress and Finish

Enter a database name and select the language to use for running SQL scripts.

Database Name: ReportServer

Temp Database Name: ReportServerTemp

Language: English (United States)

Report Server Mode: Native

Previous Next Cancel

شکل ۱۰-۱۳

در صفحه‌ی شکل ۱۰-۱۴ بر روی Next کلیک کنید.

Report Server Database Configuration Wizard

Change Database

Choose whether to create or configure a report server database.

Action

Database Server

Database

Credentials

Summary

Progress and Finish

Specify the credentials of an existing account that the report server will use to connect to the report server database. Permission to access the report server database will be automatically granted to the account you specify.

Credentials:

Authentication Type: Service Credentials

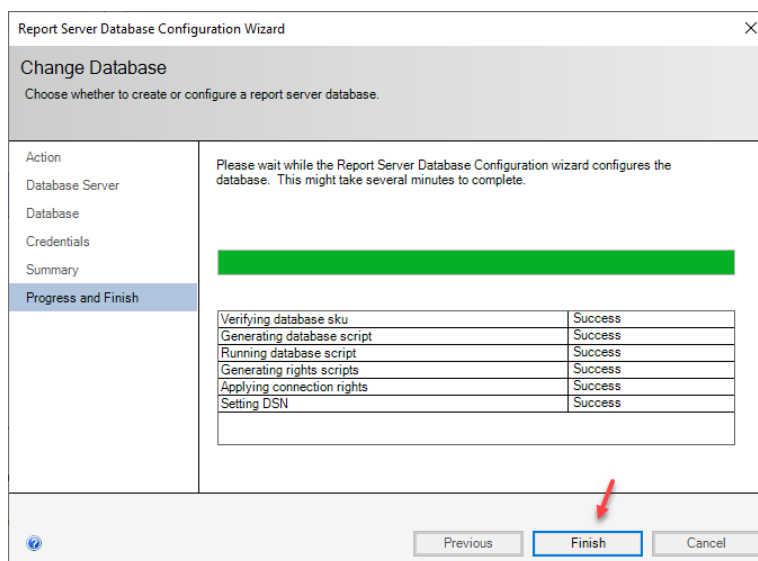
User name: NT SERVICE\SQLServerReportingServices

Password:

Previous Next Cancel

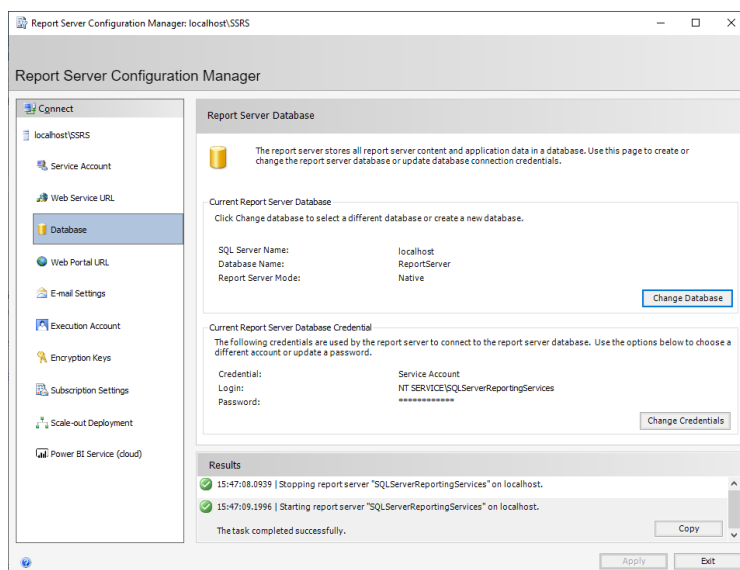
شکل ۱۰-۱۴

نتیجه نهایی کار باید مانند شکل ۱۰-۱۵ باشد، بر روی Finish کلیک کنید.



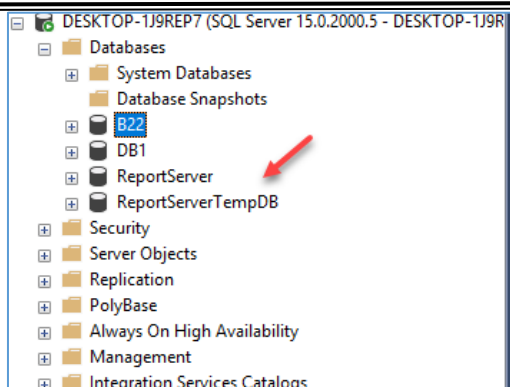
شکل ۱۰-۱۵

همان‌طور که در شکل ۱۰-۱۶ مشاهده می‌کنید دیتابیس مورد نظر به درستی ایجاد شده است و بر روی دکمه Exit کلیک کنید.



شکل ۱۰-۱۶

اگر وارد SQL Server Engine شوید دو دیتابیس ایجاد شده است که دیتابیس ReportServer / دیتابیس اصلی است و دیتابیس ReportServerTempDB برای نگه‌داشتن موقت اطلاعات در حال اجرا.

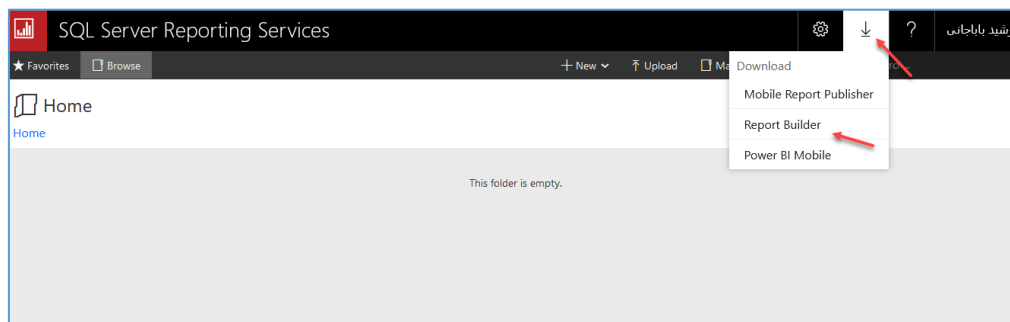


شکل ۱۰-۱۷

بعد از انجام مراحل بالا اگر مرورگر خود را باز کنید و آدرس زیر را اجرا کنید شکل ۱۸-۱۰ را مشاهده خواهید کرد.

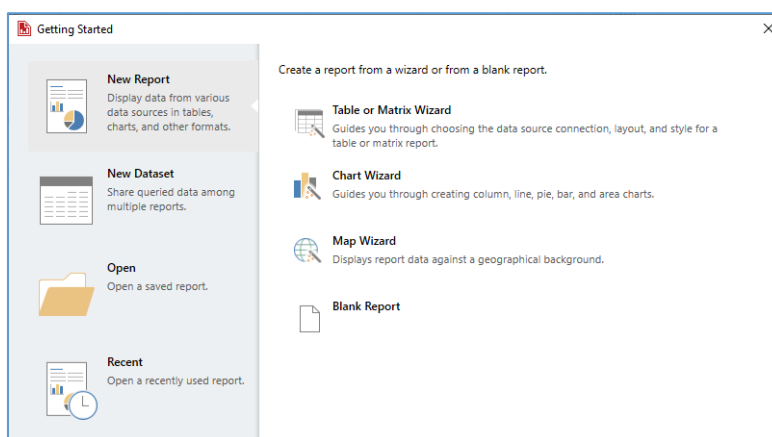
<http://sqlserver/reports/>

در شکل ۱۸-۱۰ صفحه‌ی اصلی گزارش‌گیری را مشاهده می‌کنید که دارای گزینه‌های مختلف است، برای اینکه بتوانیم در SQL یک گزارش ایجاد و به آن نما دهیم باید از ابزارهای مختلفی استفاده کنیم و بهترین و در دسترس‌ترین ابزار، Report Builder است که باید آن را به‌مانند شکل ۱۸-۱۰ دانلود و بر روی سرور مورد نظر نصب کنید، البته می‌توانید آن را در سیستم‌های دیگر هم نصب کنید.



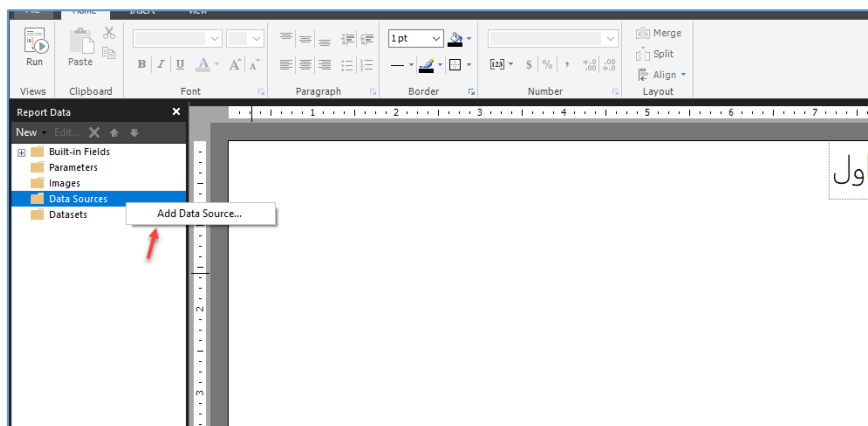
شکل ۱۰-۱۸

بعد از اجرا، اولین صفحه‌ای که باز می‌شود مانند شکل ۱۹-۱۰ است که برای شروع کار بر روی Blank Report کلیک کنید.



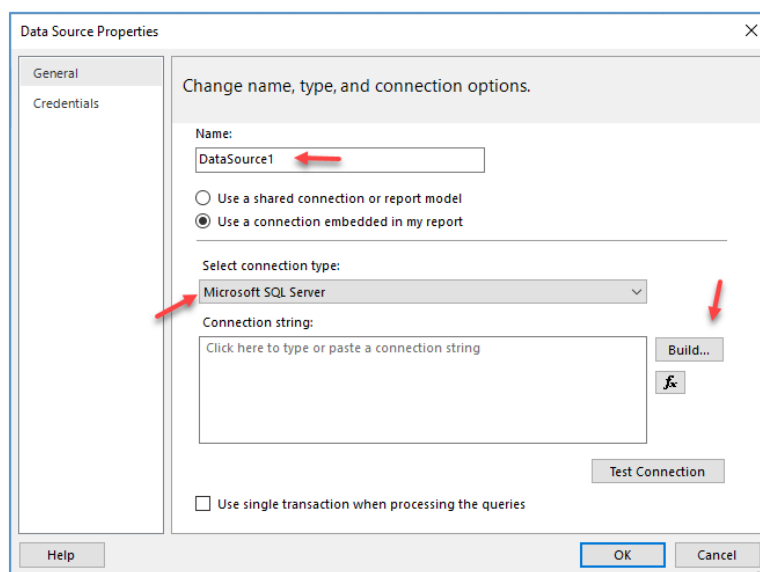
شکل ۱۰-۱۹

در شکل ۱۰-۲۰ باید طراحی خود را بر روی صفحه ایجاد کنید که این طراحی بستگی به نظر شما هم دارد شاید بخواهید از عکس استفاده کنید و یا چیزهای دیگر، اما اصل کار این است که بتوانیم جداول خود را وارد این گزارش‌ها کنیم، برای این کار بر روی Data Sources کلیک راست کنید و بر روی Add Data Sources کلیک کنید.



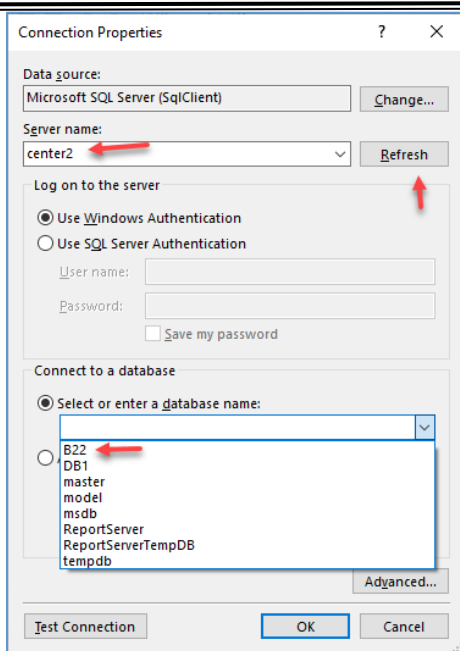
شکل ۱۰-۲۰

در شکل ۱۰-۲۱ نام مورد نظر خود را وارد کنید و در قسمت Select Connection Type گزینه‌ی Microsoft SQL Server را انتخاب کنید و بر روی Build کلیک کنید.



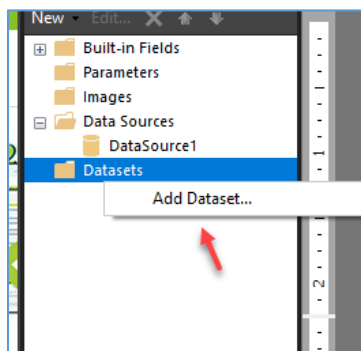
شکل ۱۰-۲۱

در شکل ۱۰-۲۲ در قسمت Server Name باید نام سرور SQL را وارد کنید، بعد از وارد کردن نام سرور بر روی Refresh کلیک کنید، بعد از کلیک بر روی دکمه Refresh نام دیتابیس‌های مورد نظر در SQL در قسمت زیری نمایش داده خواهد شد که شما باید یکی از دیتابیس‌ها که قرار است از آن گزارش تهیه کنید را انتخاب و بر روی OK کلیک کنید تا این ارتباط به لیست اضافه شود.



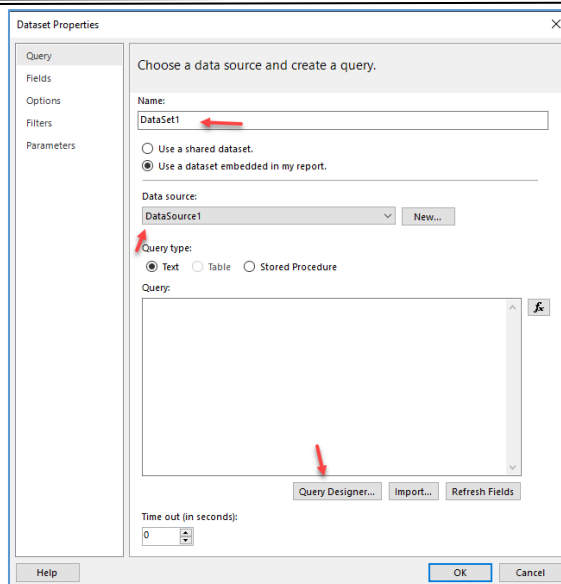
شکل ۱۰-۲۲

در شکل ۱۰-۲۳ بعد از ایجاد Data Source باید یک Datasets ایجاد کنید تا بتوانیم اطلاعات موجود در جداول را در خروجی نمایش دهیم برای این کار بر روی Datasets مورد نظر کلیک راست کنید و گزینه‌ی Add Dataset را انتخاب کنید.



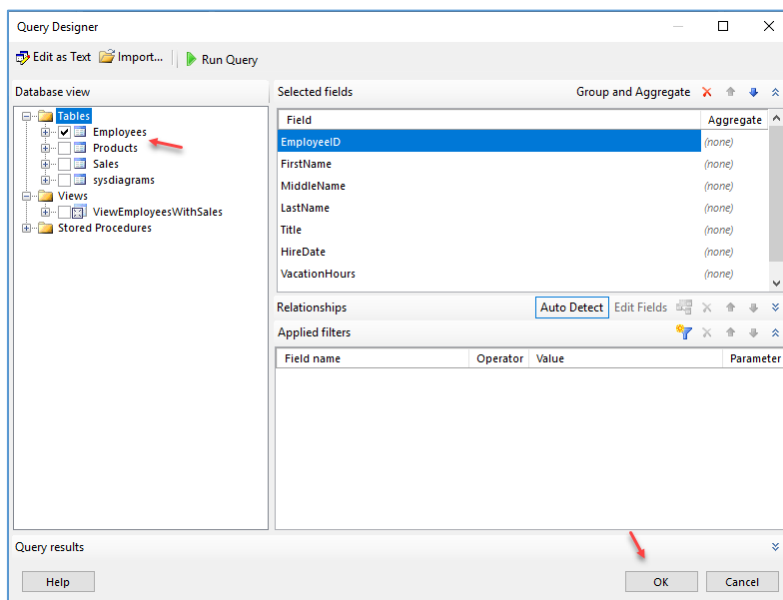
شکل ۱۰-۲۳

در شکل ۱۰-۲۴ باید نام مورد نظر خود را وارد کنید و از قسمت Data Source باید گزینه‌ی DataSource1 را که با هم ایجاد کردیم را انتخاب کنید و برای اینکه جداول آن را در خروجی مشخص کنیم بر روی Query Designer کلیک کنید.



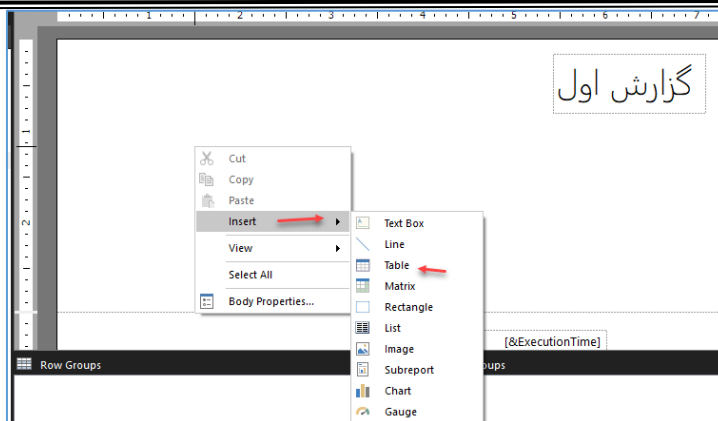
شکل ۱۰-۲۴

در شکل ۱۰-۲۵ باید جدول مورد نظر خود را انتخاب کنید، بعد از انتخاب لیست آن نمایش داده خواهد شد.



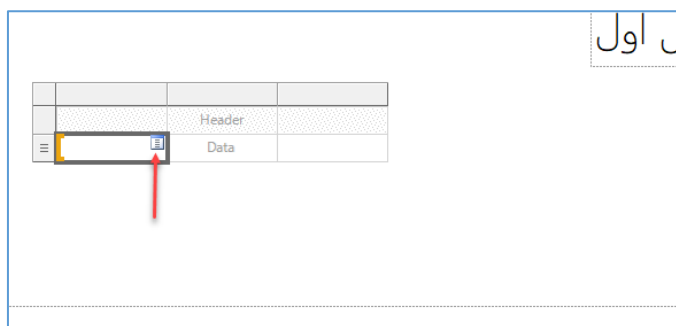
شکل ۱۰-۲۵

در شکل ۱۰-۲۶ می‌خواهیم یک جدول ایجاد کنیم و اطلاعات مورد نظر در جدول Employees را در این جدول قرار دهیم. برای این کار بر روی صفحه مورد نظر کلیک راست کنید و از قسمت Insert گزینه‌ی Table را انتخاب کنید.



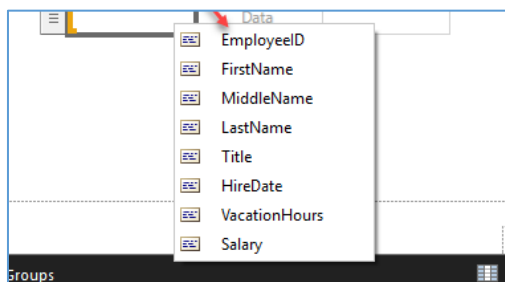
شکل ۱۰-۲۶

در شکل ۱۰-۲۷ در جدول مورد نظر اگر موس را قرار دهید آیکون مورد نظر را مشاهده خواهید کرد که بر روی آن کلیک کنید.



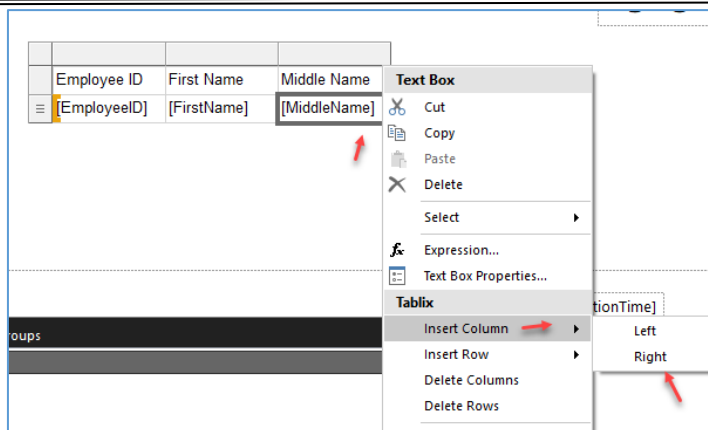
شکل ۱۰-۲۷

در شکل ۱۰-۲۸ باید یکی از ستون‌های موجود را انتخاب کنید تا در خروجی اطلاعات آن نمایش داده شود.



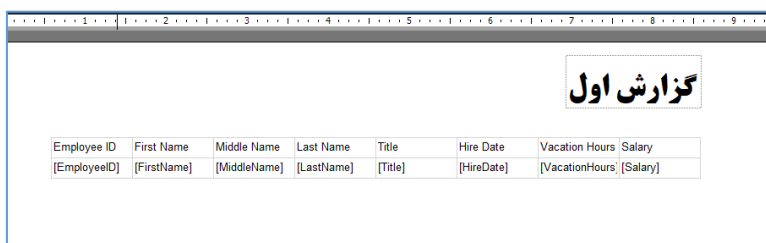
شکل ۱۰-۲۸

در شکل ۱۰-۲۹ بعد از اینکه سه ستون اول را تکمیل کردید، باید چند ستون دیگر هم اضافه کنید، برای این کار بر روی ستون آخر کلیک راست کنید و از قسمت Insert Column گزینه‌ی Right را انتخاب کنید.



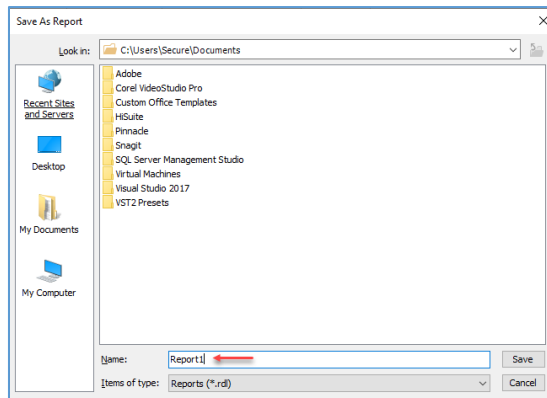
شکل ۱۰-۲۹

در شکل ۱۰-۳۰ تمام ستون‌های مورد نظر به صفحه اضافه شده است، همان‌طور که گفتیم می‌توانید به دلخواه خود صفحه را ویرایش و زیبا کنید.



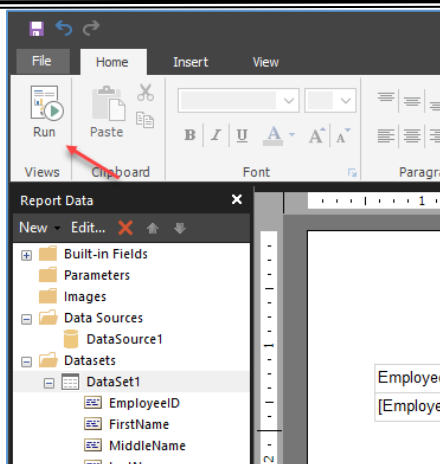
شکل ۱۰-۳۰

با کلیک بر روی دکمه ترکیبی CTRL + S این فرم گزارش را در جای مناسب ذخیره کنید.



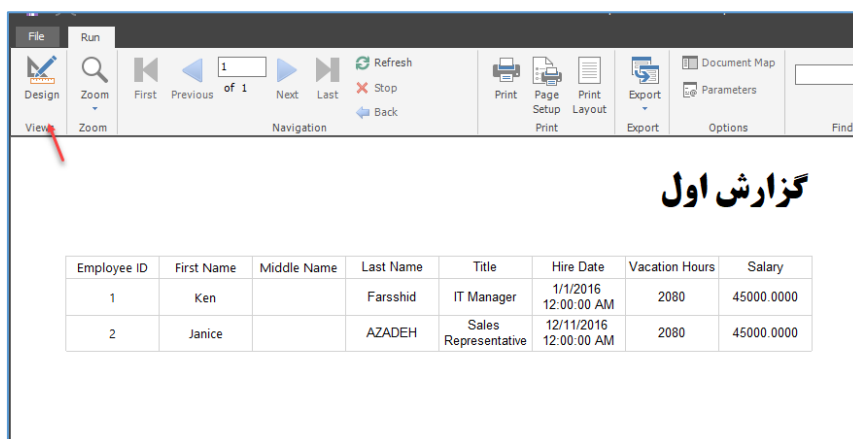
شکل ۱۰-۳۱

اگر بخواهید گزارش مورد نظر را در همین نرم‌افزار Report Builder اجرا کنید باید به‌مانند شکل ۱۰-۳۲ بر روی Run کلیک کنید.



شکل ۱۰-۳۲

همان‌طور که در شکل ۱۰-۳۳ مشاهده می‌کنید نتیجه خروجی جدول مورد نظر است، پس شما توانستید یک فایل گزارش از جدول SQL ایجاد کنید.

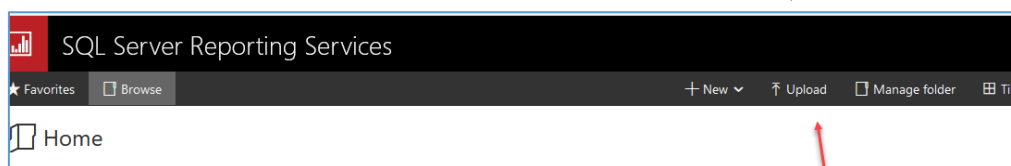


شکل ۱۰-۳۳

خوب این یک مثال ساده از طراحی گزارش در Report Builder است که فقط با همان نرم‌افزار اجرا می‌شود، ولی اگر بخواهیم به صورت تحت وب آن را مشاهده کنیم چه کاری باید انجام دهیم؟ برای این کار باید وارد آدرس زیر شوید و سرویس تحت وب Report مربوط به سرویس SQL Reporting را اجرا کنید.

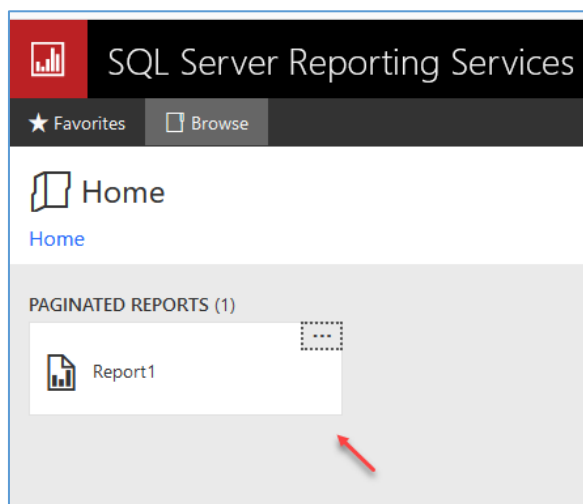
<http://center2/ReportS/browse/>

بعد از بازکردن آدرس مورد نظر باید به‌مانند شکل ۱۰-۳۴ بر روی Upload کلیک کنید و گزارشی را که در قسمت قبل با نام Report1 ذخیره کردیم را انتخاب و آپلود کنید.



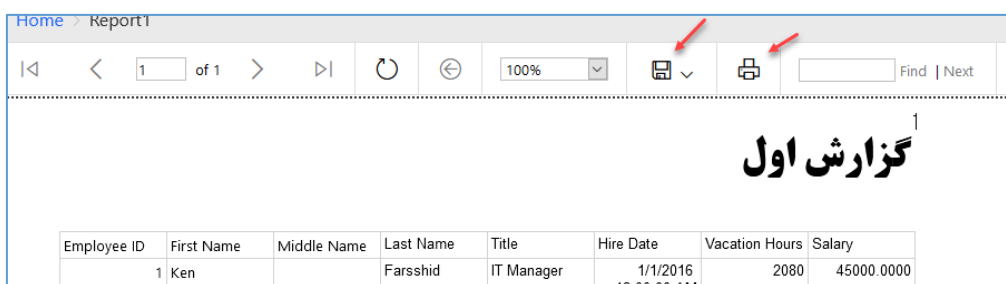
شکل ۱۰-۳۴

همان‌طور که در شکل ۱۰-۳۵ مشاهده می‌کنید فایل Report1 به صفحه اضافه شده است و با کلیک بر روی آن گزارش مورد نظر باز خواهد شد.



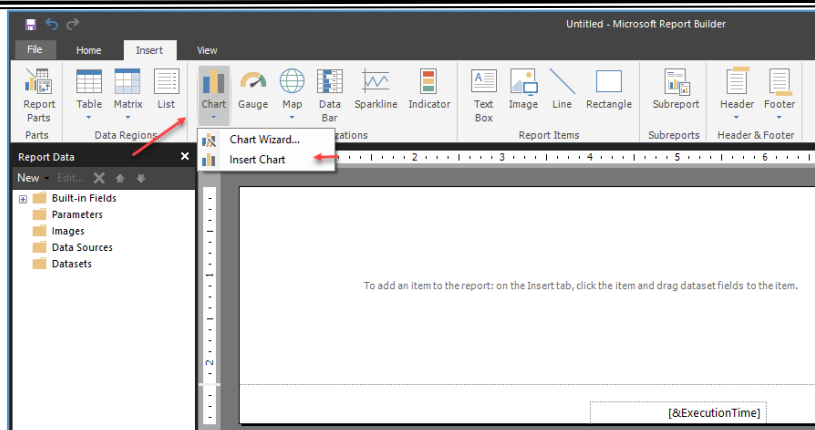
شکل ۱۰-۳۵

همان‌طور که در شکل ۱۰-۳۶ مشاهده می‌کنید گزارش مورد نظر تحت وب اجرا شده است، در این صفحه گزینه‌های مختلفی وجود دارد، مثلاً می‌توانید از گزارش مورد نظر خروجی‌های مختلف تهیه کنید و یا اینکه آن را پرینت بگیرید.



شکل ۱۰-۳۶

شما می‌توانید با استفاده از نرم‌افزار Report Builder گزارش‌های زیبا و متنوعی ایجاد کنید، مانند ایجاد چارت‌ها و نمودارهای مختلف برای نمایش زیبا و دقیق اطلاعات، برای اینکه کلی این موضوع را بیان کنیم باید به‌مانند شکل ۱۰-۳۷ باید وارد Insert شوید و بر روی Chart کلیک کنید که دو گزینه را به شما نشان می‌دهد که اگر بخواهید سریع یک نمودار از محصولات خود ایجاد کنید می‌توانید گزینه‌ی Chart Wizard را انتخاب کنید و یا اگر بخواهید به‌صورت دستی انتخاب کنید می‌توانید گزینه‌ی Insert Chart را انتخاب کنید.



شکل ۱۰-۳۷

خوب بهتر است تا به اینجا، کار با Report Builder را به اتمام برسانیم و به سراغ نرم افزار قدرتمند دیگری از مایکروسافت برویم و آن هم کسی نیست جزء Power BI که به جرات می توان گفت که یکی از قدرتمندترین نرم افزارهای موجود در بازار است.

نصب و راه اندازی Power BI

Power BI بعد از معرفی توسط مایکروسافت تبدیل یک ابزار محبوب در هوش تجاری شده است و کاربران زیادی به علت زیبایی، سهولت و کیفیت از این نرم افزار استفاده می کنند، برای استفاده از این نرم افزار باید لایسنس آن را تعریف کنید، البته لایسنس رایگان آن در اینترنت وجود دارد اما بعضی از دوستان همین لایسنس رایگان را به قیمت گزاف به مشتری می فروشند.



شکل ۱۰-۳۸

مایکروسافت برای گزارش گیری پیشرفته سه ابزار را در گروه Power Platform قرار داده است که شامل Power BI, Power Apps, Microsoft Flow است و هر کدام برای یک کار خاص طراحی شده اند.

برای نصب و راه اندازی این برنامه در ویندوز نیاز به دو نسخه ی آن داریم:

Power BI Reporting Server که از اسم آن مشخص است که مربوط به نسخه سرور می باشد و باید بر روی سرور SQL شما نصب شود، توجه داشته باشید که این نسخه نیاز به لایسنس دارد که باید تهیه کنید.

Power BI Desktop که یک نسخه ی رایگان است که برای متصل شدن به نسخه ی سرور باید آن را بر روی کلاینت خود نصب کنید، البته نسخه ی موبایل آن هم وجود دارد که در صورت نیاز از آن هم استفاده می کنیم.

برای شروع کار باید نسخه ی سرور را بر روی سرور SQL نصب و تنظیمات آن را انجام دهیم، از طریق لینک زیر می توانید نسخه ی سرور به همراه نسخه ی دستکتاپ را دانلود کنید.

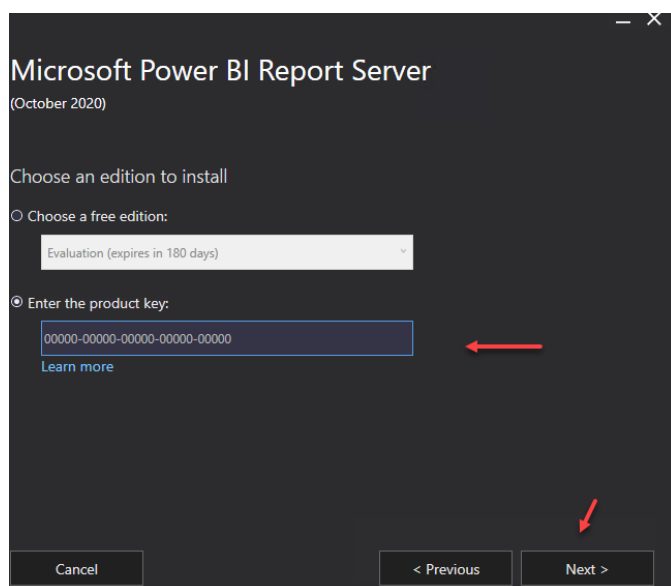
<https://www.microsoft.com/en-us/download/details.aspx?id=57270>

بعد از دانلود فایل‌های مورد نظر، نسخه‌ی سرور آن را در سرور SQL اجرا کنید، به‌مانند شکل ۱۰-۳۹ بر روی **Install** **Power BI Report Server** کلیک کنید.



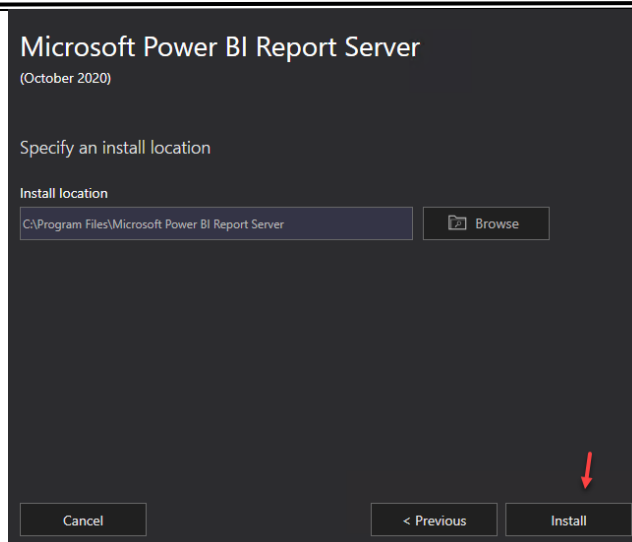
شکل ۱۰-۳۹

در شکل ۱۰-۴۰ باید لایسنس مورد نظر را وارد و بر روی **Next** کلیک کنید.



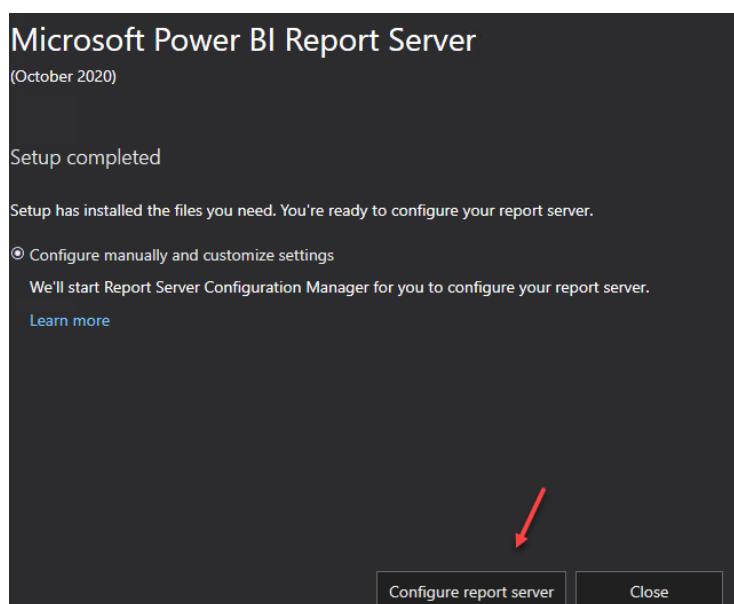
شکل ۱۰-۴۰

در شکل ۱۰-۴۱ باید بر روی **Install** کلیک کنید تا کار نصب آغاز شود.



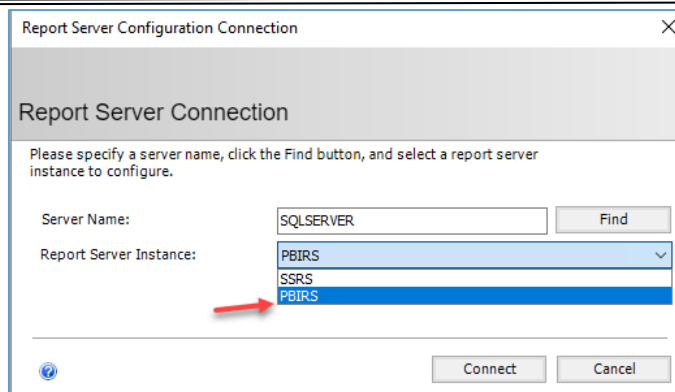
شکل ۱۰-۴۱

بعد از نصب نرم افزار باید به مانند شکل ۱۰-۴۲ بر روی Configure Report Server کلیک کنید تا سرویس مورد نظر اجرا شود.



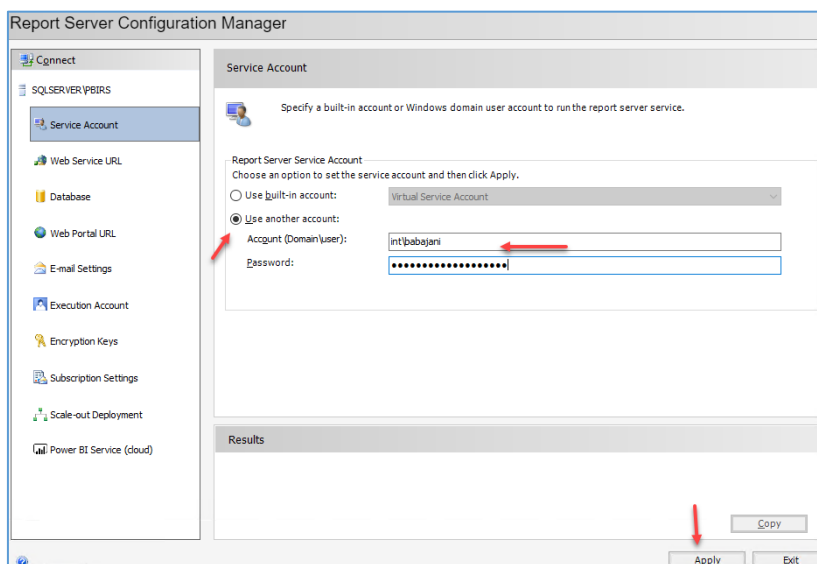
شکل ۱۰-۴۲

در شکل ۱۰-۴۳ در قسمت Server name نام سرور خود را وارد و در قسمت Instance باید PBIRS را انتخاب کنید که بعد از نصب نرم افزار Power BI ایجاد شده است و باید تنظیمات را بر روی آن انجام دهیم.



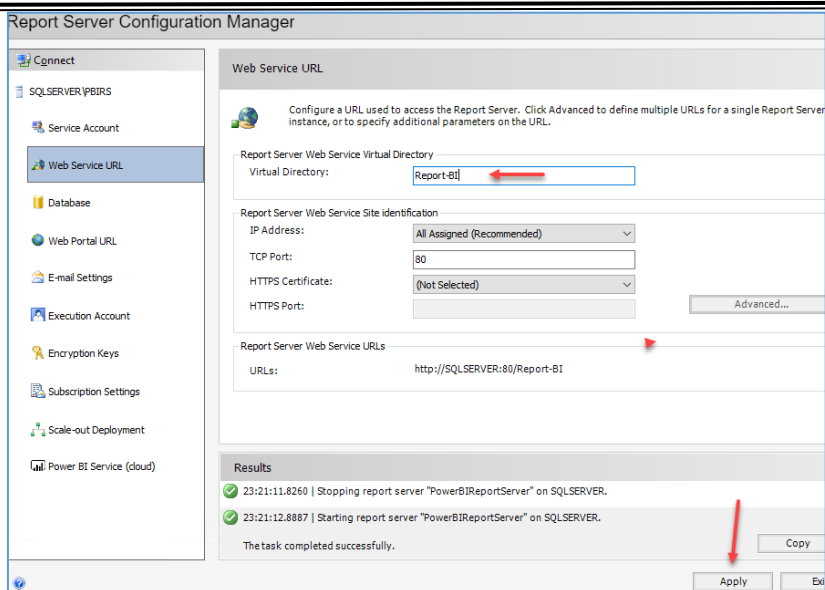
شکل ۱۰-۴۳

در شکل ۱۰-۴۴ باید نام کاربری مورد نظر را برای دسترسی به سرویس Power BI وارد و بر روی APPLY کلیک کنید تا تغییرات مورد نظر اعمال شود.



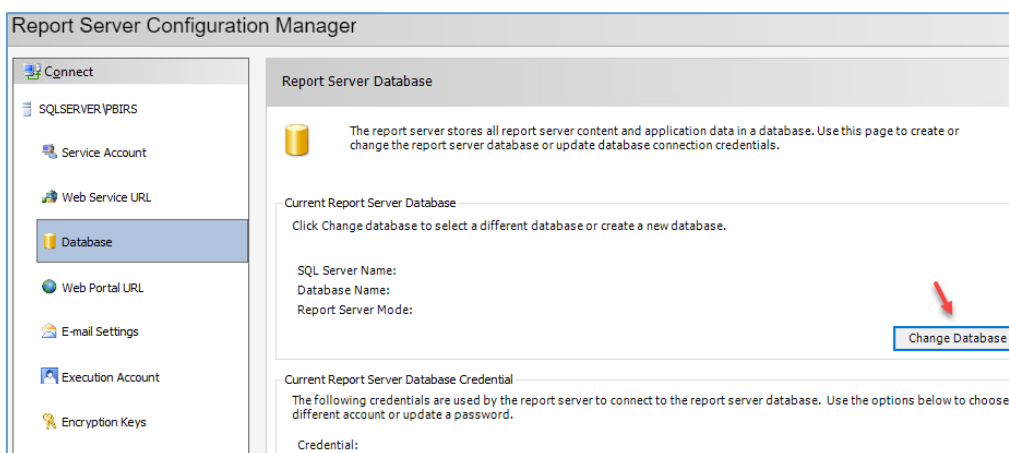
شکل ۱۰-۴۴

در شکل ۱۰-۴۵ باید یک نام جدید در قسمت Virtual Directory وارد کنید تا با نام قبلی که در سرویس Reporting تنظیم کردیم برخورد نداشته باشد.



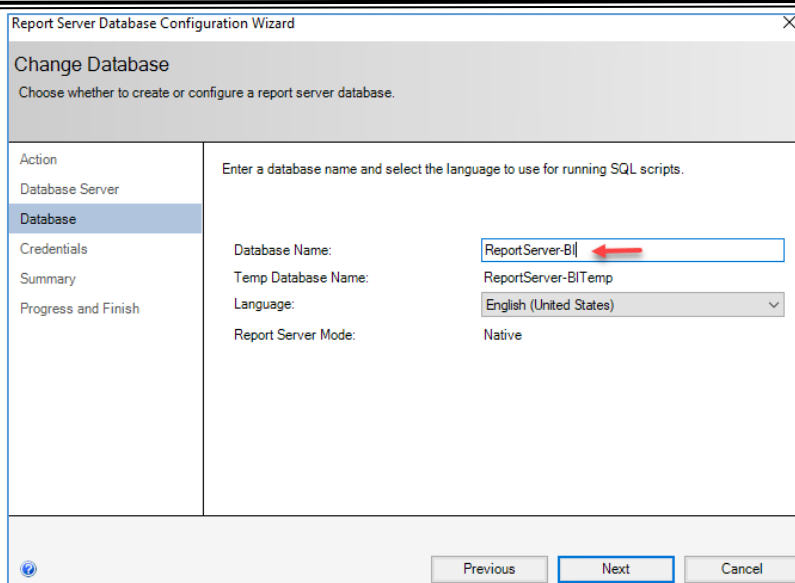
شکل ۴۶-۱۰

در شکل ۴۷-۱۰ باید یک دیتابیس جدید برای سرویس Power BI ایجاد کنید برای انجام این کار بر روی Change Database کلیک کنید.



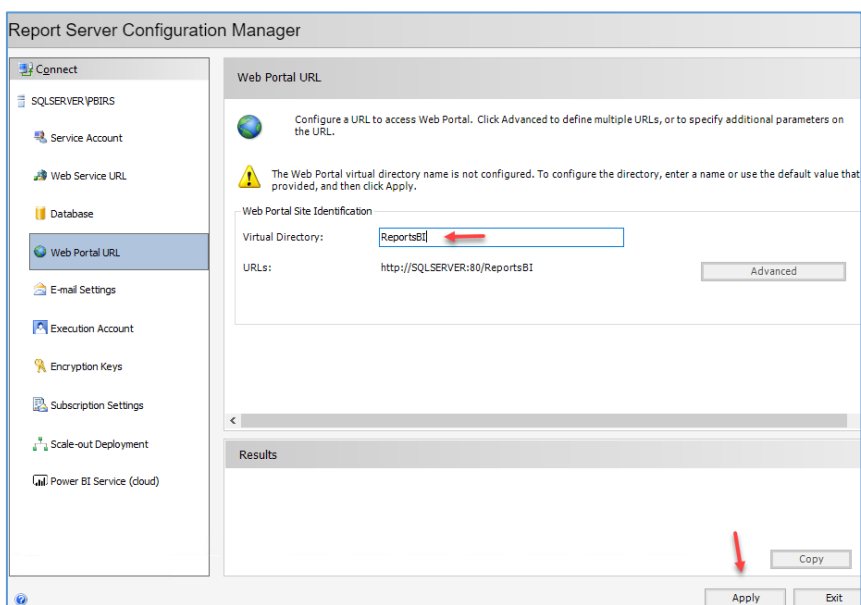
شکل ۴۷-۱۰

در شکل ۴۸-۱۰ باید نام دیتابیس را وارد کنید، اگر نام را تغییر ندهید، این دیتابیس با دیتابیس قبلی که برای ReportService ایجاد کردیم برخورد می‌کند و با مشکلاتی روبرو می‌شود که بهتر است نام آن را تغییر دهیم، بر روی Next کلیک کنید تا دیتابیس مورد نظر ایجاد شود.



شکل ۱۰-۴۸

در شکل ۱۰-۴۹ باید یک نام جدید ایجاد کنید تا با نام قبلی مشکلی پیش نیاید، بعد از وارد کردن نام بر روی APPLY کلیک کنید تا تنظیمات جدید اعمال شود و در آخر بر روی Exit کلیک کنید تا پنجره بسته شود.

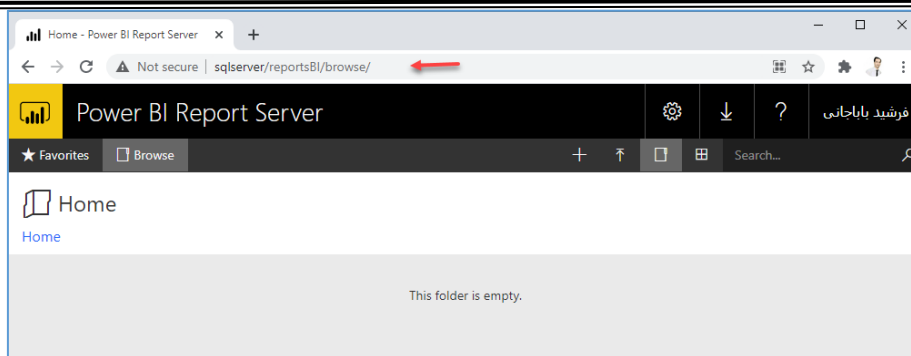


شکل ۱۰-۴۹

در ادامه برای اینکه سرویس POWER BI را تحت وب مشاهده کنید باید از آدرس زیر استفاده کنید.

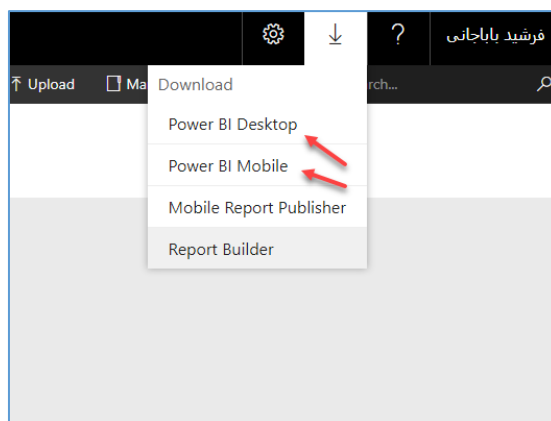
<http://sqlserver/reportsBI/browse/>

در آدرس بالا باید به جای sqlserver نام و یا IP سرور را وارد کنید، همانطور که در شکل ۱۰-۵۰ مشاهده می کنید صفحه Reporting Power BI اجرا شده است و می توانیم عملیات خود را بر روی آن آغاز کنیم.



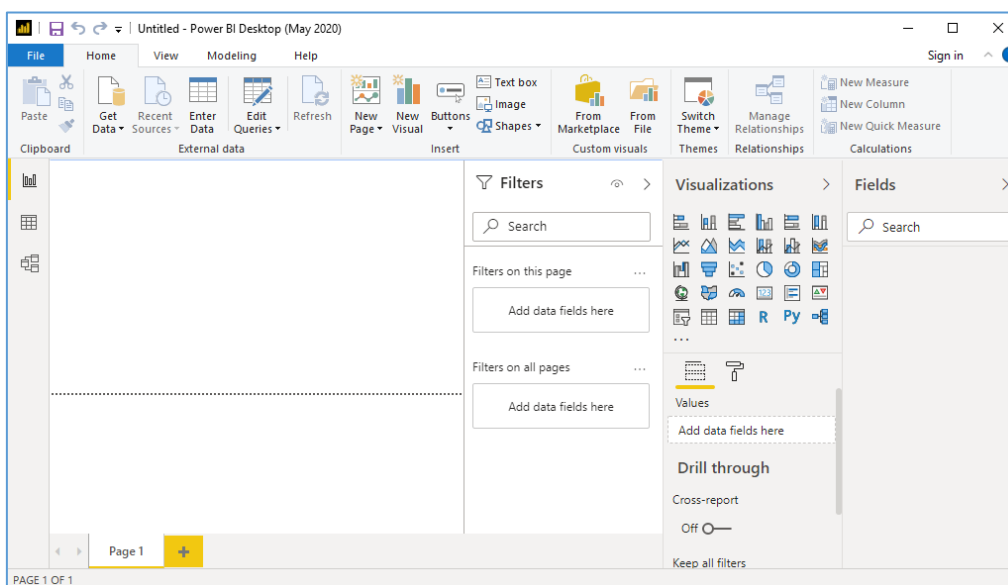
شکل ۱۰-۵۰

در شکل ۱۰-۵۱ با کلید بر روی آیکن مورد نظر می‌توانید نرم‌افزارهایی که قرار است با این سرور کار کنند را مشاهده می‌کنید، برای اینکه بتوانیم بر روی Desktop کار کنیم نسخه‌ی Power BI Desktop را راه‌اندازی می‌کنیم.



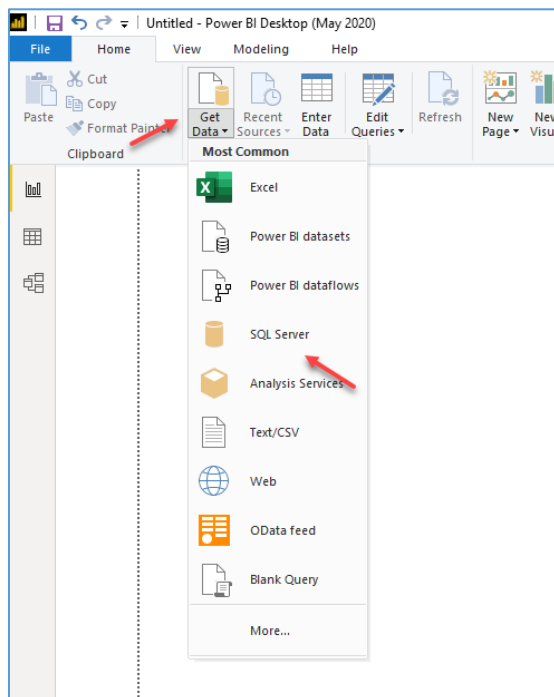
شکل ۱۰-۵۱

همان‌طور که در شکل ۱۰-۵۲ مشاهده می‌کنید، نرم‌افزار Power BI Desktop اجرا شده و کارهای مورد نظر خود را می‌توانید بر روی آن انجام دهید.



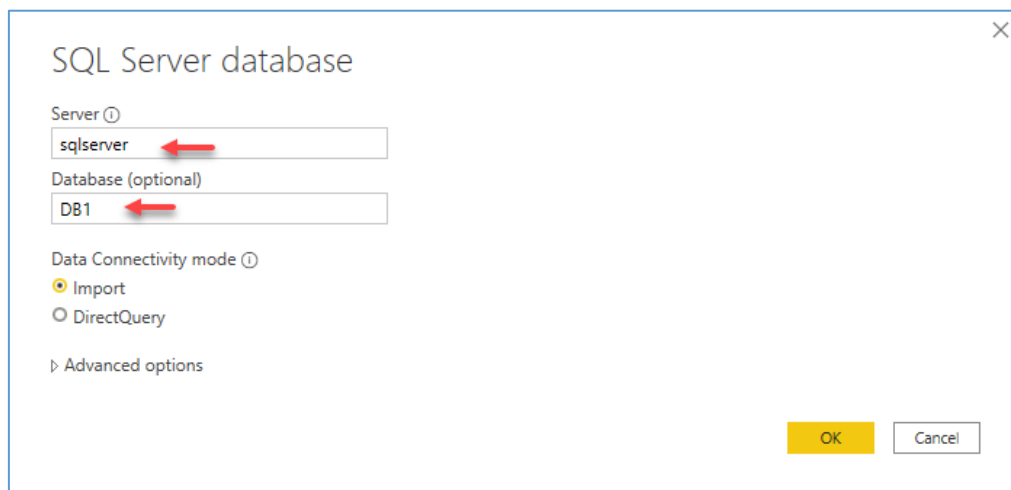
شکل ۱۰-۵۲

برای شروع کار و برای اینکه به دیتابیس SQL متصل شویم و اطلاعات موجود در جداول را بارگذاری کنیم باید از تب Home بر روی Get Data کلیک کنید و گزینه‌ی SQL Server را انتخاب کنید.



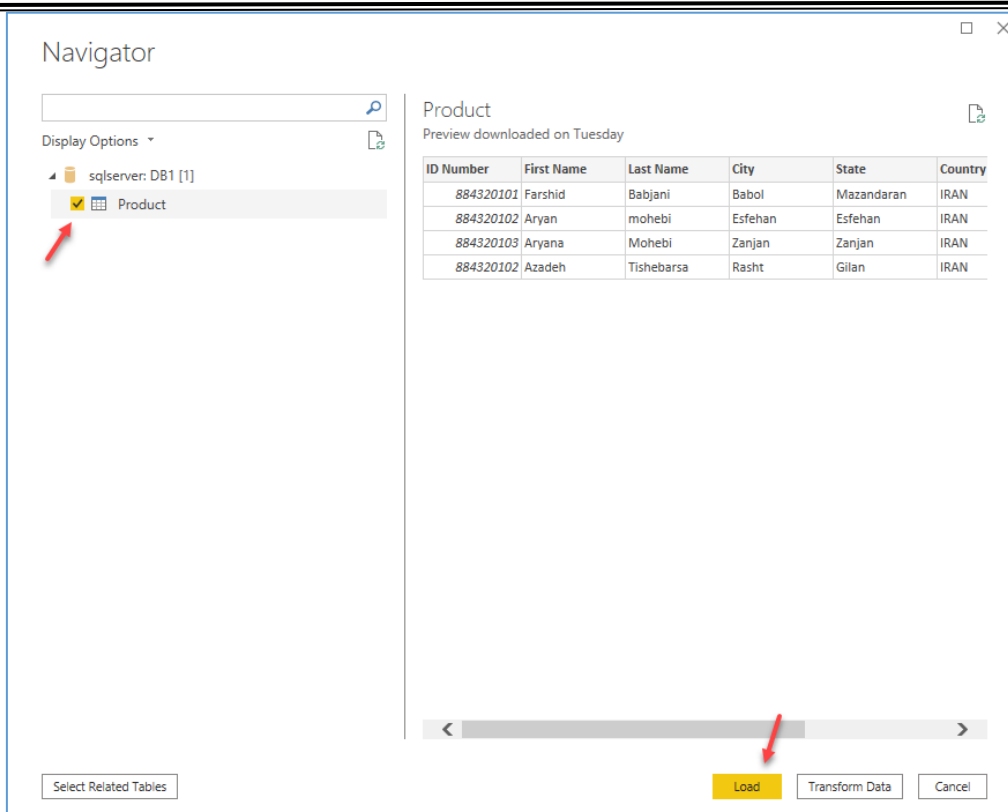
شکل ۱۰-۵۳

در شکل ۱۰-۵۴ باید نام سرور SQL را به همراه نام دیتابیس آن وارد و بر روی OK کلیک کنید.



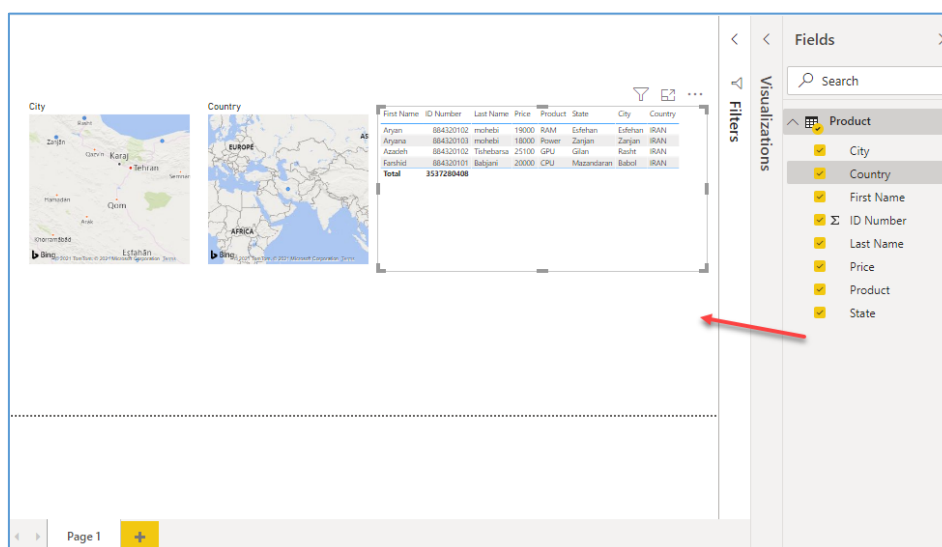
شکل ۱۰-۵۴

به‌مانند شکل ۱۰-۵۵ باید جدول مورد نظر خود را که در دیتابیس وجود دارد انتخاب کنید و بعد بر روی Load کلیک کنید.



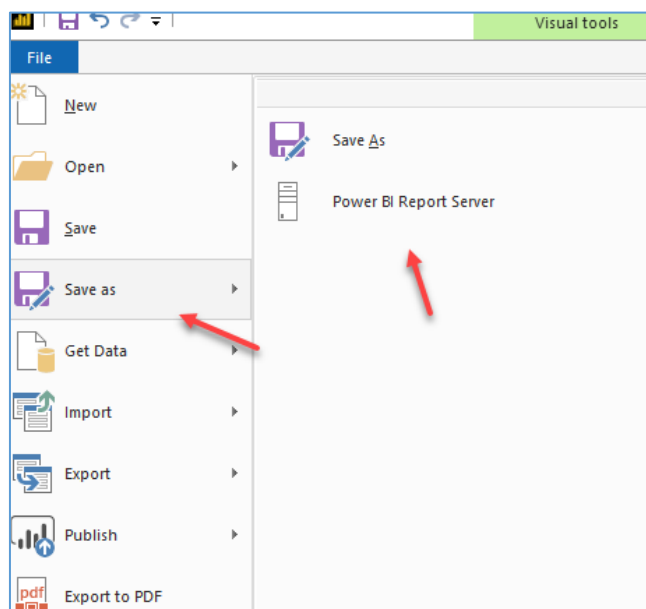
شکل ۱۰-۵۵

بعد از اضافه کردن جدول مورد نظر، حالا می‌توانید از قسمت Product ستون‌های مورد نظر خود را کشیده و در Reports رها کنید، همان‌طور که در شکل ۱۰-۵۶ مشاهده می‌کنید به محض رها کردن فیلد مورد نظر اطلاعات آن در صفحه به صورت آنلاین نمایش داده خواهد شد، مثلاً اگر شهر را در صفحه قرار دهید نقشه و موقعیت آن شهر مشخص خواهد شد که این یک قابلیت جالب خواهد بود.



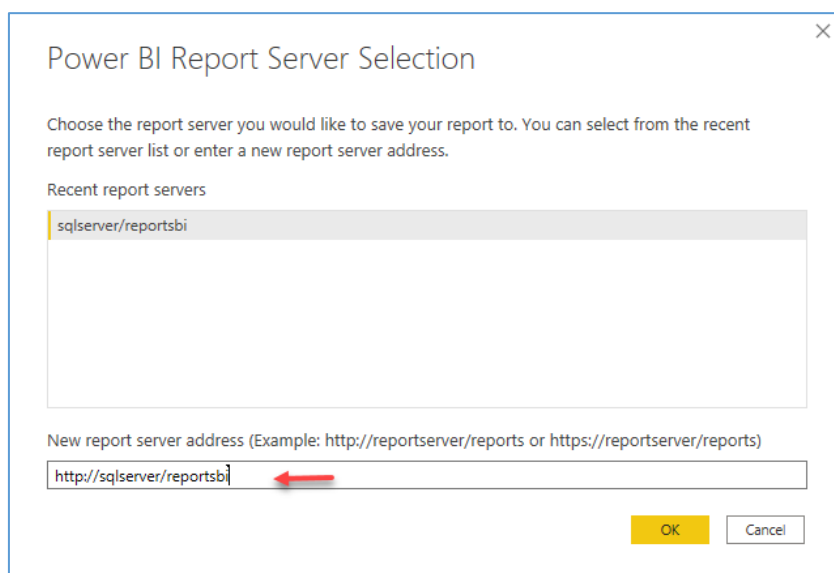
شکل ۱۰-۵۶

خوب اگر بخواهیم این فرم ساده را در Reporting Service Power BI و تحت وب نمایش دهیم می‌توانیم به صورت شکل ۱۰-۵۷ از منوی File و از قسمت Save as گزینه‌ی مورد نظر خود را انتخاب کنید، اگر بخواهید گزارش خود را اول در سیستم خود ذخیره و بعد آن را در Reporting Service تحت وب آپلود کنید می‌توانید گزینه‌ی اول را انتخاب کنید و یا اگر بخواهید به صورت مستقیم آن را در سرور Reporting آپلود کنید باید بر روی Power BI Report Server کلیک کنید.



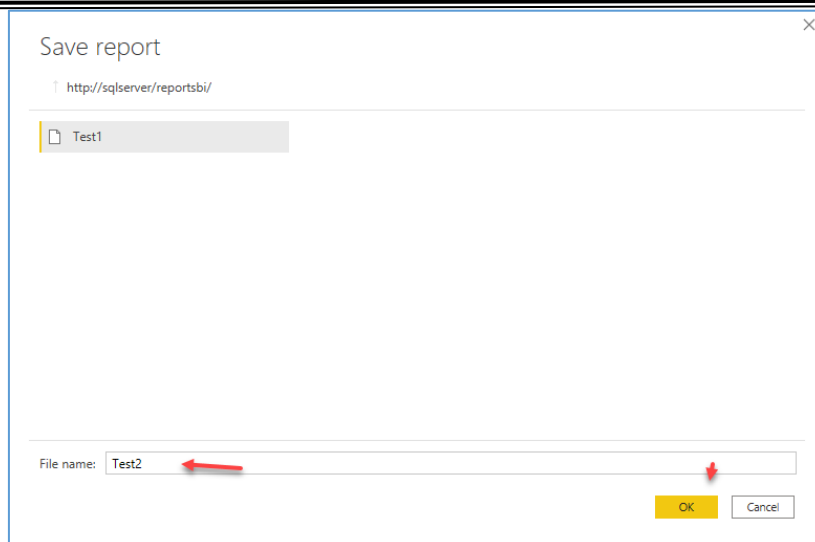
شکل ۱۰-۵۷

در شکل ۱۰-۵۸ باید آدرس سرور Reporting را که در قسمت‌های قبلی و در شکل ۱۰-۴۹ وارد کردیم را در این قسمت وارد و بر روی OK کلیک کنید.



شکل ۱۰-۵۸

در شکل ۱۰-۵۹ باید نام گزارش خود را وارد و بر روی OK کلیک کنید.

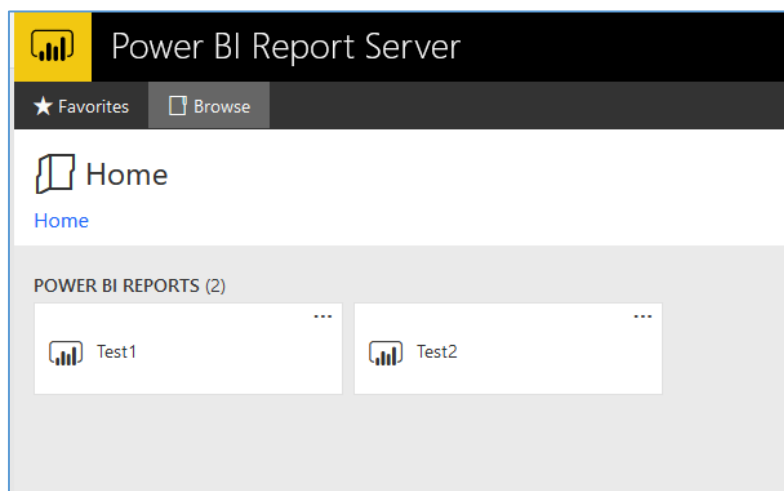


شکل ۱۰-۵۹

در ادامه کار وارد آدرس زیر شوید:

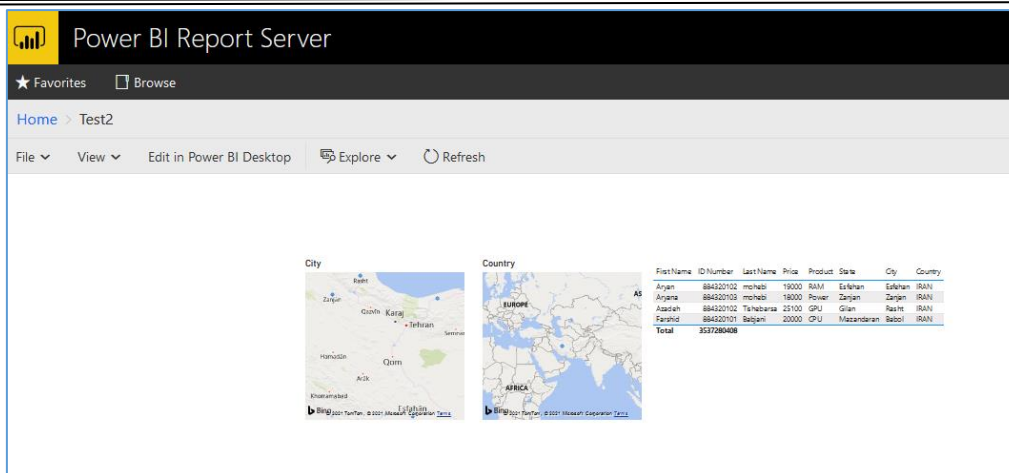
<http://sqlserver/reportsBI/browse/>

همان‌طور که در شکل ۱۰-۶۰ مشاهده می‌کنید گزارش مورد نظر به‌درستی ایجاد و بر روی تست آن کلیک کنید.



شکل ۱۰-۶۰

همان‌طور که در شکل ۱۰-۶۱ مشاهده می‌کنید گزارش مورد نظر به‌درستی اجرا شده است.



شکل ۱۰-۶

خوب برای اینکه بهتر با نحوه‌ی گزارش‌گیری توسط Power BI آشنا شوید می‌خواهیم در SQL دو جدول به‌مانند جداول یک و دو ایجاد کنیم و از طریق Power BI از آنها خروجی تهیه کنیم.

جدول یک - CLASS

Column Name	Data Type	Allow Nulls
[Class Name]	nchar(10)	<input checked="" type="checkbox"/>
[Class ID]	int	<input checked="" type="checkbox"/>
[Number Students]	nchar(10)	<input checked="" type="checkbox"/>
[Satart Date]	date	<input checked="" type="checkbox"/>
[End Date]	date	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

جدول دو - Students

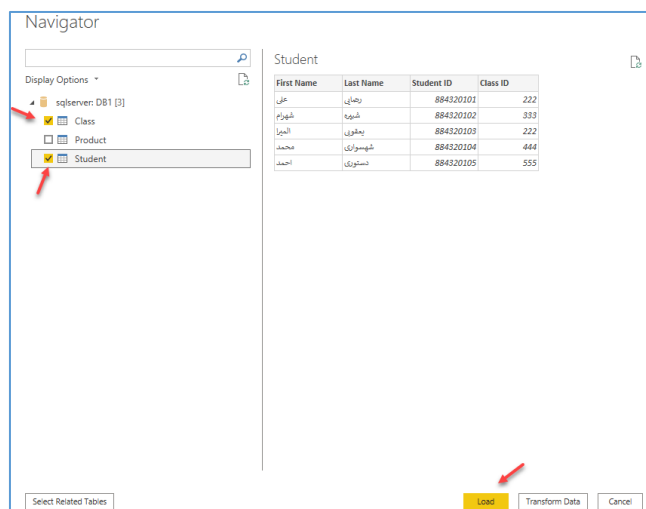
Column Name	Data Type	Allow Nulls
[First Name]	nchar(10)	<input checked="" type="checkbox"/>
[Last Name]	nchar(10)	<input checked="" type="checkbox"/>
[Student ID]	int	<input checked="" type="checkbox"/>
[Class ID]	int	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

در دو جدول بالا یک سری درس به همراه یک سری دانش‌آموزش تعریف شده است که قرار است اطلاعات خروجی آنها را به هم متصل کنیم، همان‌طور که در جدول زیر مشاهده می‌کنید اطلاعات دانش‌آموزان به همراه کلاس درس آنها وارد شده است و حالا می‌خواهیم از طریق نرم‌افزار Power BI از آنها خروجی تهیه کنیم.

	First Name	Last Name	Student ID	Class ID
	علی	رضایی	884320101	222
	شهرام	شبیرو	884320102	333
	المیرا	یعقوبی	884320103	222
	محمد	شهسواری	884320104	444
	احمد	دستوری	884320105	555
▼	NULL	NULL	NULL	NULL

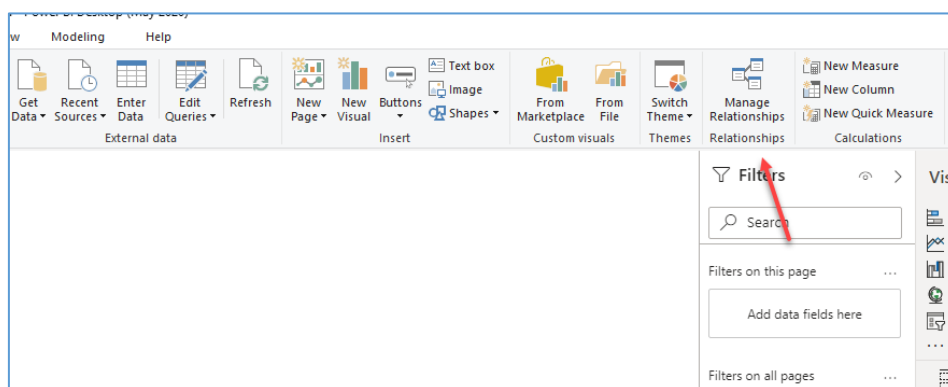
	Class Name	Class ID	Number Stude...	Satart Date	End Date
▶	ادبیات	222	NULL	1400-05-01	1400-05-30
	علوم	333	NULL	1400-05-02	1400-05-30
	ریاضی	444	NULL	1400-05-05	1400-05-30
	زبان	555	NULL	1400-05-09	1400-05-30
*	NULL	NULL	NULL	NULL	NULL

طبق مراحل قبلی نرم‌افزار Power BI را باز کرده و دو جدول جدید را به‌مانند شکل ۶۲-۱۰ به لیست اضافه کنید.



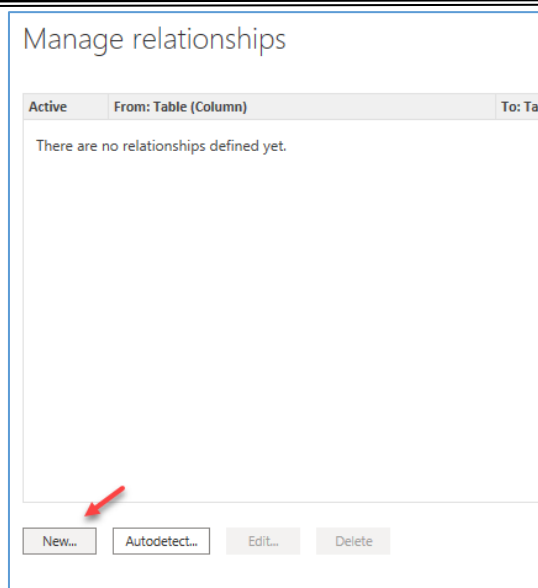
شکل ۶۲-۱۰

به‌مانند شکل ۶۳-۱۰ وارد تب Home شوید و بر روی Manage Relationships کلیک کنید.



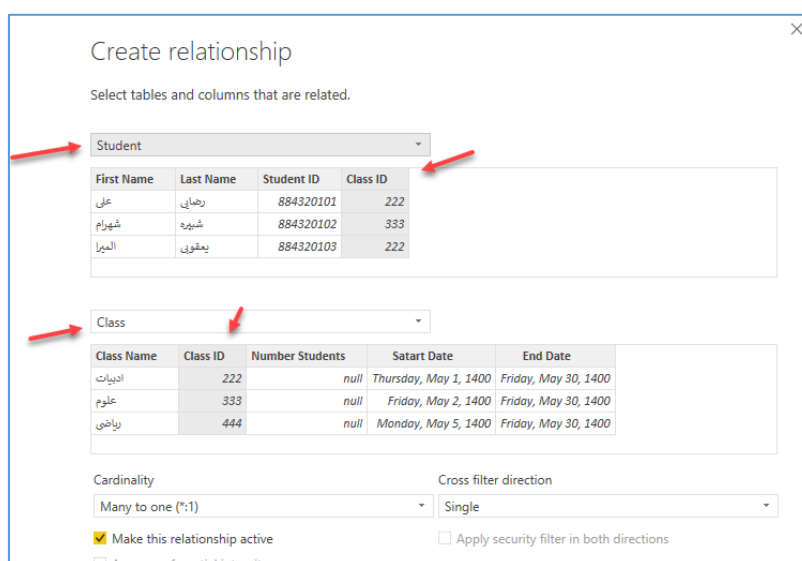
شکل ۶۳-۱۰

در شکل ۶۴-۱۰ باید یک ارتباط بین دو جدول مورد نظر ایجاد کنید که در این دو جدول Class ID توانایی ارتباط درست با هم را دارند، برای همین موضوع بر روی New کلیک کنید.



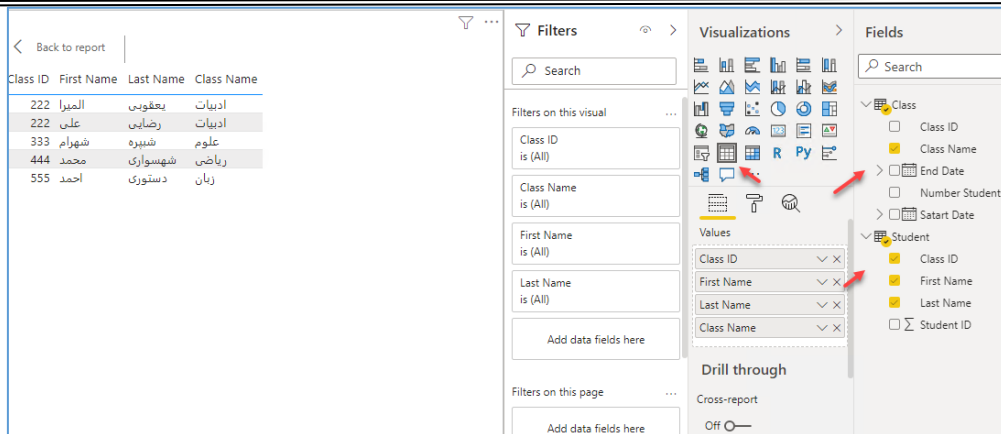
شکل ۱۰-۶۴

در شکل ۱۰-۶۵ و در قسمت اول باید جدولی که قرار است در خروجی نمایش داده شود را انتخاب کنید که بعد از انتخاب ستون Class ID که در هر دو جدول یکی است به عنوان پل ارتباطی این دو جدول به صورت اتوماتیک انتخاب می‌شود.



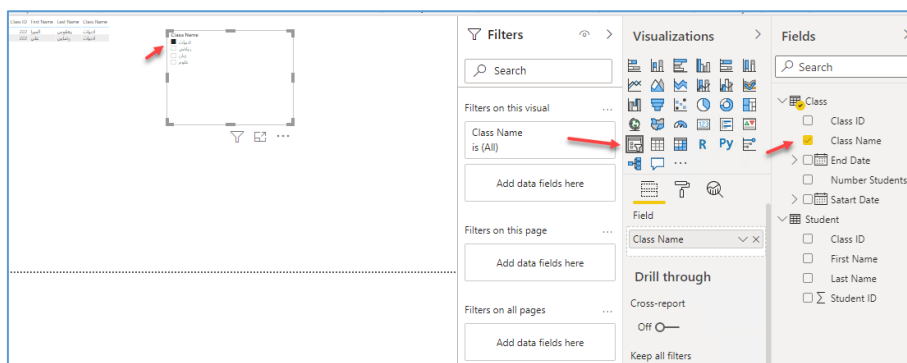
شکل ۱۰-۶۵

در شکل ۱۰-۶۶ می‌توانید از قسمت Visualization یک جدول را انتخاب کنید و بعد از انتخاب باید تیک گزینه‌هایی که قرار است در خروجی چاپ شوند را مشخص کنید که نتیجه آن را مشاهده می‌کنید.



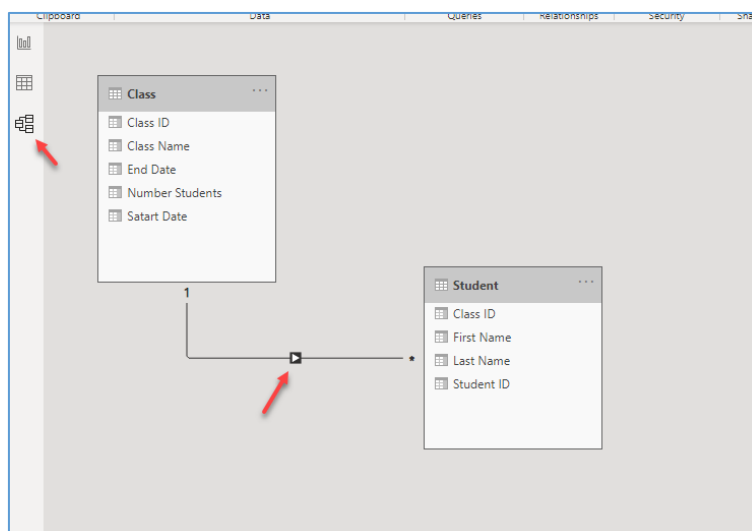
شکل ۱۰-۶۶

اگر چنانچه بخواهید کمی کار را زیباتر کنید می‌توانید به‌مانند شکل ۱۰-۶۷ ابزار فیلتر قرار دهید تا کاربر با انتخاب هر کلاس بتواند لیست دانشجویان آن را مشاهده کند.



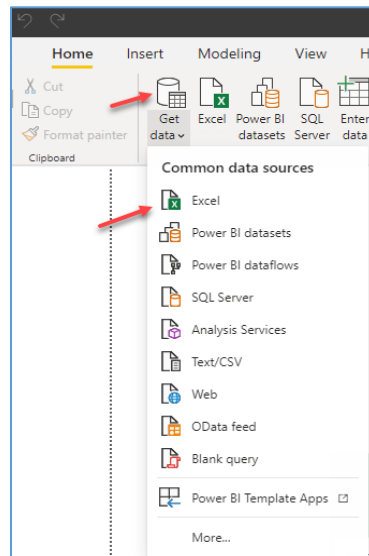
شکل ۱۰-۶۷

برای اینکه ارتباط بین دو جدول را به‌صورت بهتر مشاهده کنید می‌توانید به‌مانند شکل ۱۰-۶۸ وارد قسمت Model شوید و در صفحه‌ی باز شده اگر نشانگر ماوس را بر روی فلش مورد نظر قرار دهید با رنگ مشخص شده ارتباط بین ستون‌ها در هر دو جدول را مشاهده خواهید کرد.



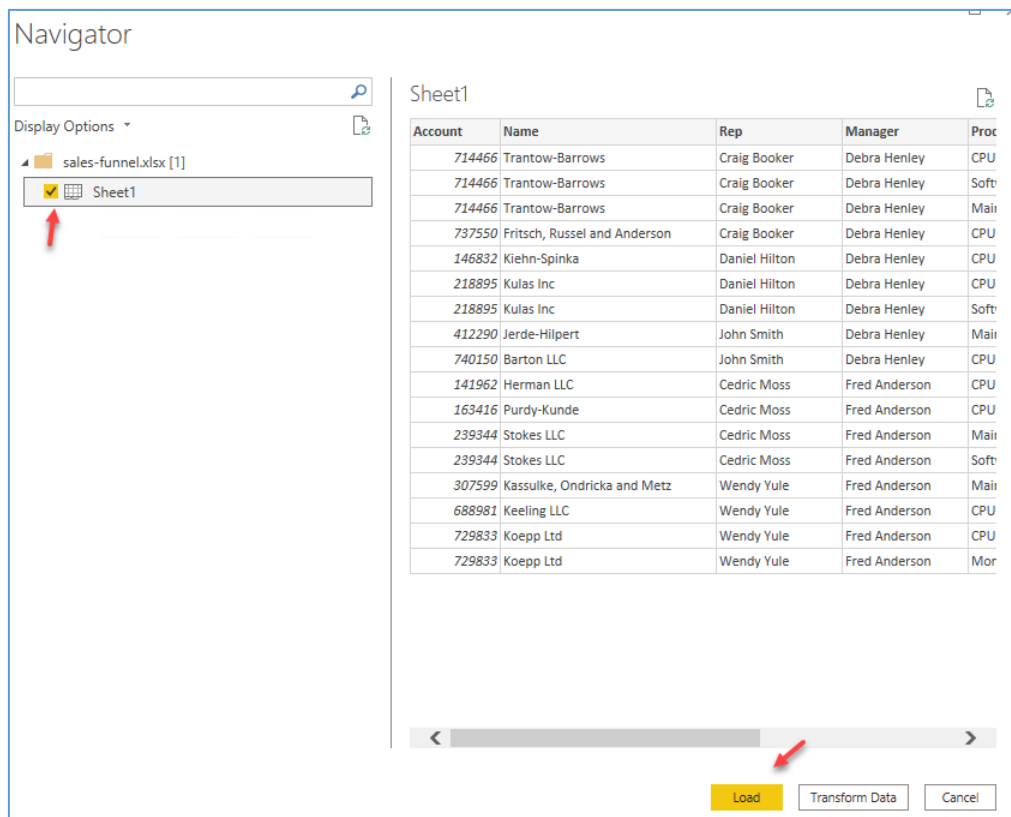
شکل ۱۰-۶۸

شما می‌توانید دیتابیس‌های دیگر مانند Excel, Oracle و ... را هم وارد Power BI Desktop کنید و از آنها یک خروجی مفید تهیه کنید، برای انجام این کار به‌مانند شکل ۶۹-۱۰ می‌توانید گزینه‌ی Excel را انتخاب کنید.



شکل ۶۹-۱۰

همان‌طور که در شکل ۷۰-۱۰ مشاهده می‌کنید فایل Excel مورد نظر انتخاب شده و شما می‌توانید هر کدام از Sheet‌های آن را انتخاب کنید.



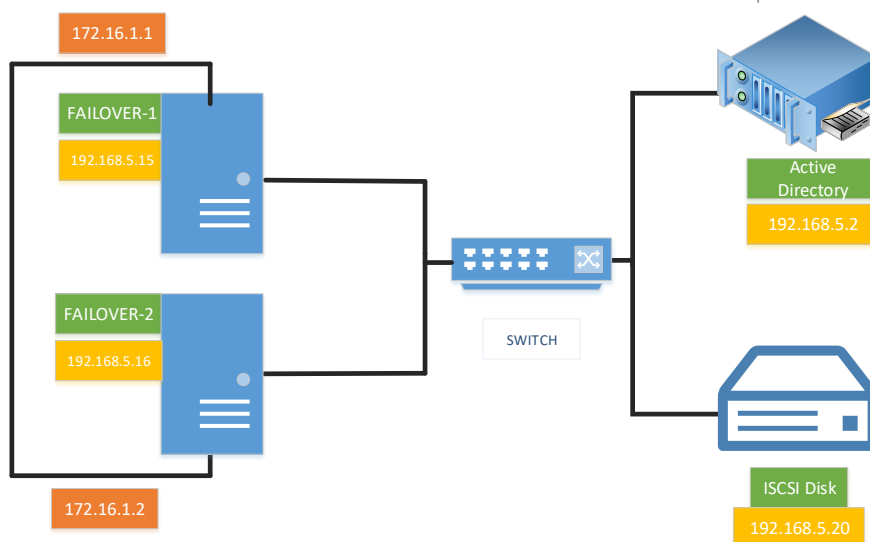
شکل ۷۰-۱۰

فصل هشتم

SQL Failover Clustering

زمانی که صحبت از SQL Server است، یعنی اینکه دسترسی به داده‌های آن در یک لحظه هم نباید با اختلال روبرو شود و برای حل این مشکل باید یک راه‌کار پیاده‌سازی کرد تا سرویس‌های SQL جتی برای ثانیه‌ای از کار نیفتن، برای حل این مشکل مایکروسافت سرویسی با نام Failover Clustering را معرفی کرد که در این سرویس چند سرور به‌عنوان Node در گروهی قرار دارند و چنانچه برای یکی از سرورها مشکلی ایجاد شود سرورهای دیگری که در آن گروه قرار دارند جایگزین آن می‌شوند.

برای انجام این کار نیاز به چند سرور به‌مانند شکل ۱-۱۱ داریم، در این سناریو نیاز به یک سرور Active Directory، یکی سرور ISCSI برای ایجاد دیسک ISCSI و دسترسی از طریق شبکه، و دو سرور FAILOVER که باید نقش FAILOVER را بازی کنند که البته این چند سرور ا طریق سوئیچ به هم متصل می‌شوند و دو سرور FAILOVER باید یک کارت شبکه‌ی دیگر هم داشته باشند.



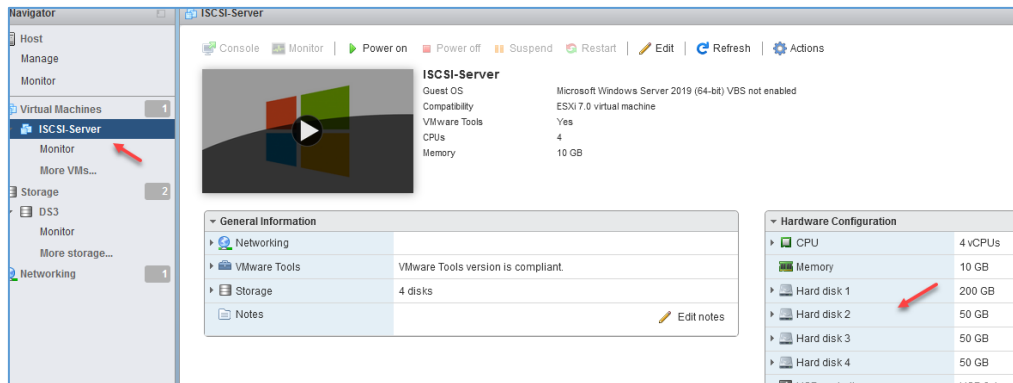
شکل ۱-۱۱

تنظیمات سرور iSCSI

برای شروع کار باید سرور ISCSI را آماده کنیم، برای ایجاد این سرور شما می‌توانید از نرم‌افزارهای رایگان مانند FreeNAS و... استفاده کنید و دیسک‌های تحت شبکه با پروتکل ISCSI را ایجاد کنید، ولی در این آموزش ما از یک ویندوز سرور ۲۰۱۹ استفاده می‌کنیم.

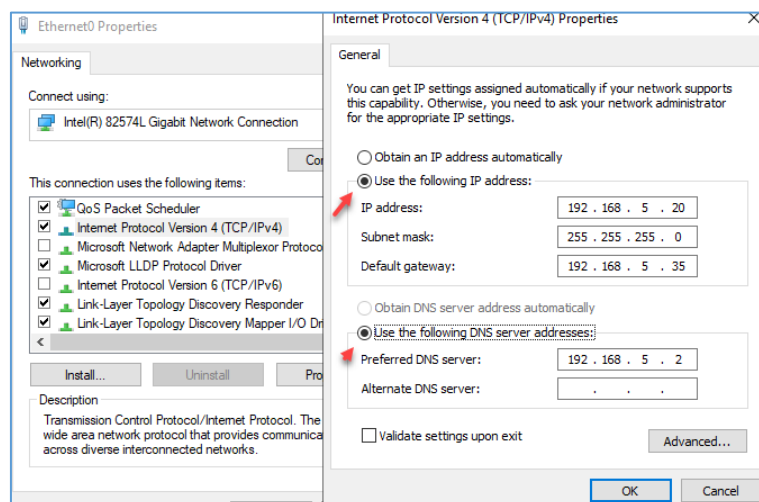
برای شروع کار و ایجاد سرور، پیشنهاد می‌شود که از سرورهای مجازی برای این کار استفاده کنید، تا نحوه ایجاد و مدیریت آن آسان باشد، برای همین ما از یک سرور ESXi استفاده کردیم و یک ماشین مجازی به‌مانند شکل ۲-۱۱ یک ماشین مجازی با نام ISCSI-Server ایجاد کردیم و به آن چهار هارد اضافه کردیم، برای کار Failover Clustering

نیاز به چهار هارد دیسک از نوع ISCSI داریم تا بتوانیم از آن در این سناریو و این سرویس استفاده کنیم، بعد از ایجاد ماشین مورد نظر بر روی آن ویندوز سرور ۲۰۱۹ نصب می‌کنیم.



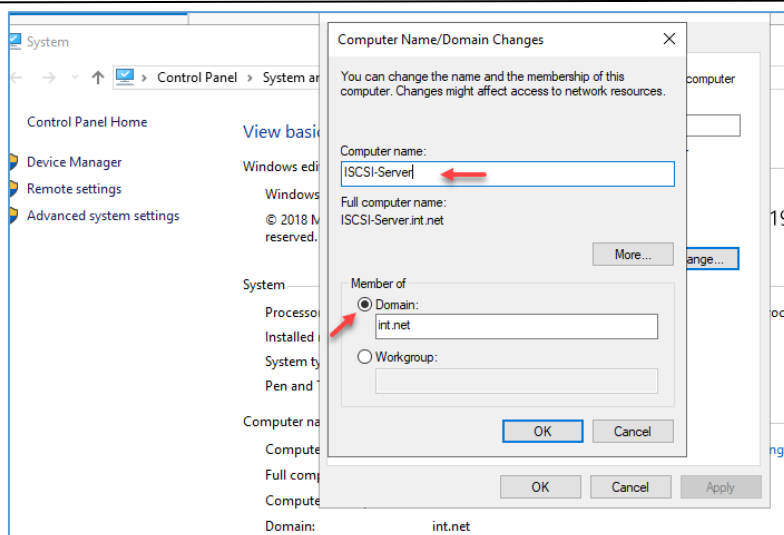
شکل ۱۱-۳

بعد از اجرا شدن ویندوز مورد نظر اولین کاری که انجام می‌دهید این است که آدرس IP را به صورت دستی وارد کنید و در قسمت نام سرور هم یک نام برای آن وارد و آن را عضو دومین کنید. به‌مانند شکل ۱۱-۳ IP سرور مورد نظر را وارد و بر روی OK کلیک کنید، البته می‌توانستید از سرویس DHCP هم برای این کار استفاده کنید.



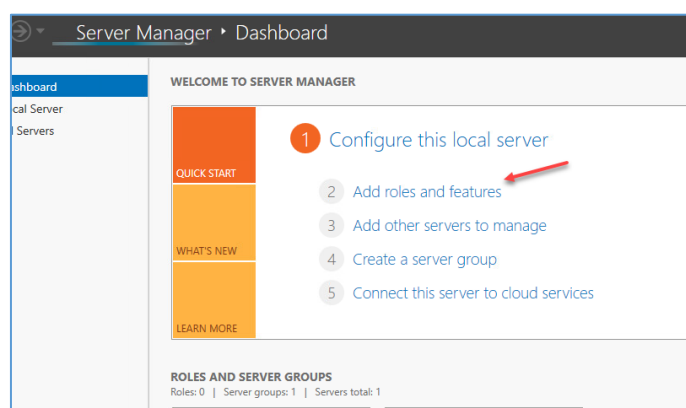
شکل ۱۱-۳

در شکل ۱۱-۴ باید نام سرور را وارد و آن را عضو دومین کنید.



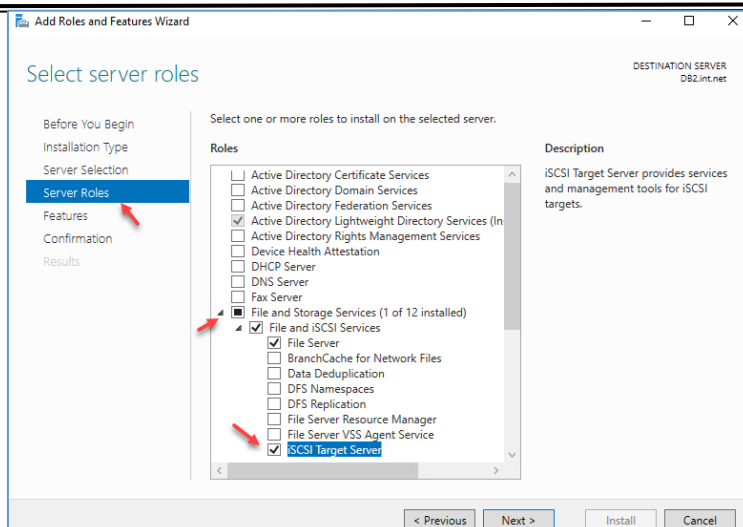
شکل ۱۱-۴

بعد از اینکه مراحل بالا را انجام دادید باید سرویس ISCSI را بر روی سرور نصب کنیم، برای این کار به‌مانند شکل ۱۱-۵ وارد Server Manager شوید و بر روی Add Roles and features کلیک کنید.



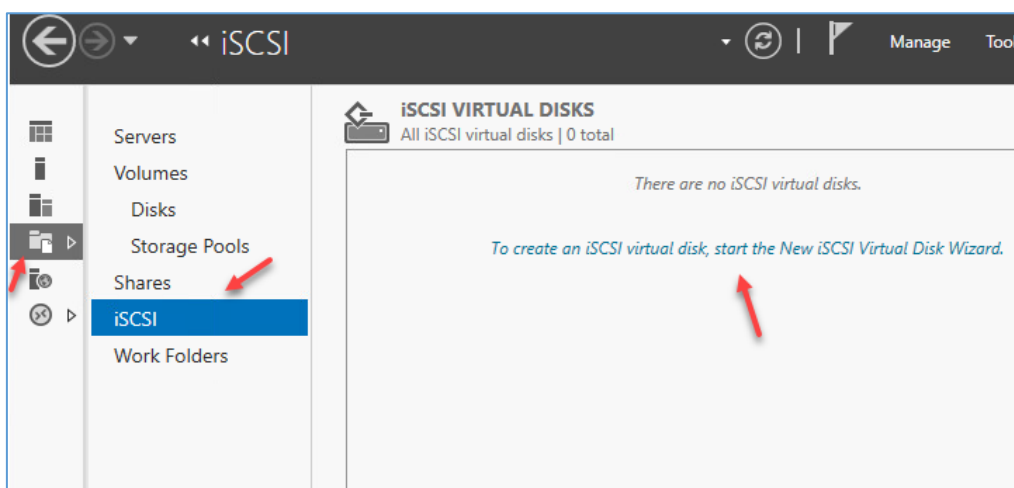
شکل ۱۱-۵

به‌مانند شکل ۱۱-۶ وارد Server Roles شوید و از زیر مجموعه‌ی File and ISCSI Services تیک گزینه‌ی ISCSI Target Server را انتخاب و بر روی Next کلیک کنید و در ادامه بر روی Install کلیک کنید، فقط توجه داشته باشید در قسمت آخر تیک گزینه‌ی Restart را انتخاب کنید تا سرویس بعد از نصب Restart شود.



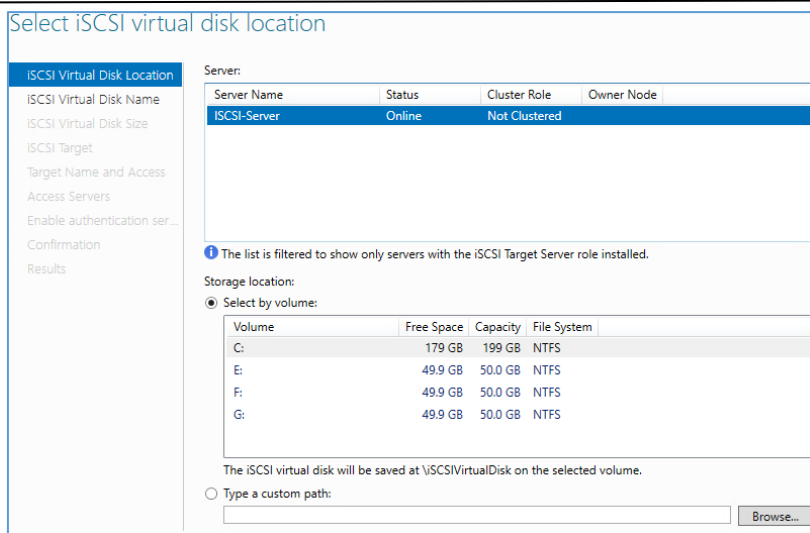
شکل ۱۱-۶

به‌مانند شکل ۱۱-۷ در قسمت File and Storage Services وارد iSCSI شوید و بر روی disk کلیک کنید.



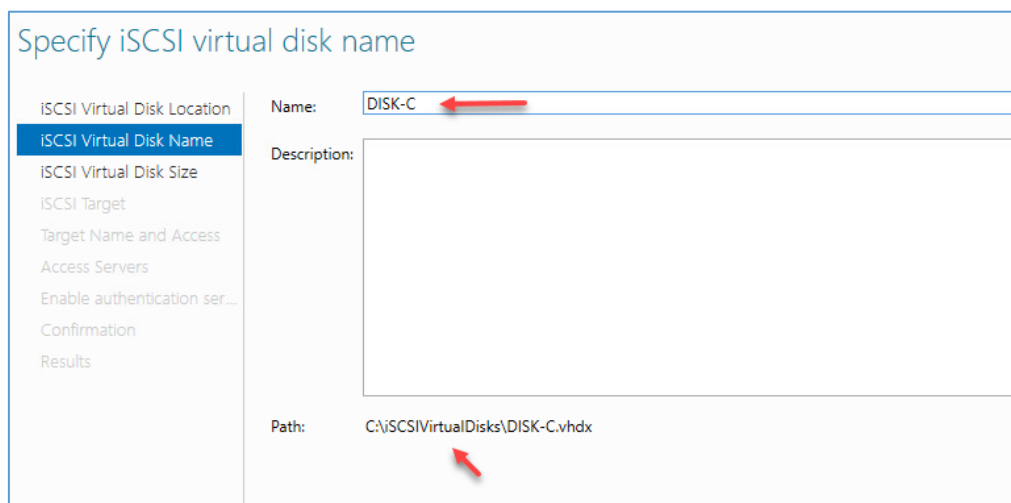
شکل ۱۱-۷

در شکل ۱۱-۸ تمام هارد دیسک‌های مجازی که به ماشین مورد نظر اضافه کردیم را مشاهده می‌کنید که در ادامه باید آن را برای سرروهای مورد نظر به اشتراک بگذاریم، یکی از هاردها را از لیست انتخاب کنید تا بیبر روی Next کلیک کنید؛ توجه داشته باشید که برای هر هارد باید این کار را انجام دهید.



شکل ۱۱-۸

در شکل ۱۱-۹ باید یک نام برای دیسک مجازی خود انتخاب کنید که در زیر صفحه مسیر ذخیره شدن این دیسک مشخص شده است.



شکل ۱۱-۹

در شکل ۱۱-۱۰ باید حجم مورد نظر برای ایجاد دیسک مجازی را مشخص کنید، که در اینجا ۴۹ گیگ وارد شده است که البته بستگی به نوع کار و حجم اطلاعات شما دارد، توجه داشته باشید اگر گزینه Fixed را انتخاب کند کل فضا به این هارد مجازی اختصاص داده خواهد شد ولی اگر گزینه Dynamically را انتخاب کنید به مرور زمان این حجم پر خواهد شد که پیشنهاد خود مایکروسافت گزینه اول است.

Specify iSCSI virtual disk size

Free space: 179 GB

Size: 49 GB

Fixed size

This type of disk provides better performance and is recommended for servers running applications with a high level of disk activity. The virtual hard disk is created using the size of the fixed virtual hard disk. It does not change when data is added or deleted.

Clear the virtual disk on allocation

Note: Un-selecting is NOT RECOMMENDED. Clearing a disk to zero will remove any fragments of data that remained on underlying storage, thus protecting from information leaks.

Dynamically expanding

شکل ۱۱-۱۰

در شکل ۱۱-۱۱ گزینه‌ی New iSCSI target را انتخاب کنید و بر روی Next کلیک کنید.

Assign iSCSI target

Assign this iSCSI virtual disk to an existing iSCSI target or create a new target for it.

Existing iSCSI target:

Target Name	Initiator IDs	Description
disk1	IQN:disk1-iscsi	
fgf	IQN:disk1-iscsi	

New iSCSI target

شکل ۱۱-۱۱

در شکل ۱۱-۱۲ باید یک نام برای target خود به دلخواه وارد و بر روی Next کلیک کنید.

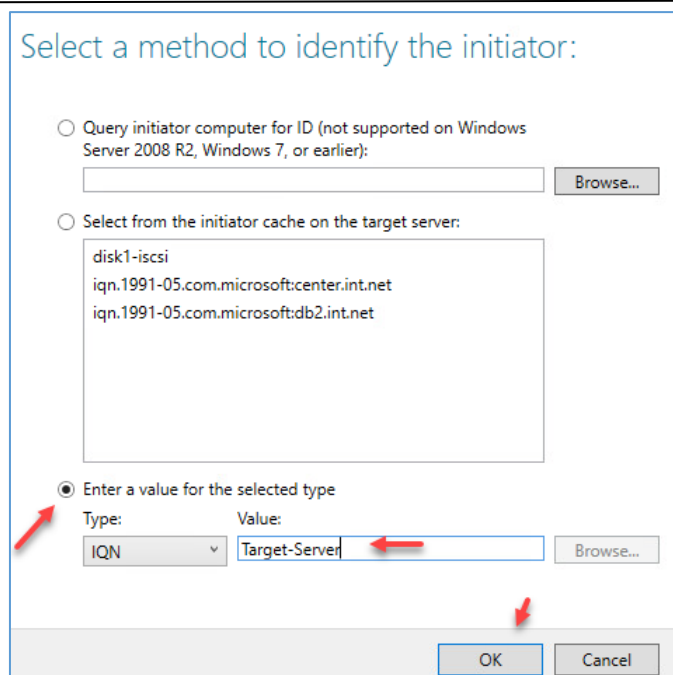
Specify target name

Name: Target-01

Description:

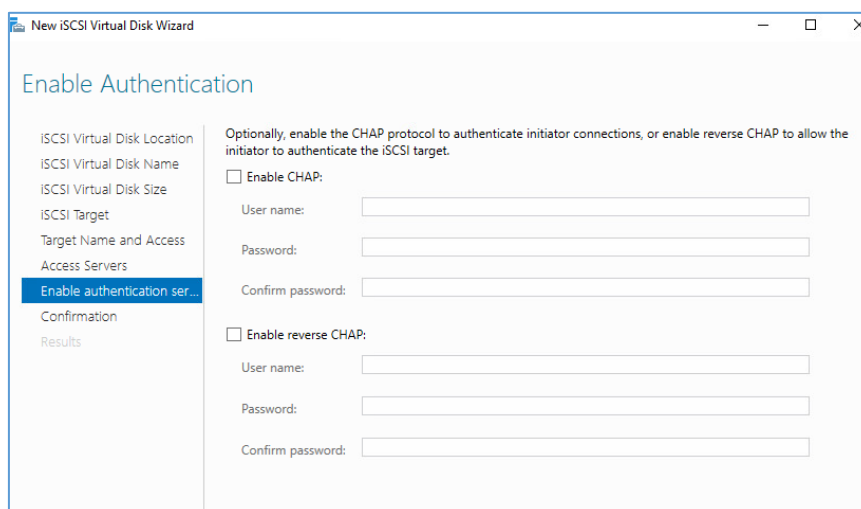
شکل ۱۱-۱۲

در شکل ۱۱-۱۳ گزینه‌ی Enter a value for selected type را انتخاب کنید و یک نام وارد کنید، توجه داشته باشید که این نام مهم بوده و در ادامه برای ارتباط با هارد دیسک مجازی از آن استفاده خواهد شد.



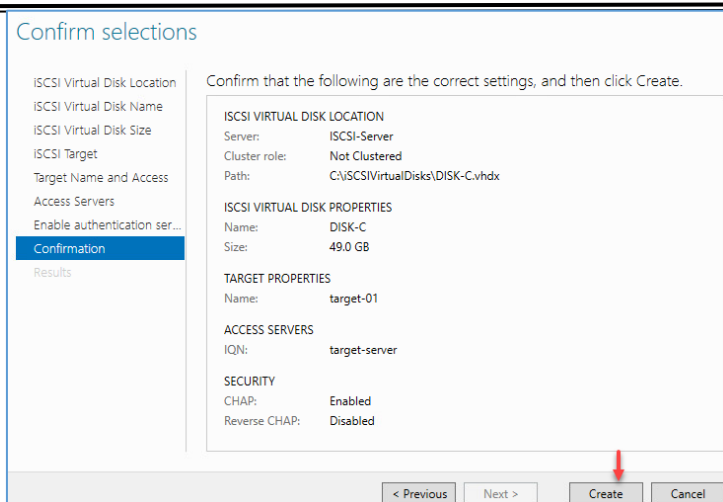
شکل ۱۱-۱۳

در شکل ۱۱-۱۴ می‌توانید ارتباط خود را رمزگذاری کنید، تا کسی نتواند به هارد دیسک مجازی مورد نظر دست پیدا کند، در حال حاضر نیازی به این کار نیست و بر روی Next کلیک کنید.

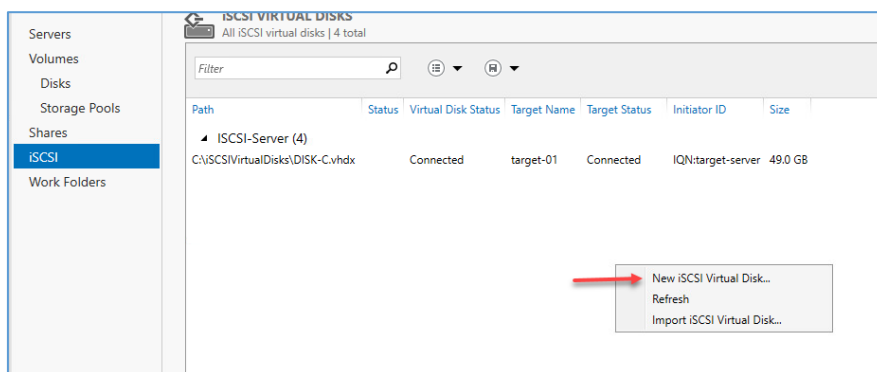


شکل ۱۱-۱۴

در شکل ۱۱-۱۵ بر روی Create کلیک کنید تا target مورد نظر به همراه هارد دیسک مجازی ایجاد شود.

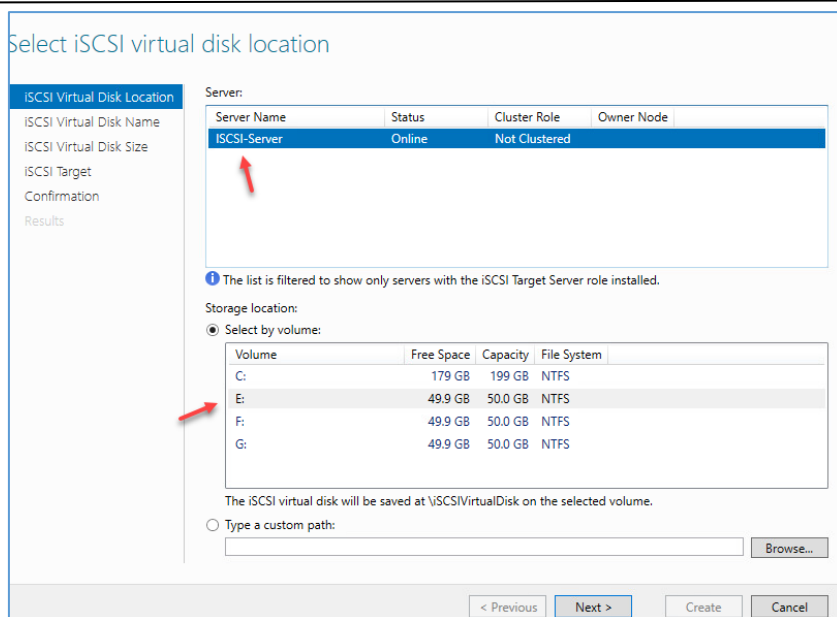


همانطور که در شکل ۱۶-۱۱ مشاهده می‌کنید، هارد دیسک مجازی مورد نظر داخل درایو C با حجم ۴۹ گیگابایت ایجاد شده است و در ادامه باید ۳ هارددیسک دیگر را هم به لیست اضافه کنید که برای اینکه به طور کامل متوجه شوید یکی از آنها را ایجاد می‌کنیم، بر روی صفحه‌ی مورد نظر کلیک راست کنید و گزینه‌ی New iSCSI Virtual Disk را انتخاب کنید.



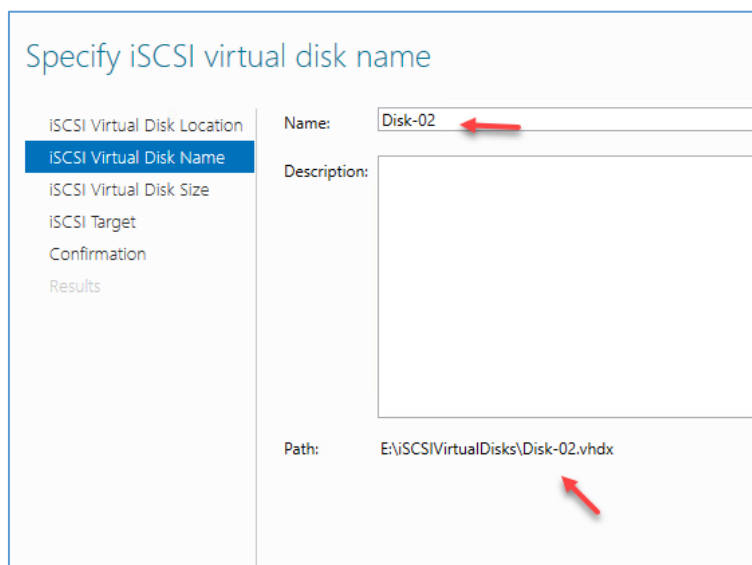
شکل ۱۶-۱۱

در شکل ۱۷-۱۱ که قبلاً درایو C را برای ایجاد هارد مجازی انتخاب کرده بودیم، حالا باید درایو بعدی یعنی E را انتخاب و بر روی Next کلیک کنیم.



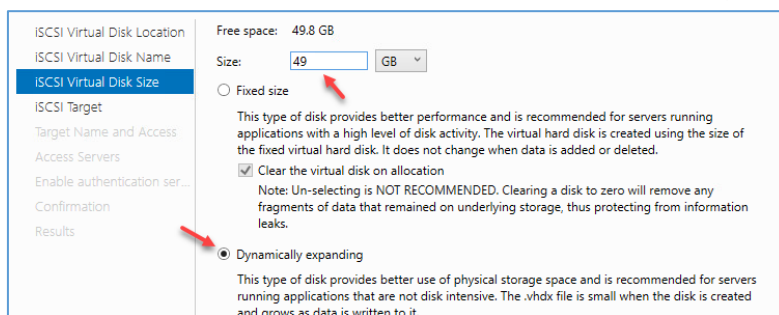
شکل ۱۱-۱۷

در شکل ۱۱-۱۸ باید نام دیسک را وارد کنید و در پایین صفحه محل ذخیره شدن آن را در درایو E مشاهده می‌کنید.



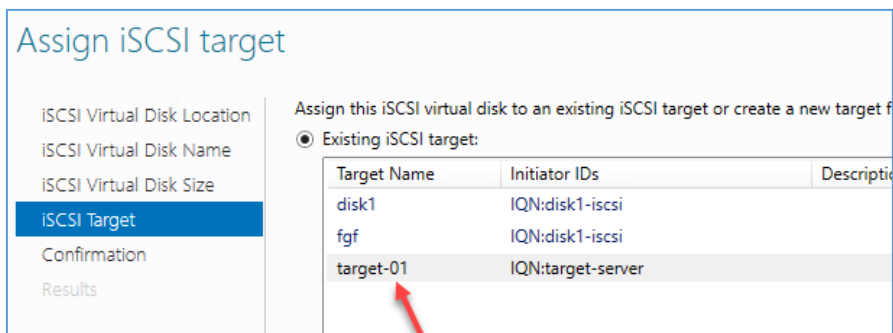
شکل ۱۱-۱۸

در شکل ۱۱-۱۹ باید حجم مورد نظر خود را وارد کنید که در اینجا ۴۹ گیگابایت را وارد و بر روی Next کلیک کنید.



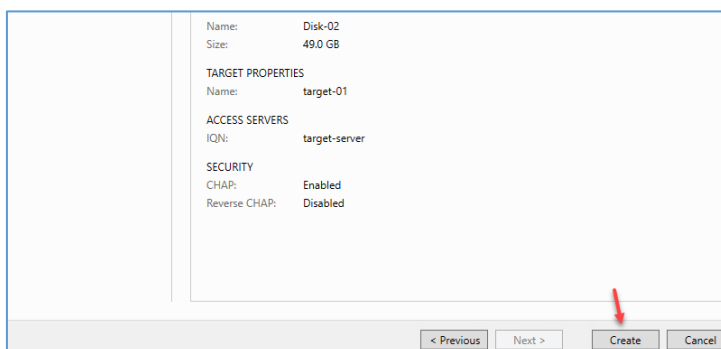
شکل ۱۱-۱۹

در شکل ۱۱-۲۰ باید در قسمت Existing همان target که در قسمت اول ایجاد کردیم را انتخاب کنید، با این کار این دیسک هم زیر مجموعه‌ی آن target خواهد شد.



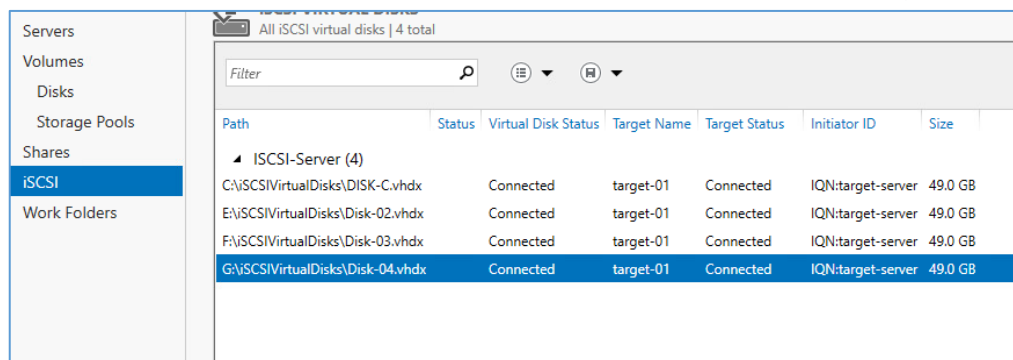
شکل ۱۱-۲۰

در شکل ۱۱-۲۱ هم باید بر روی Create کلیک کنید تا هارد دیسک مورد نظر ایجاد شود، توجه داشته باشید که دقیقاً همین مراحل را برای دو دیسک دیگر هم انجام دهید تا جمعاً ۴ دیسک به لیست اضافه شود.



شکل ۱۱-۲۱

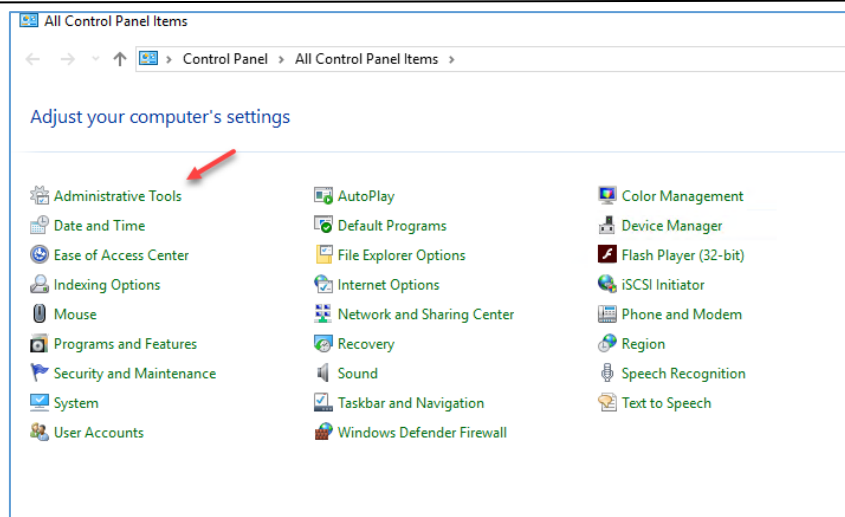
همان‌طور که در شکل ۱۱-۲۲ مشاهده می‌کنید هر چهار دیسک مورد نظر ایجاد شده است و حالا می‌توانیم به آنها متصل شویم.



شکل ۱۱-۲۲

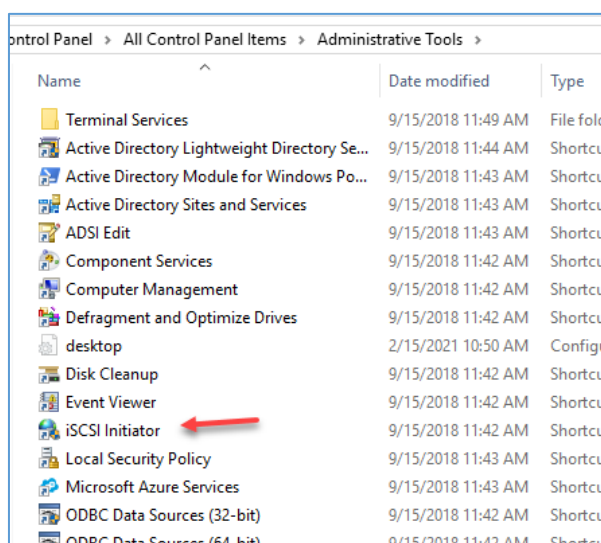
تنظیمات سرور Failover-1

در ادامه کار باید دو سرور Failover شویم و این چهار هارد را به آنها اضافه کنیم، به‌مانند شکل ۱۱-۲۳ وارد کنترل پنل هر کدام از سرورها شوید و بر روی Administrative Tools کلیک کنید.



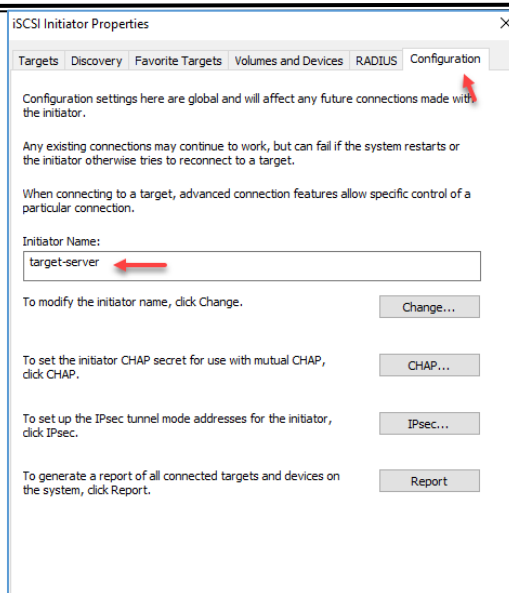
شکل ۱۱-۲۳

برای متصل شدن به دیسک‌های iSCSI ابزاری وجود دارد با نام iSCSI Initiator که در تمام ویندوزهای سرور موجود است و حتی می‌توانید آن را از اینترنت دانلود کنید و بر روی ویندوز مورد نظر خود نصب کنید، مانند شکل ۱۱-۲۴ بر روی iSCSI Initiator دو بار کلیک کنید.



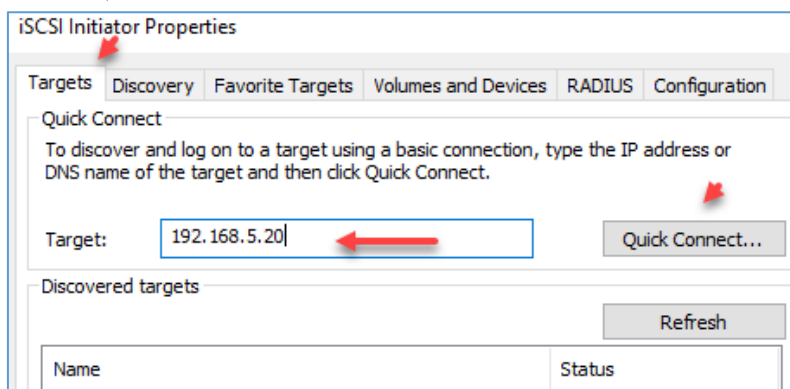
شکل ۱۱-۲۴

در شکل ۱۱-۲۵ و در تب Configuration باید نام Initiator Name را که با نام target—server ایجاد کردیم را وارد کنید.



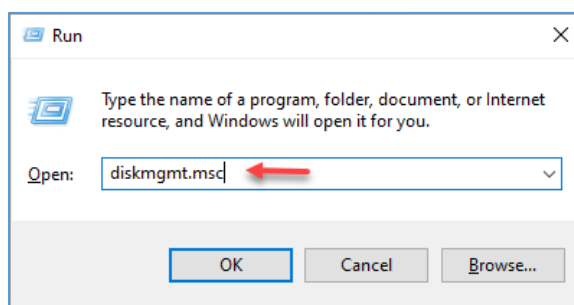
شکل ۱۱-۲۵

در شکل ۱۱-۲۶ و در تب Targets باید آدرس IP سرور iSCSI را وارد و بر روی Connect کلیک کنید، با این کار شما توانستید هارد دیسک مورد نظر را به لیست پارتیشن‌های خود اضافه کنید که با هم آنها را آماده‌سازی می‌کنیم.



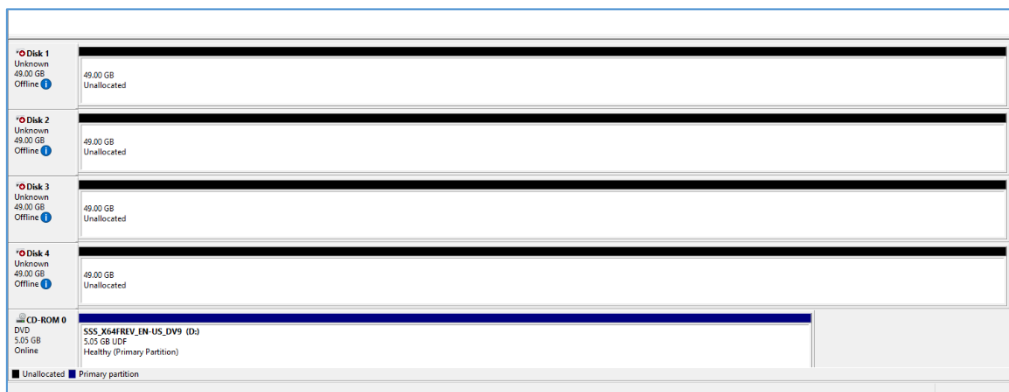
شکل ۱۱-۲۶

برای اینکه در سرور Failover-01 که عملیات بالا را در آن انجام دادیم بتوانیم هارد دیسک‌های iSCSI را مشاهده کنیم باید به‌مانند شکل ۱۱-۲۷ در RUN دستور diskmgmt.msc را اجرا کنید تا سرویس Disk Management اجرا شود.



شکل ۱۱-۲۷

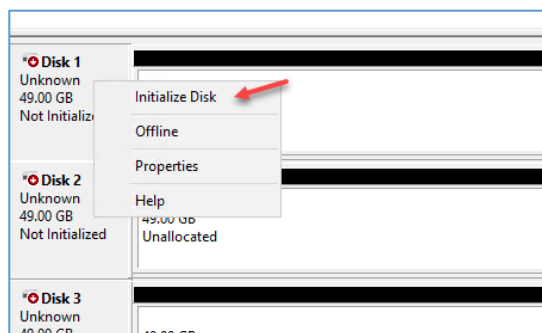
همان‌طور که در شکل ۱۱-۲۸ مشاهده می‌کنید هر چهار هارد به لیست اضافه شده است و باید تنظیمات آنها را انجام دهیم.



شکل ۱۱-۲۸

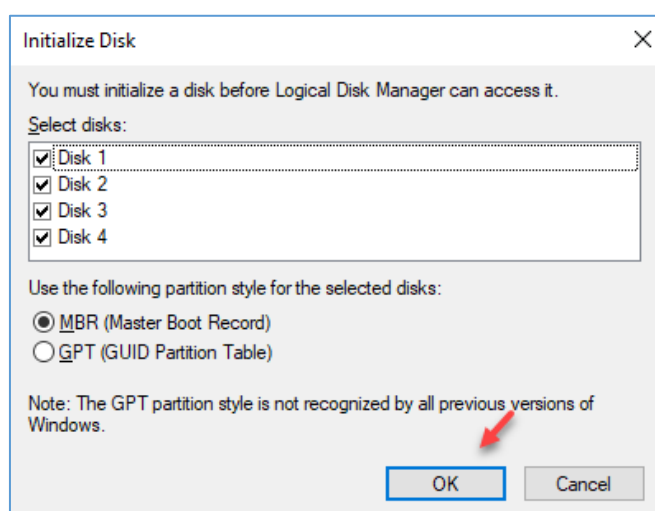
برای اینکه این

هارد دیسک‌ها را تنظیم کنیم باید به‌مانند شکل ۱۱-۲۹ بر روی دیسک مورد نظر کلیک راست کنید و گزینه‌ی اول یعنی Online و بعد Initialize Disk را انتخاب کنید.



شکل ۱۱-۲۹

در شکل ۱۱-۳۰ می‌توانید MBR که برای هارد دیسک‌هایی با حداکثر ظرفیت ۲ ترابایت است و GPT برای هارد دیسک‌هایی بیشتر از ۲ ترابایت، پس گزینه‌ی MBR را انتخاب و بر روی OK کلیک کنید.

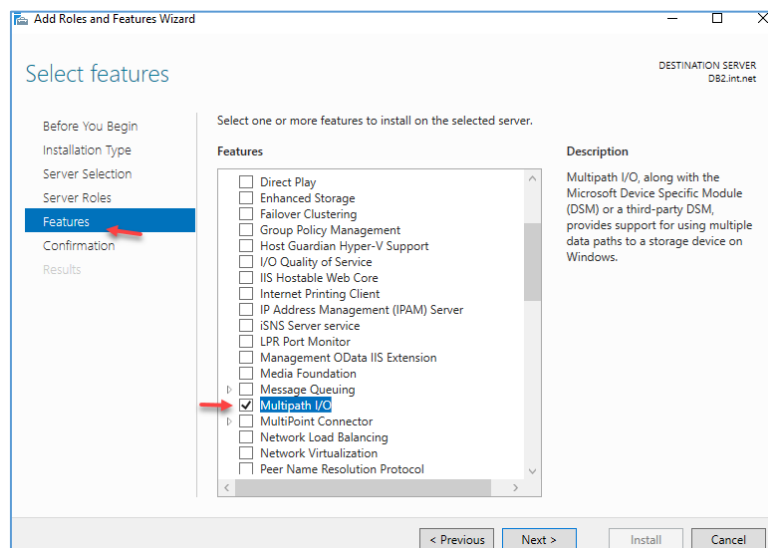


شکل ۱۱-۳۰

خوب کمی کارهایی که تا این مرحله انجام دادیم را بررسی کنیم، یک سرور iSCSI ایجاد کردیم و روی آن چند هارد متصل کردیم و با سرویس iSCSI چهار هارد مجازی که برای ادامه کار طراحی شده است را ایجاد کردیم و در ادامه از طریق سرور Failover-1 توانستیم از طریق سرویس iSCSI Initiator به هاردهای iSCSI متصل شویم و بعد در سرویس Diskmgmt.msc هاردهای اضافه شده را اول Online و بعد Initialize کردیم و دیگر نباید کار انجام دهیم. حالا باید وارد سرور Failover-2 شویم و هاردهای iSCSI را به آن هم اضافه کنیم.

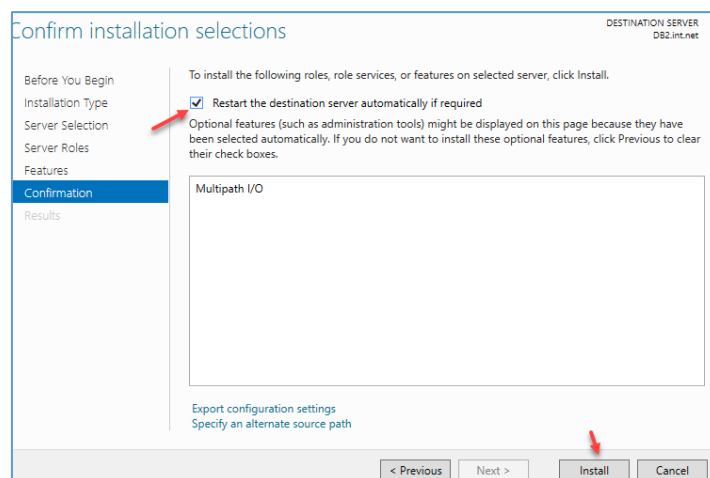
تنظیمات سرور Failover-2

شما در این سرور هم باید آن چهار هارد iSCSI را به لیست اضافه کنید، اما یک مشکل اساسی وجود دارد و آن هم این است که شما نمی‌توانید بیشتر از یک سرور را به صورت پیش فرض به سرور iSCSI متصل کنید، برای این کار باید یک سرویس دیگر را بر روی سرور iSCSI نصب کنید و آن هم سرویس Multipath I/O است که به مانند شکل ۱۱-۳۱ باید وارد Server Manager در سرور iSCSI شوید و از قسمت Features تیک گزینه‌ی Multipath I/O را انتخاب و بر روی Next کلیک کنید.



شکل ۱۱-۳۱

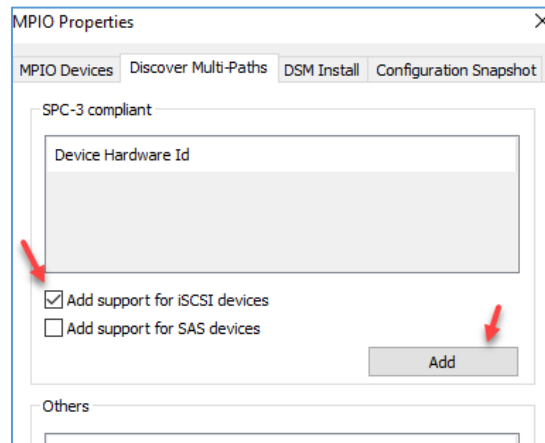
در شکل ۱۱-۳۲ حتماً تیک گزینه‌ی Restart the destination... را انتخاب و بر روی Install کلیک کنید.



شکل ۱۱-۳۲

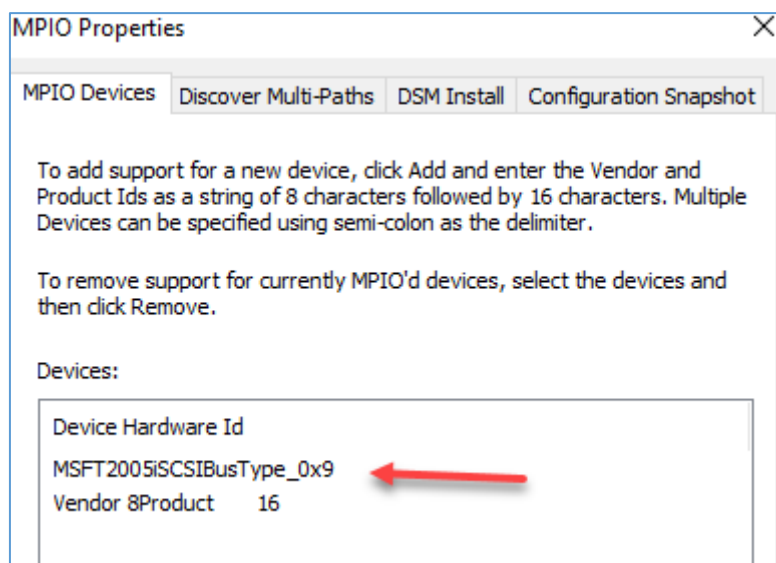
نکته مهم: این سرویس را در هر سه سرور iSCSI, Failover-1, Failover-2 حتماً باید نصب کنید و به صورت زیر تنظیمات آن را انجام دهید.

بعد از نصب سرویس وارد Administrative Tools شوید و سرویس MPIO را اجرا کنید، بعد از اجرا به مانند شکل ۱۱-۳۳ وارد تب Discover Multi-Paths شوید و تیک گزینه Add support را انتخاب کنید و بر روی Add کلیک کنید.



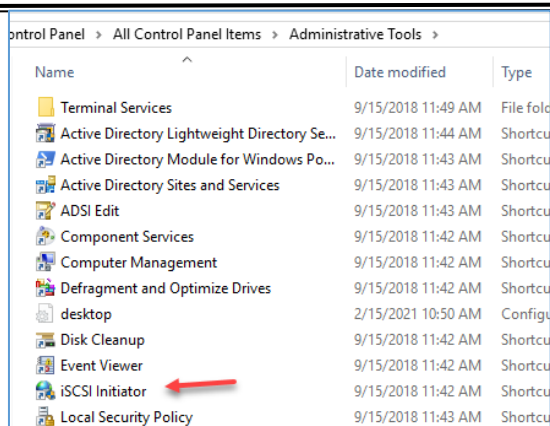
شکل ۱۱-۳۳

به مانند شکل ۱۱-۳۴ وارد تب MPIO Devices شوید و باید در لیست مورد نظر Device مورد نظر را حتماً مشاهده کنید تا عملیات موفقیت آمیز باشد.

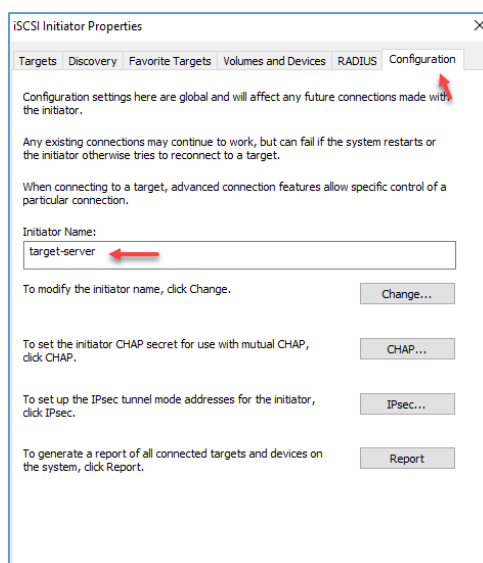


شکل ۱۱-۳۴

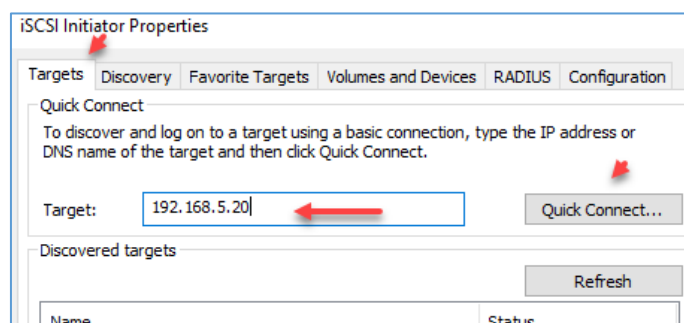
بعد از انجام مراحل بالا باید وارد سرور Failover-2 شوید و به مانند شکل ۱۱-۳۵ iSCSI Initiator را اجرا کنید و دقیقاً همان کاری که در سرور اول یعنی Failover-1 انجام دادید در این قسمت هم انجام دهید.



شکل ۱۱-۳۵



شکل ۱۱-۳۶

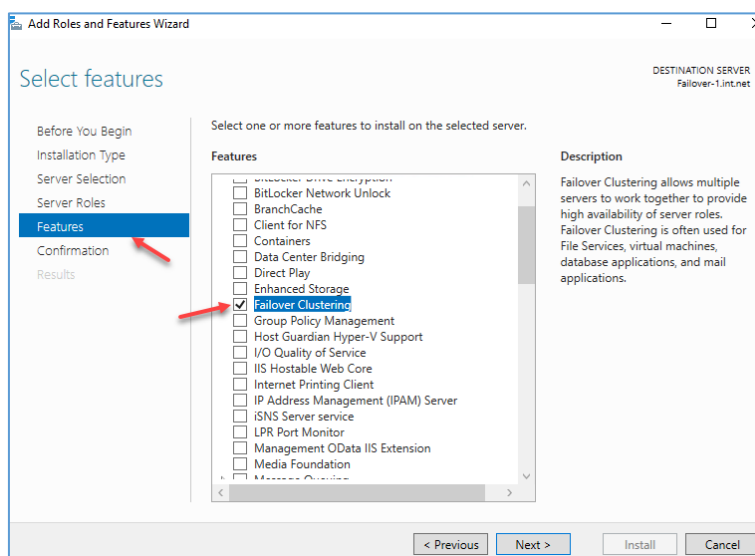


شکل ۱۱-۳۷

در ادامه هارد دیسک مورد نظر به لیست اضافه و کارهایی را که در سرور Failover-1 انجام دادیم در این سرور هم باید انجام دهیم.

بعد از اینکه هارد دیسک‌های مجازی را در هر سرور ایجاد کردیم باید سرویس‌های دیگری را هم فعال کنیم، سرویس دیگری که از گذشته وجود داشته با نام Application Server بوده که تا ویندوز سرور ۲۰۱۲ وجود داشته ولی در ویندوز سرور ۲۰۱۶ و ۲۰۱۹ این گزینه وجود ندارد ولی سرویس‌های آن به صورت پیش فرض بر روی ویندوز فعال است و مشکلی از این لحاظ نخواهید داشت.

در ادامه باید سرویس Clustering را بر روی هر دو سرور Failover-1 و Failover-2 نصب کنید، به این صورت که مانند شکل ۱۱-۳۸ وارد Server Manager شوید و تیک گزینهی Failover Clustering را از قسمت Features انتخاب و نصب کنید.

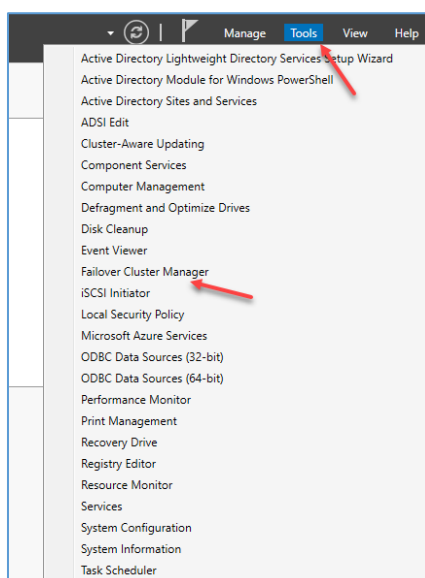


شکل ۱۱-۳۸

نصب این سرویس کمی زمان بر خواهد بود، توجه داشته باشید که بعد از نصب سرویس حتماً سرور باید Restart شود.

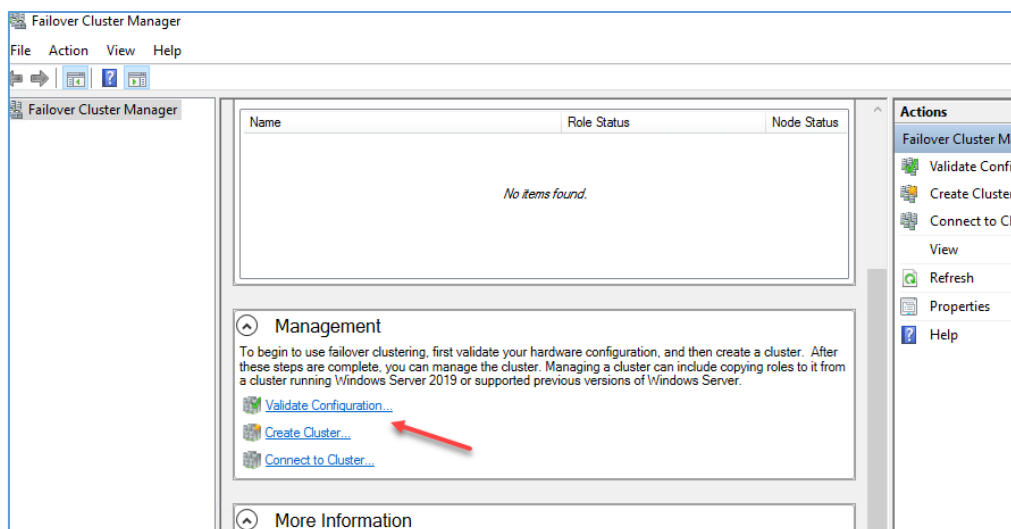
در ادامه باید کنسول مدیریتی سرویس Clustering را در هر دو سرور اجرا کنید و کارهای زیر را در هر دو سرور انجام دهید.

به مانند شکل ۱۱-۳۹ وارد Server Manager شوید و از منوی Tools سرویس Clustering را اجرا کنید.



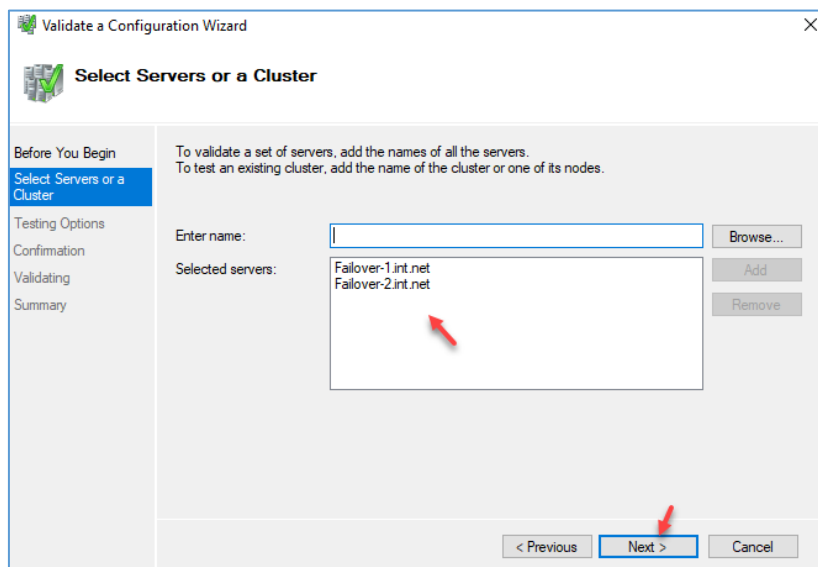
شکل ۱۱-۳۹

اولین کاری که در این سرورس باید انجام دهیم این است که تست می‌گیریم ببینیم ارتباط و تنظیمات انجام شده در هر دو سرور که قرار است کلاستر شوند به درستی انجام شده است یا نه، برای این کار به‌مانند شکل ۴۰-۱۱ بر روی Validate Configuration کلیک کنید.



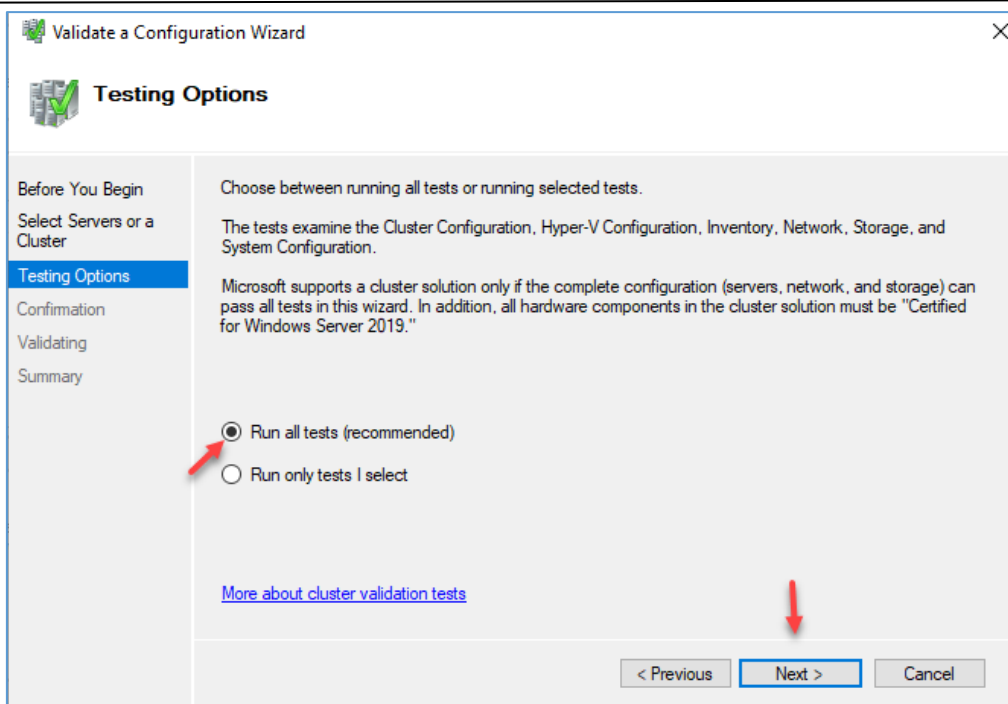
شکل ۴۰-۱۱

در شکل ۴۱-۱۱ باید نام سرور مورد نظر خود را وارد و بر روی Add کلیک کنید، توجه داشته باشید باید نام هر دو سرور را به لیست اضافه کنید.



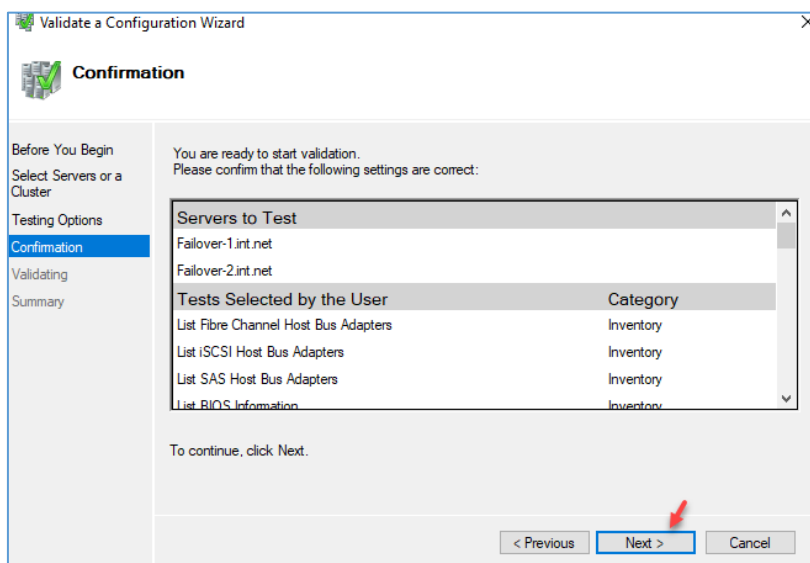
شکل ۴۱-۱۱

در شکل ۴۲-۱۱ باید گزینه‌ی Run all tests را انتخاب کنید تا قبل از اینکه موضوع را عملیاتی کنیم مراحل کار تست شود.



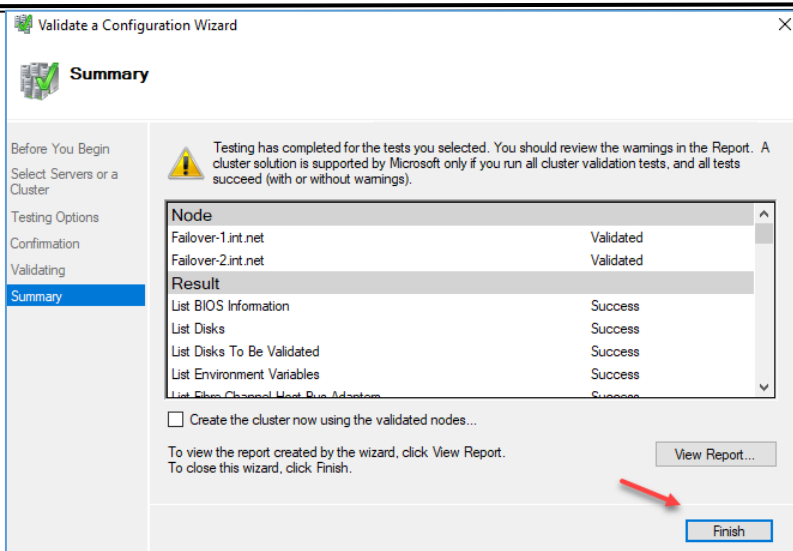
شکل ۱۱-۴۲

در ادامه کار و در شکل ۱۱-۴۳ بر روی Next کلیک کنید.



شکل ۱۱-۴۳

همانطور که در شکل ۱۱-۴۴ مشاهده می‌کنید عملیات با موفقیت انجام شده است، اما شاید برای شما با خطاهای مختلفی تامل شده باشد که حتماً باید آن خطاها را بررسی کنیم.



شکل ۱۱-۴۴

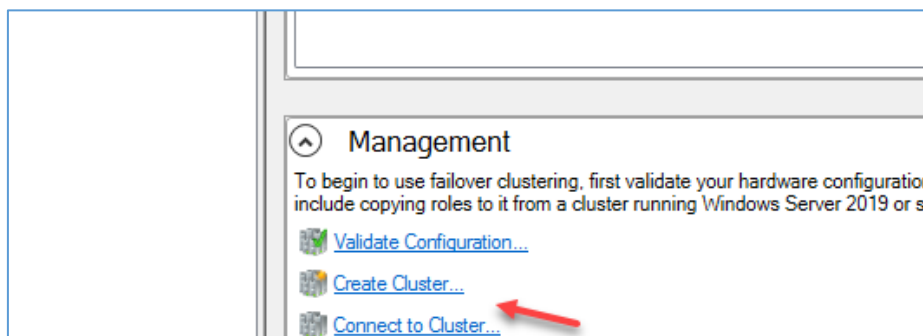
خطاهایی که معمولاً در این قسمت روی می‌دهد شامل موارد زیر است:

- ۱- کارت شبکه‌ی دوم تعریف نشده باشد.
- ۲- سیستم تحت دامین نباشد.
- ۳- هارد دیسک به درستی در هر دو سرور ایجاد نشده باشد، که این خطا یکی از متداولترین خطاها است.
- ۴- سرویس MPIO در هر سه سرور فعال نباشد.

اگر تمامی مواردی بالا را به درستی انجام دهید دیگر با خطا مواجه نخواهید شد.

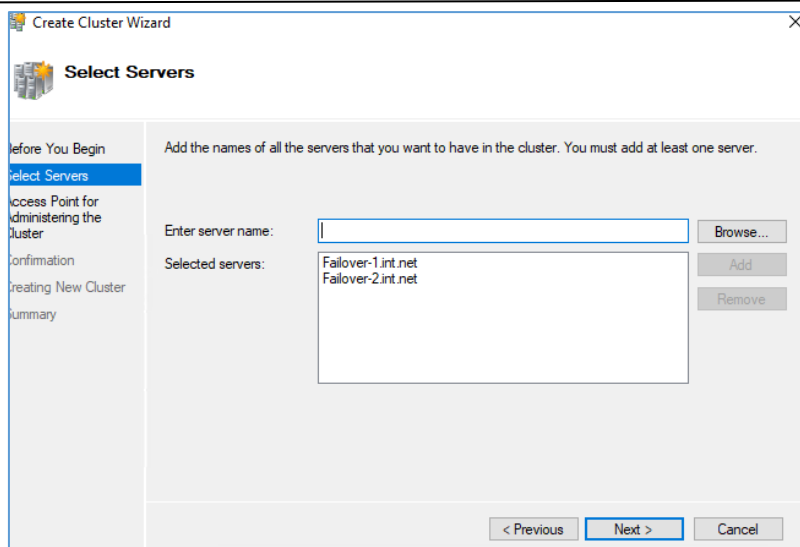
در ادامه کار و بعد از تست موفقیت آمیز کار باید Cluster اصلی خود را ایجاد کنیم برای این کار به‌مانند شکل ۴۵-۱۱

۱۱ بر روی Create Cluster



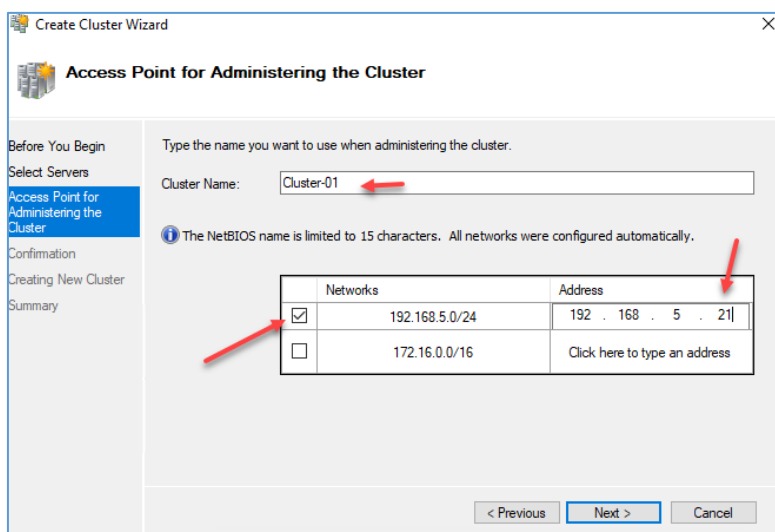
شکل ۱۱-۴۵

در شکل ۴۶-۱۱ هر دو سرور failover را وارد و بر روی Next کلیک کنید.



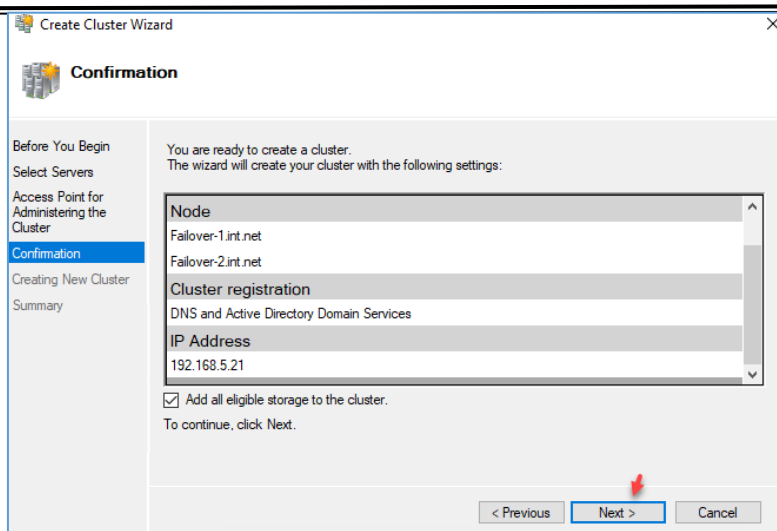
شکل ۱۱-۴۶

در شکل ۱۱-۴۷ باید مشخص کنید که کدام شبکه به عنوان شبکه‌ی Cluster باید انتخاب شود، و زمانی که شبکه‌ی Cluster را انتخاب می‌کنید باید برای آن یک آدرس IP مشخص در نظر بگیرید، که در اینجا شبکه‌ی اصلی یعنی 192.168.5.21 انتخاب شده و آدرس 192.168.5.21 به عنوان شبکه‌ی Cluster در نظر گرفته خواهد شد و بعد از اتمام کانفیگ اگر همه چیز به درستی پیش برود شما می‌توانید این IP را Ping کنید و از آن استفاده کنید.



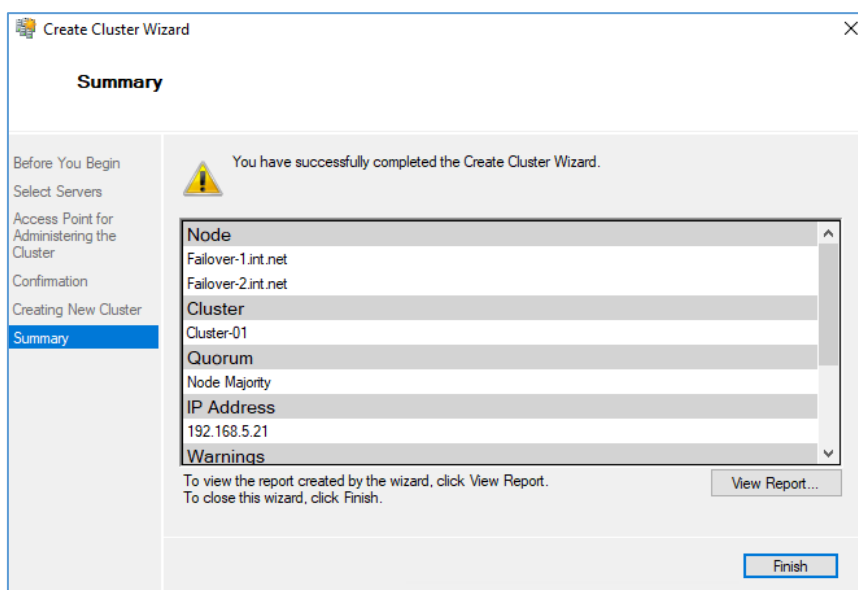
شکل ۱۱-۴۷

به مانند شکل ۱۱-۴۸ بر روی Next کلیک کنید تا کار نصب آغاز شود.



شکل ۱۱-۴۸

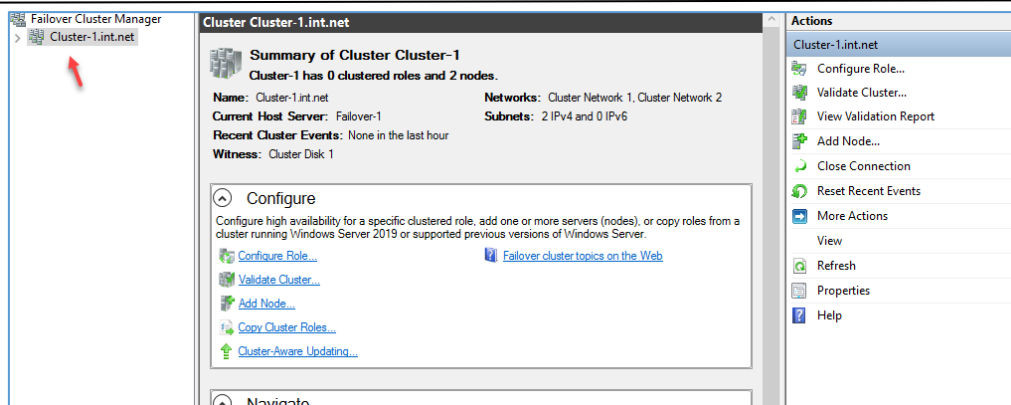
همان‌طور که در شکل ۱۱-۴۹ مشاهده می‌کنید عملیات Clustering به درستی انجام شده است و حالا ما بر روی آدرس 192.168.5.21 سرویس کلاستر داریم و می‌توانید در ادامه از آن استفاده کنیم.



شکل ۱۱-۴۹

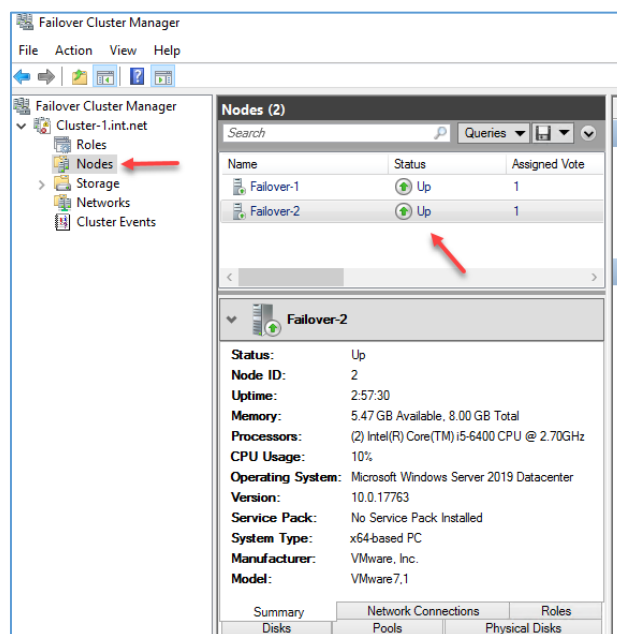
توجه داشته باشید که اگر در یک سرور عملیات ایجاد Clustering را انجام دهید در سرور دیگر هم همین تنظیمات انجام خواهد شد.

همان‌طور که در شکل ۱۱-۵۰ مشاهده می‌کنید Cluster مورد نظر با موفقیت ایجاد شده است.



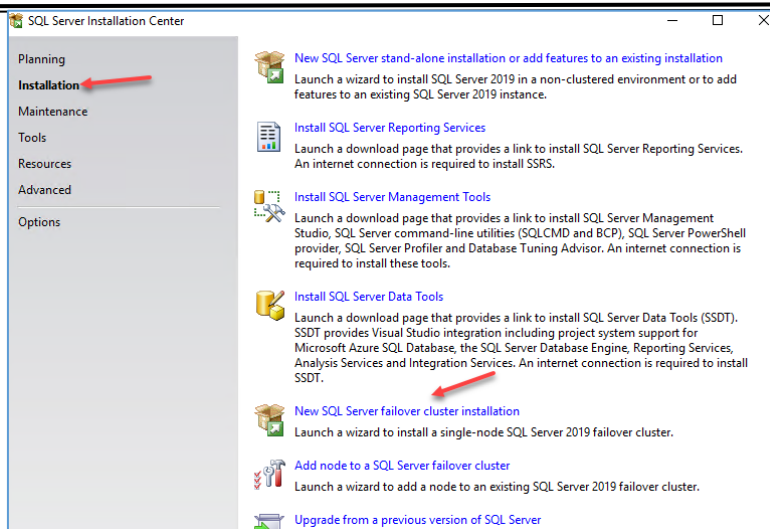
شکل ۵۰-۱۱

اگر به‌مانند شکل ۵۱-۱۱ وارد قسمت Node شوید دو سرور Failover-1 و Failover-2 را مشاهده می‌کنید که هر دو در حالت روشن یعنی UP قرار دارند و می‌توانند سرویس ارائه دهند.



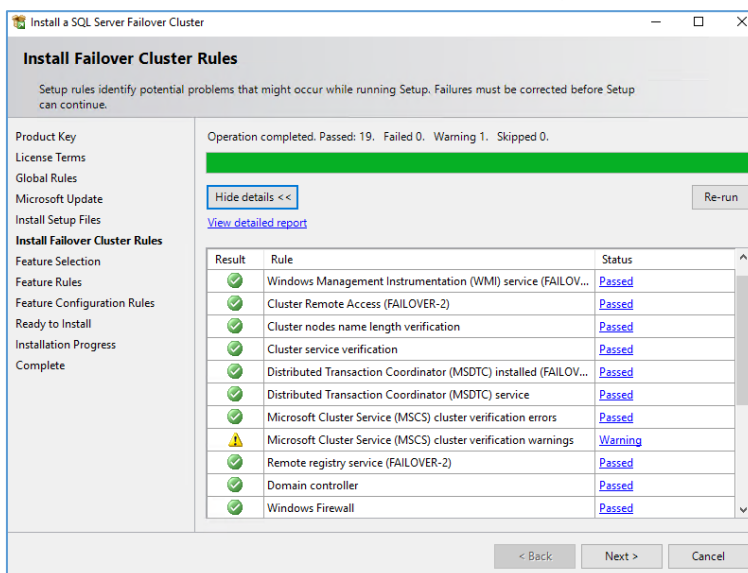
شکل ۵۱-۱۱

در ادامه کار باید نرم‌افزار SQL Server را بر روی هر دو سرور Failover-1 و Failover-2 نصب کنیم، البته روش نصب با قبل تفاوت خواهد داشت. وارد یکی از سرورها مثلاً Failover-2 می‌شویم و نرم‌افزار SQL Server را برای نصب اجرا می‌کنیم. به‌مانند شکل ۵۲-۱۱ وارد Installation شوید و بر روی Installation کلیک کنید و در صفحه‌ی باز شده بر روی New failover cluster SQL Server کلیک کنید.



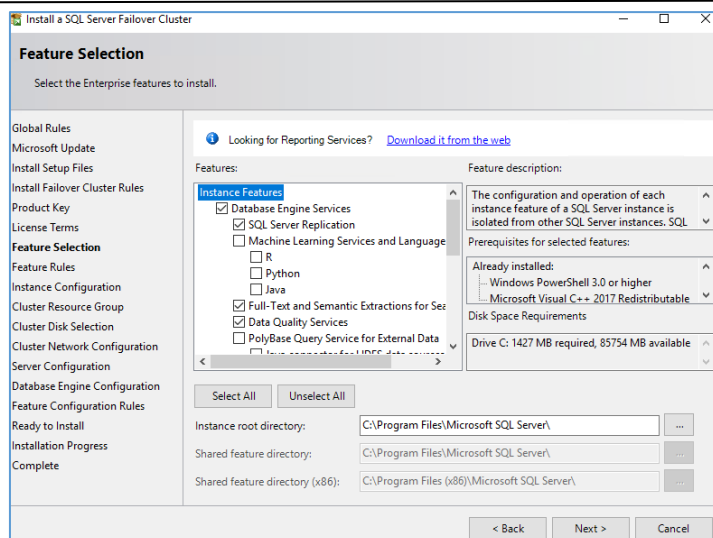
شکل ۱۱-۵۲

در شکل ۱۱-۵۳ بررسی‌های اولیه انجام می‌شود که اگر از نظر Clustering مشکلی نباشد می‌توانید کار را ادامه دهید.



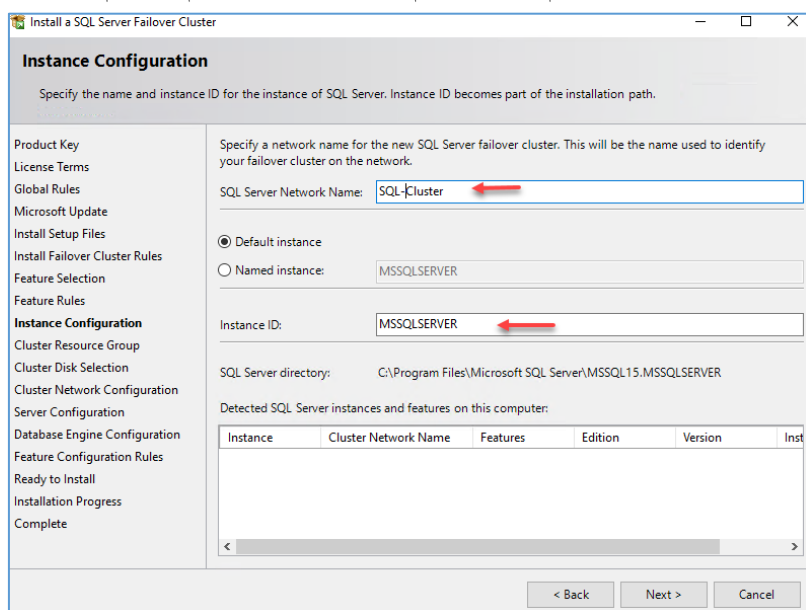
شکل ۱۱-۵۳

در شکل ۱۱-۵۴ تیک گزینه‌ی Database Engine Services را انتخاب و بر روی Next کلیک کنید، البته این موارد را قبلاً توضیح دادیم.



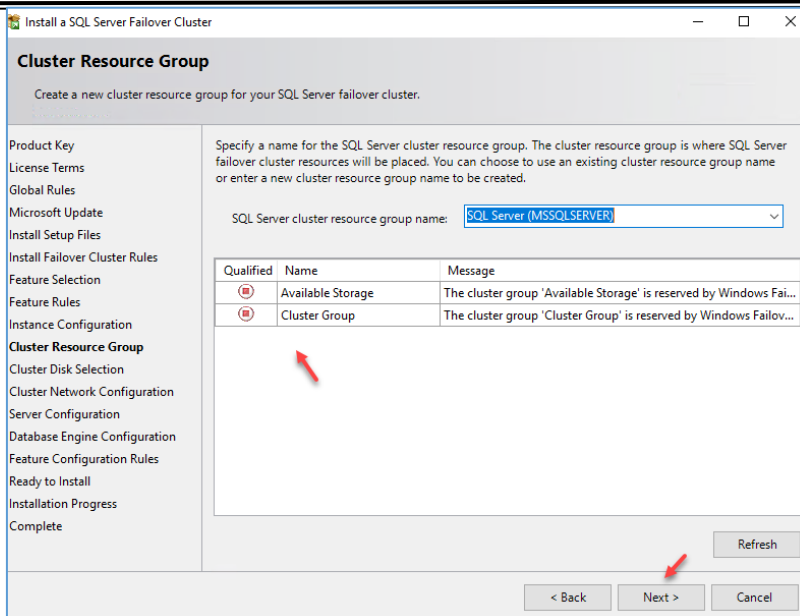
شکل ۱۱-۵۴

در شکل ۱۱-۵۵ باید در قسمت SQL Server Network Name یک نام به دلخواه خود وارد کنید، البته این نام مهم است و برای دسترسی به SQL باید از این نام استفاده کنیم، در قسمت بعدی هم باید نام Instance را وارد کنید.



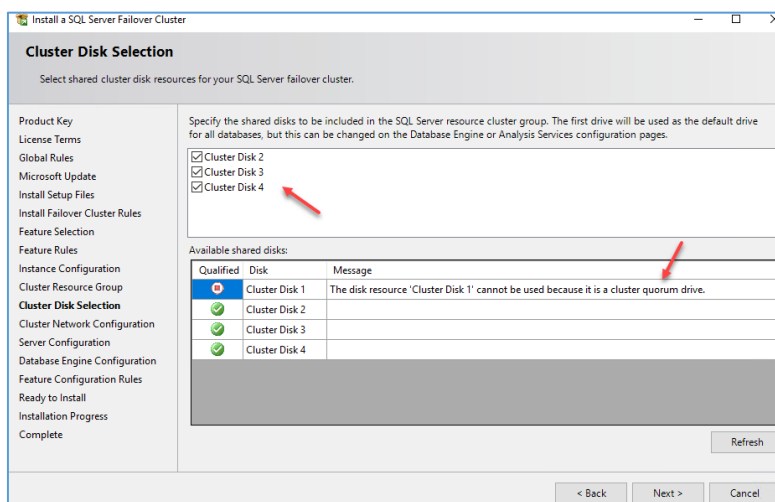
شکل ۱۱-۵۵

در شکل زیر دو سرویس موردنیاز برای فعال‌سازی Clustering در SQL را مشاهده می‌کنید، بر روی Next کلیک کنید.



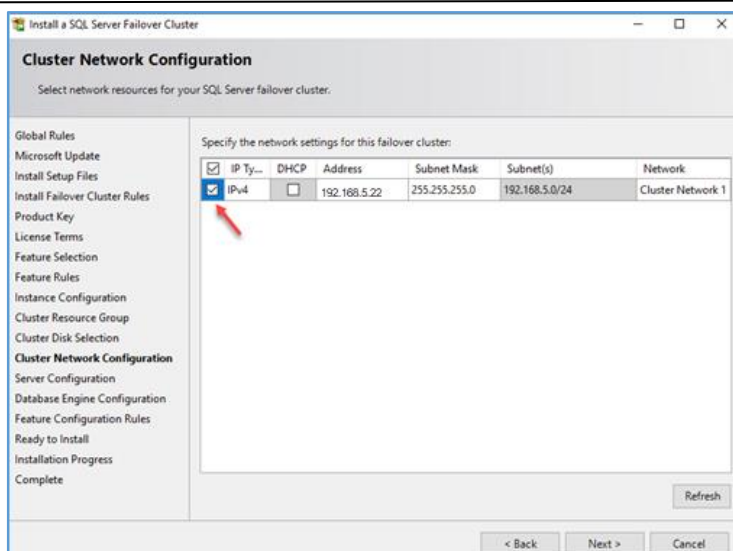
شکل ۱۱-۵۶

در شکل ۱۱-۵۷ باید تمامی دیسک‌های موجود را انتخاب کنید، فقط سه دیسک در لیست قرار دارد، اما یک دیسک دیگر که دیسک شماره‌ی یک باشد در لیست برای انتخاب وجود ندارد، دلیل آن هم این است که اصولاً زمانی که سیستم Clustering را فعال می‌کنید، دیسک اول به صورت پیش فرض به عنوان دیسک Quorum انتخاب می‌شود، این دیسک تشکیل دهنده‌ی تنظیمات اصلی Clustering است و به صورت خودکار تنظیم می‌شود، البته می‌توانستیم در قسمت ایجاد Clustering این دیسک را تنظیم کنیم تا خودش یک دیسک را انتخاب نکند، در شکل ۱۱-۵۷ مشخص شده است که کدام دیسک به عنوان دیسک Quorum انتخاب شده است.



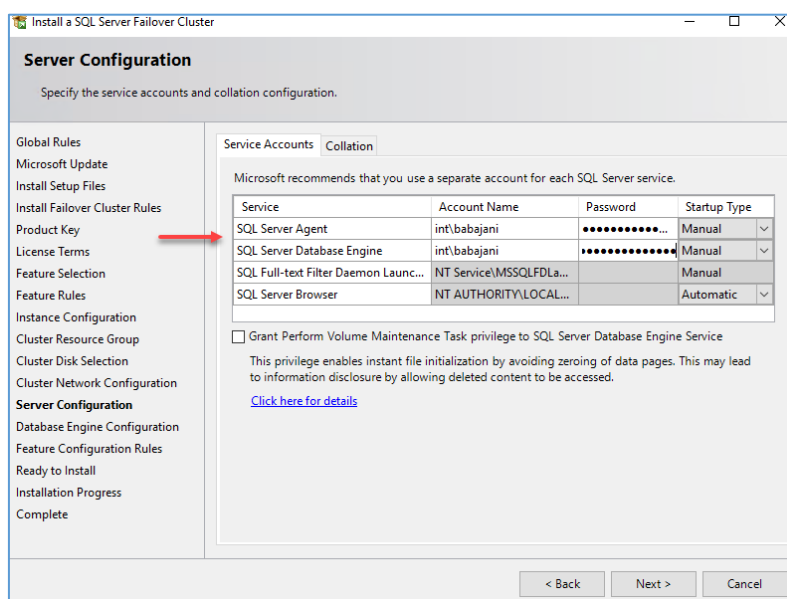
شکل ۱۱-۵۷

در شکل ۱۱-۵۸ باید شبکه‌ی IPV4 خود را انتخاب کنید و یک آدرس IP جدید برای SQL وارد کنید، توجه داشته باشید این IP با IP که در قسمت Clustering وارد کردیم متفاوت خواهد بود.



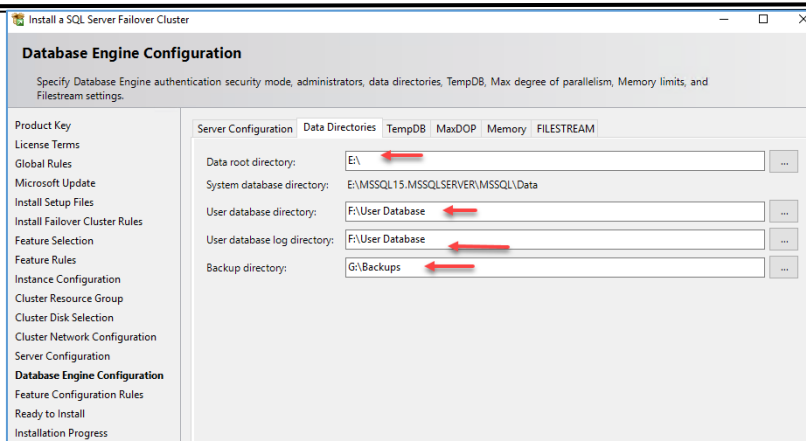
شکل ۱۱-۵۸

در شکل ۱۱-۵۹ باید نام کاربری که قرار است دسترسی کامل برای اجرای سرویس‌ها را داشته باشد وارد کنید و بر روی Next کلیک کنید.



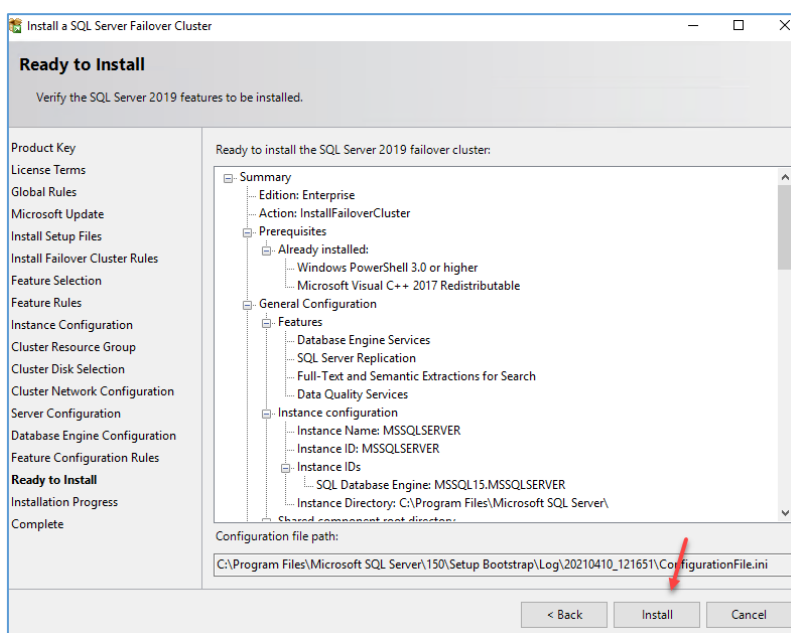
شکل ۱۱-۵۹

در شکل ۱۱-۶۰ و در تب Server Configuration بر روی Current User کلیک کنید و بعد در تب Data Directories باید محل ذخیره شدن دیتابیس‌ها و اطلاعات کاربران و فایل‌های پشتیبان را مشخص کنید، سعی کنید هر قسمت را در یکی از هاردهای Cluster که ایجاد کردیم وارد کنید و بر روی Next کلیک کنید.



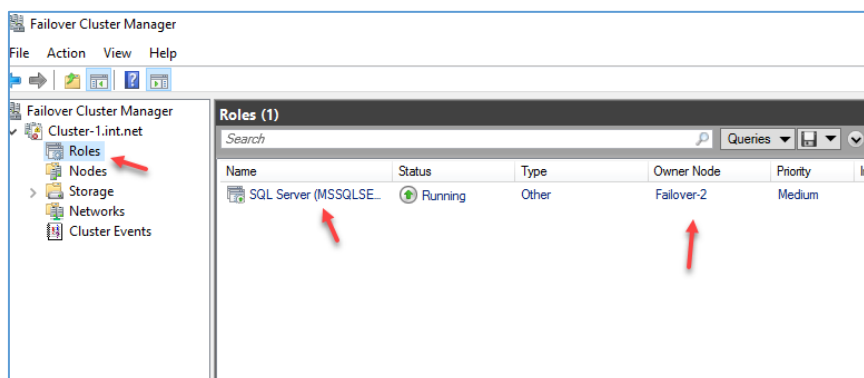
شکل ۱۱-۶۰

در شکل ۱۱-۶۱ بر روی Install کلیک کنید تا کار نصب آغاز شود.



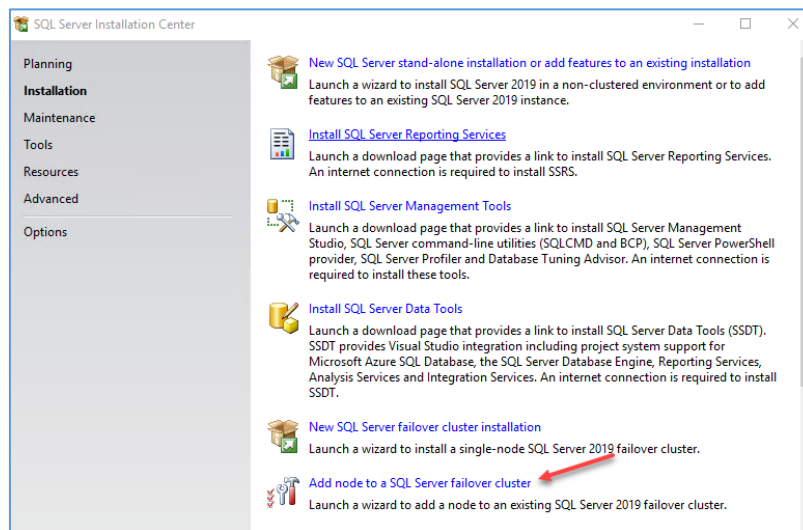
شکل ۱۱-۶۱

بعد از نصب سرویس Clustering برای SQL سرور اگر وارد سرویس Failover Cluster Manager به مانند شکل ۱۱-۶۲ شوید از سمت چپ بر روی Roles کلیک کنید، همان طور که مشاهده می کنید با SQL Server Instance مورد نظر توسط سرور 2-Failover ایجاد شده است.



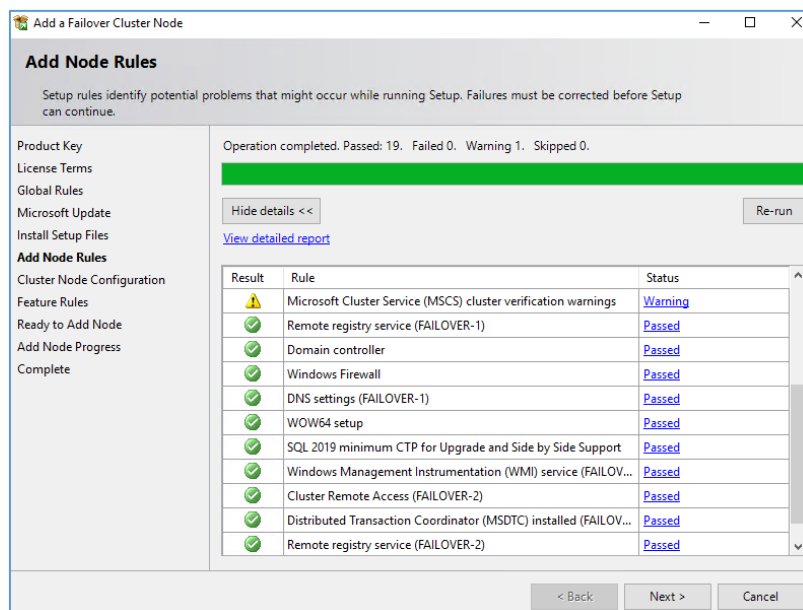
شکل ۱۱-۶۲

بعد از اینکه سرویس SQL را برای سرور Failover-2 راه اندازی کردیم باید حتماً در سرور Failover-1 هم این سرویس را فعال کنیم تا عملیات Clustering به درستی انجام شود، برای این کار باید دوباره نرم افزار نصب SQL را اجرا کنید و به مانند شکل ۱۱-۶۳ بر روی Add node to a SQL Server Failover Clustering کلیک کنید.



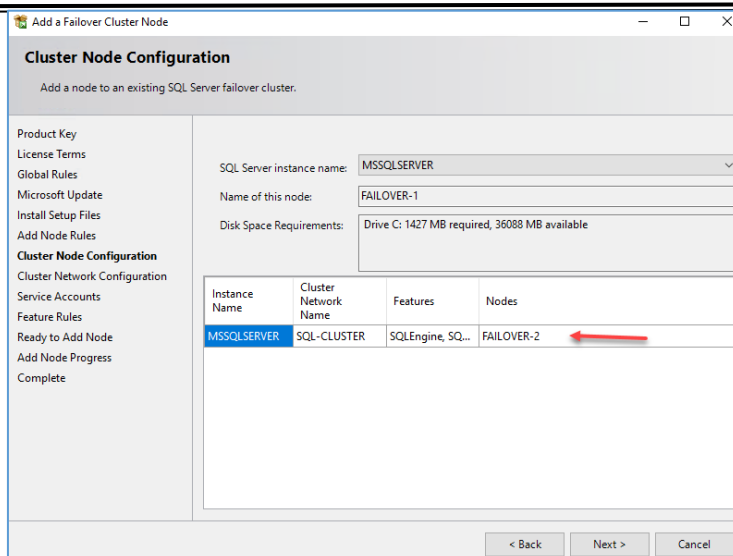
شکل ۱۱-۶۳

در شکل ۱۱-۶۴ بررسی های مربوط به Clustering انجام شده و در صورت نبود خطا بر روی Next کلیک کنید.



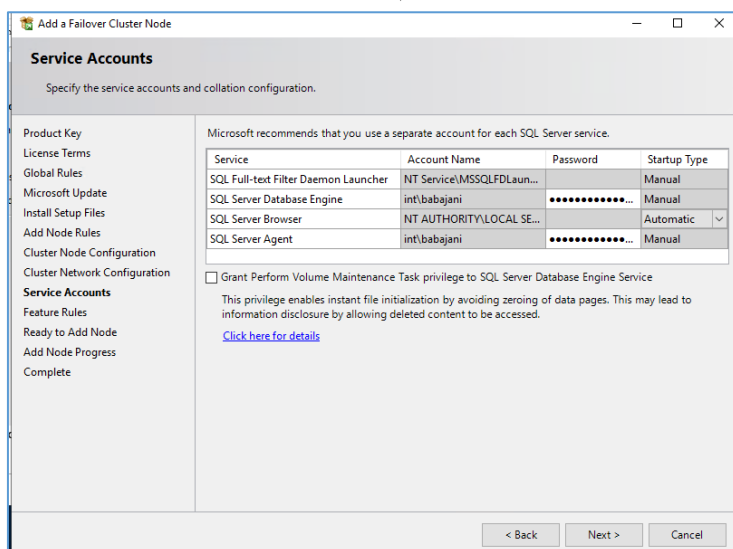
شکل ۱۱-۶۴

همان طور که در شکل ۱۱-۶۵ مشاهده می کنید کلاستر مورد نظر که با نام شبکه ی SQL-Cluster ایجاد کرده بودیم به صورت اتوماتیک پیدا شده که نشان می دهد این نام را از سرور Failover-2 گرفته است، برای ادامه کار بر روی Next کلیک کنید.



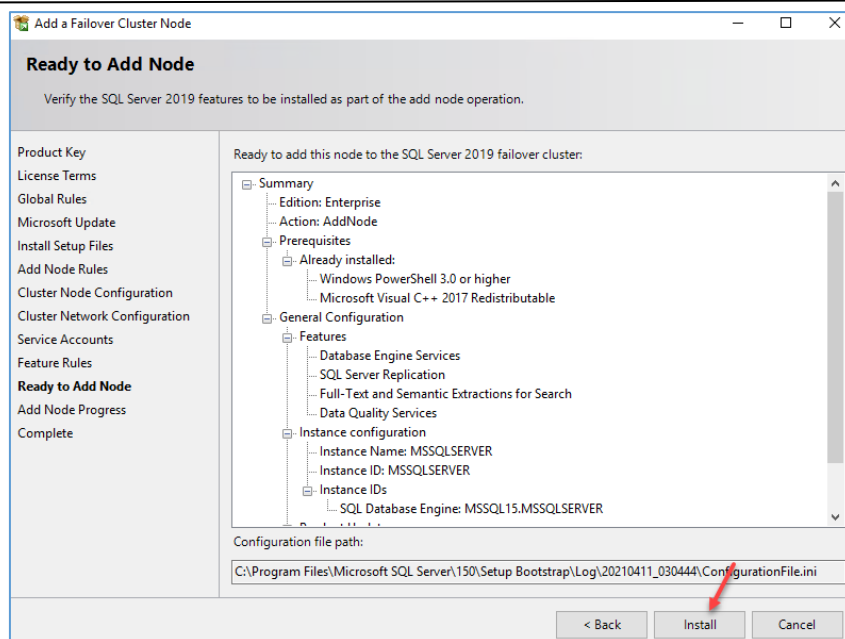
شکل ۱۱-۶۵

در شکل ۱۱-۶۶ هم باید نام کاربری که قرار است سرویس‌های مورد نظر را اجرا کنید وارد کنید، سعی کنید همان کاربری که در Failover-2 وارد کردید در این قسمت هم وارد کنید.



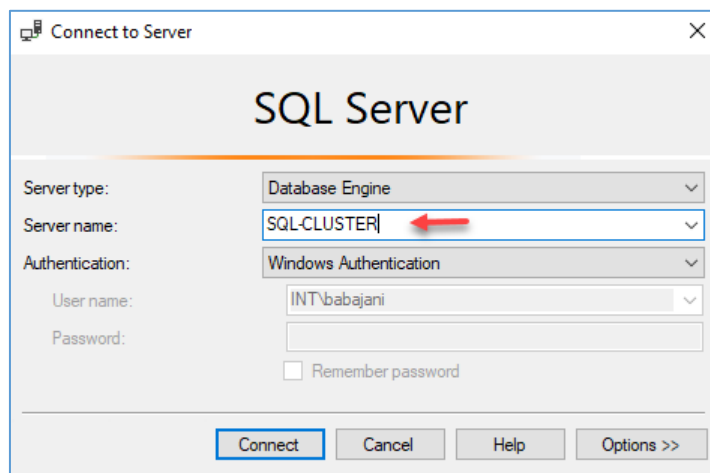
شکل ۱۱-۶۶

در شکل ۱۱-۶۷ بر روی Install کلیک کنید تا کار نصب آغاز شود.



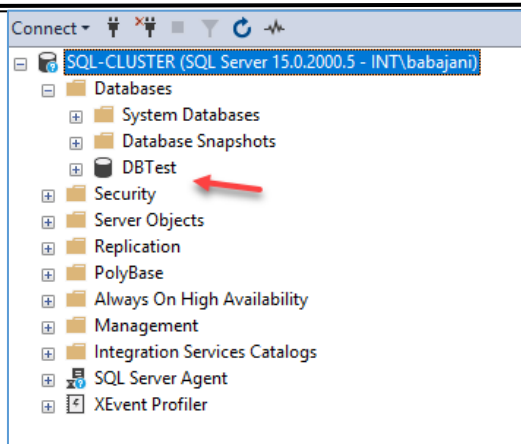
شکل ۱۱-۶۷

خوب برای تست عملکرد Clustering به مانند شکل ۶۸-۱۱ نرم افزار SQL Server Management Studio را اجرا کنید و در قسمت Server name باید همان نام شبکه که در مراحل راه اندازی SQL Clustering وارد کردید در این قسمت هم وارد کنید.



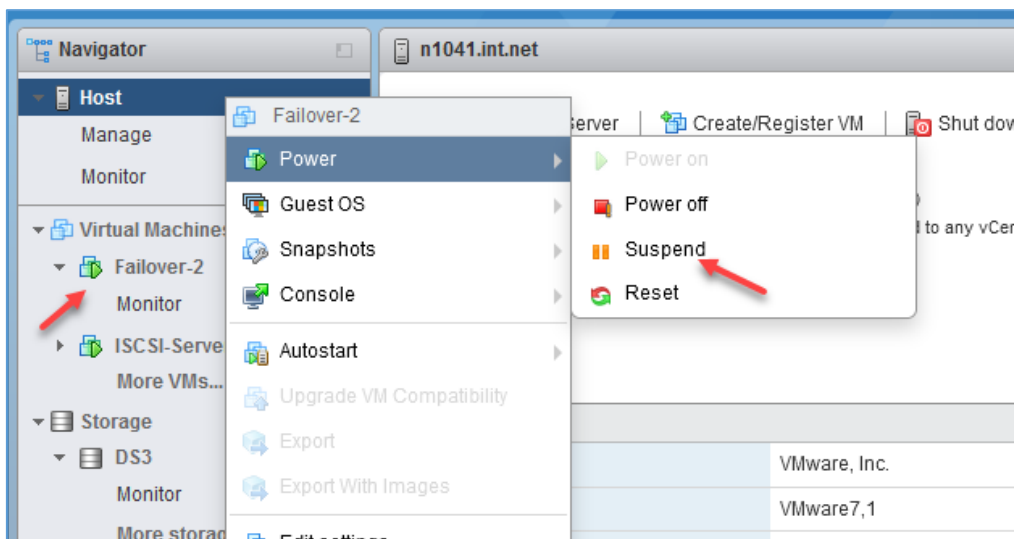
شکل ۱۱-۶۸

در شکل ۶۹-۱۱ به عنوان تست یک دیتابیس در سرور ایجاد کردیم و می خواهیم ببینیم زمانی که یکی از این سرورهای Failover را از دست دهیم چه اتفاقی رخ می دهد.



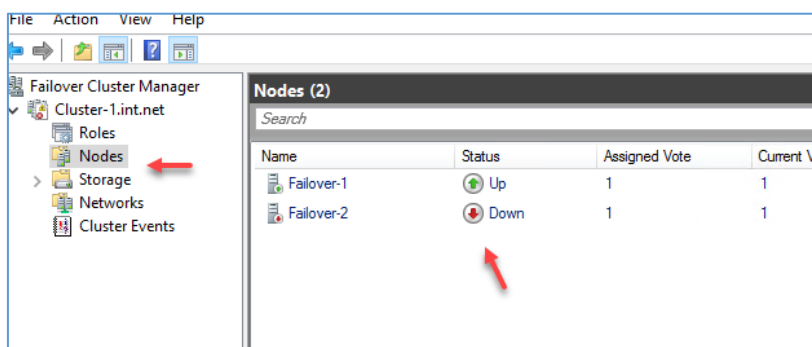
شکل ۱۱-۶۹

در شکل ۱۱-۷۰ سرور Failover-2 را که سرور اصلی در کلاسترینگ بوده را خاموش و یا Suspend می‌کنیم.



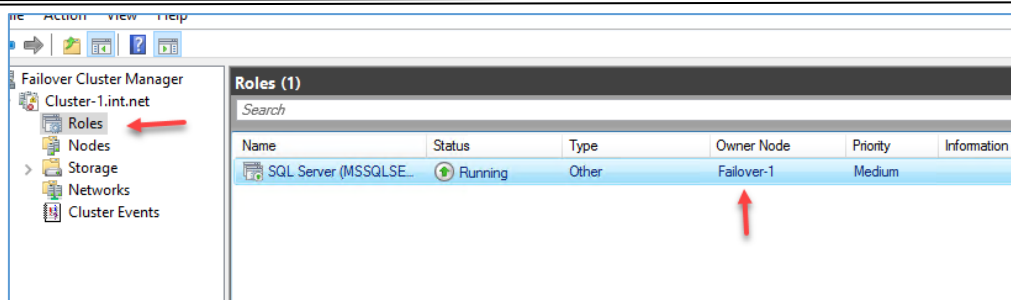
شکل ۱۱-۷۰

اگر به‌مانند شکل ۱۱-۷۱ وارد سرویس Clustering شوید و در قسمت Node مشاهده خواهید کرد سرور Failover-2 از کار افتاده.



شکل ۱۱-۷۱

در شکل ۱۱-۷۲ وارد قسمت Roles شوید و همان‌طور که مشاهده می‌کنید در ستون Owner node سرور تغییر کرده و Failover-1 فعال است.



شکل ۱۱-۷۲

خوب در این فصل توانستیم با کمک چهار سرور سرویس Failover Clustering را راه‌اندازی کنیم و توانایی آن را مشاهده کنیم، این نوع سرویس‌ها در سازمان‌هایی مانند بانک که قطع شدن یک لحظه از سیستم می‌تواند خسارت جبران‌ناپذیری وارد کند خیلی حیاتی و مهم است، پس شما به‌عنوان یک DBA باید نحوه‌ی راه‌اندازی این سرویس را کاملاً مسلط باشید.

فصل نهم

هوش تجاری – Business Intelligence

شما به عنوان متخصص SQL و برنامه‌نویس یک مجموعه عظیم قادر هستید از پیچیده‌ترین اطلاعات در داخل نرم‌افزارها با خبر شوید و مشکلات از نظر فنی نخواهید داشته باشید مثلاً می‌توانید وارد دیتابیس SQL شوید و از اطلاعات جدول آن با خبر شوید، ولی اگر مدیر آن سازمان بخواهد اطلاعات مورد نظر را مشاهده کند، به علت نداشتن دانش فنی لازم توانایی درک مسئله را نخواهد داشت که این یک مشکل بزرگ در مجموعه خواهد بود و برای حل مشکل باید راهکارهایی ارائه شود تا مدیران و افراد دیگر بتوانند اطلاعات را به راحتی مشاهده و تحلیل آن را بررسی کنند.

۹-۱ هوش تجاری چیست

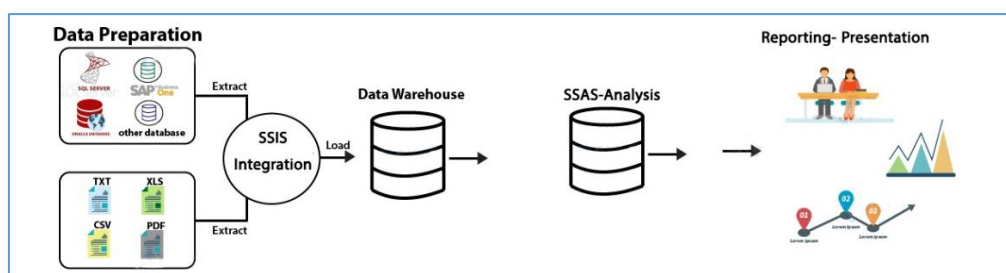
هوش تجاری به ابزارهایی گفته می‌شود که یک تحلیل کامل از اطلاعات سازمان مورد نظر انجام می‌دهد و نتیجه آن را در خروجی با اشکال مختلف نمایش می‌دهد و طبق آن نتایج، تصمیم‌گیری‌هایی برای پیشرفت سازمان گرفته خواهد شد که البته اگر داده‌ها و تحلیل‌ها اشتباه باشد سازمان ضرر سنگی خواهد کرد.

۹-۲ چگونه کار را آغاز کنیم

برای شروع کار باید یک سری موارد را آماده کنید.

- ۱- اولین قدم این است که هدف‌های پیش روی خود را بشناسیم و طبق آن عمل کنیم تا بتوانیم داده‌های درست را از آن استخراج کنیم و بعد بتوانیم از این شاخص‌ها برای دستیابی به هدفمان استفاده کنیم.
- ۲- قدم دوم این است که با استفاده از موارد شماره‌ی یک بتوانیم طراحی خوبی در قسمت داشبوردهای BI یا همان Business Intelligence داشته باشیم.

خوب در ادامه کار می‌خواهیم به طور کلی عملیات هوش تجاری را در سازمان بیان کنیم، اگر به شکل ۹-۱ توجه کنید، هوش تجاری در چند مرحله به هم وابسته انجام می‌شود که آنها را با هم بررسی می‌کنیم.



شکل ۹-۱ هوش تجاری

فراخوانی داده‌ها از منابع

داده‌های خام از منابع اطلاعاتی مختلفی جمع‌آوری شده و در لایه SSIS وارد می‌شوند. منبع داده خام می‌تواند یک سیستم ERP، پایگاه داده یک برنامه کاربردی یا یک فایل Excel باشد. در اصل، یک پایگاه داده تحلیلی باید امکان دسترسی و پردازش انواع منابع داده اعم از ساخت‌یافته (مانند پایگاه‌های داده)، نیمه ساخت‌یافته (مانند فایل‌های Excel) یا غیر ساخت‌یافته (مانند فایل‌های متنی یا صفحات اینترنتی) را داشته باشد.

جمع‌آوری اطلاعات خام توسط سرویس SSIS انجام می‌شود که خود شامل دو قسمت زیر است:

ETL مخفف Extract-Transformation-Load که رویکرد سه‌لایه دارد که به ترتیب Staging, Integration و Presentation نام دارد. در این رویکرد، داده‌ها قبل از فراخوانی به پایگاه داده تحلیلی (Data warehouse)، ساختار مناسب پیدا می‌کنند و سپس ذخیره می‌شوند.

ELT مخفف Extract-Load-Transformation که بدون توجه به روال سه‌لایه ETL، داده‌ها را مستقیماً به پایگاه داده‌های تحلیلی فراخوانی می‌کند و تغییر ساختار داده‌ها با توجه به نیاز انجام و در جداول جدید ذخیره می‌شود. پایگاه داده تحلیلی یا همان Data warehouse یک بانک اطلاعاتی بزرگ است که از طریق آن کلیه داده‌های حال و گذشته یک سازمان جهت انجام عملیات گزارش‌گیری و آنالیز در دسترس مدیران قرار می‌گیرد. پایگاه داده تحلیلی که برخی آن را انبار داده می‌خوانند، نقش مهمی در تصمیم‌گیری مدیران سازمان‌ها برای تعیین یک استراتژی موفق دارد. پایگاه داده نقش محوری در سامانه‌های هوش تجاری در سازمان‌ها ایفا می‌کنند.

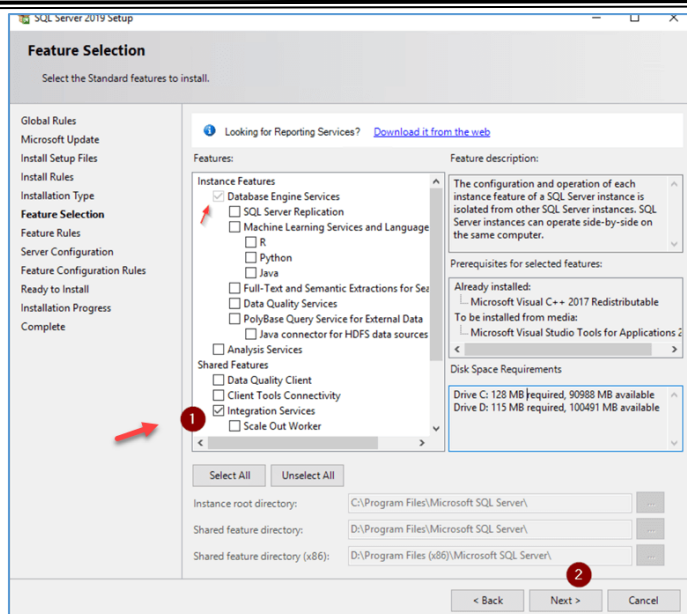
در لایه Presentation یا Reports داده‌ها در دسترس کاربران قرار می‌گیرد. نرم‌افزارهای تهیه گزارش مانند Cognos, SAP, Business Object، با دسترسی به این لایه می‌توانند اطلاعات موردنیاز مدیران و تحلیلگران را استخراج و در قالب گزارش یا Dashboard عرضه نمایند، البته بهترین نرم‌افزار در این زمینه Power BI است. در ادامه کار می‌خواهیم نحوه‌ی نصب و راه‌اندازی آن را بررسی کنیم.

۳-۹ مقدمه‌ای بر SSIS

Microsoft SQL Server Integration Services با نام SQL Integration Service یا SSIS شناخته می‌شوند. SSIS ابزاری برای هوش تجاری است که راه‌حل‌های تبدیل داده را برای سازمان‌های مختلف فراهم می‌کند. SQL Server Integration Services یک ابزار ETL (Extract, Transform و Load) است. این بدین معنی است که می‌توان از SSIS برای استخراج داده‌ها از طیف گسترده‌ای از منابع مانند فایل‌های اکسل، فایل‌های Flat، پرونده‌های XML، پایگاه‌های داده رابطه‌ای و غیره استفاده کرد. علاوه بر این، آنها را مطابق با نیاز خود تبدیل کنید و در آخر داده‌ها را در مقصد بارگیری کنید. IS یک ابزار هوشمند تجاری است که راه‌حل‌های تبدیل داده را برای سازمان‌های مختلف فراهم می‌کند.

۴-۹ نصب و راه‌اندازی سرویس SSIS

برای نصب و راه‌اندازی سرویس SSIS نیاز داریم تا در نرم‌افزار SQL Server سرویس Integration Services را انتخاب کنید که این موضوع را در شکل ۲-۹ مشاهده می‌کنید.

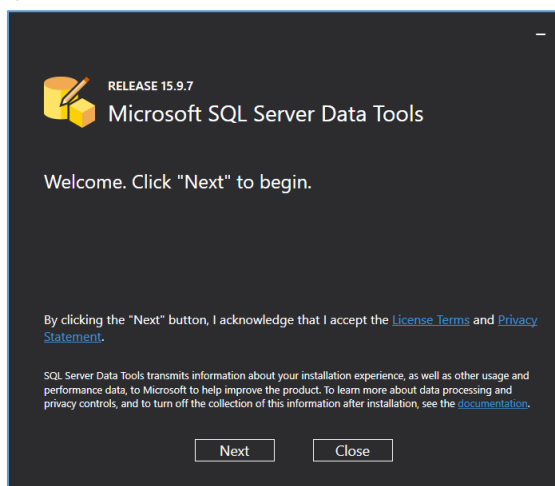


شکل ۹-۲ نصب سرویس Integration

بعد از اینکه سرویس Integration را بر روی سرور SQL نصب کردید در ادامه باید به صورت زیر عمل کنید: برای اینکه این سرویس SSIS را به صورت کامل راه اندازی کنیم نیاز به نرم افزار Visual Studio استفاده داریم، برای دانلود و نصب آن از طریق لینک زیر فایل مربوط به Visual Studio 2017 را که برای این کار طراحی شده است را دانلود کنید.

<https://go.microsoft.com/fwlink/?linkid=2161172&clid=0x409>

همان طور که در شکل ۹-۳ مشاهده می کنید فایل نصبی اجرا شده است و برای شروع باید بر روی Next کلیک کنید.



شکل ۹-۳ نصب SQL Server Data Tools

در شکل ۹-۴ باید یک Instance name برای این سرویس مشخص و بر روی Install کلیک کنید.

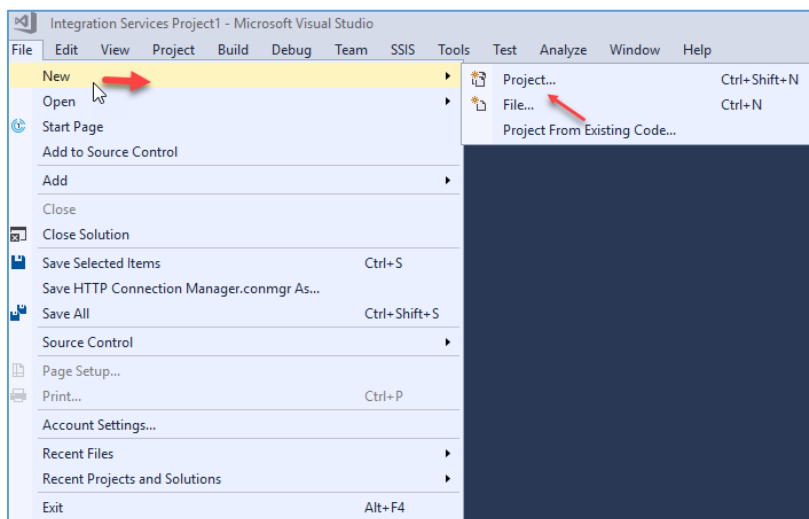


شکل ۹-۴ نصب SQL Server Data Tools

بعد از نصب می‌توانید (SSDT) Visual Studio 2017 را اجرا کنید.

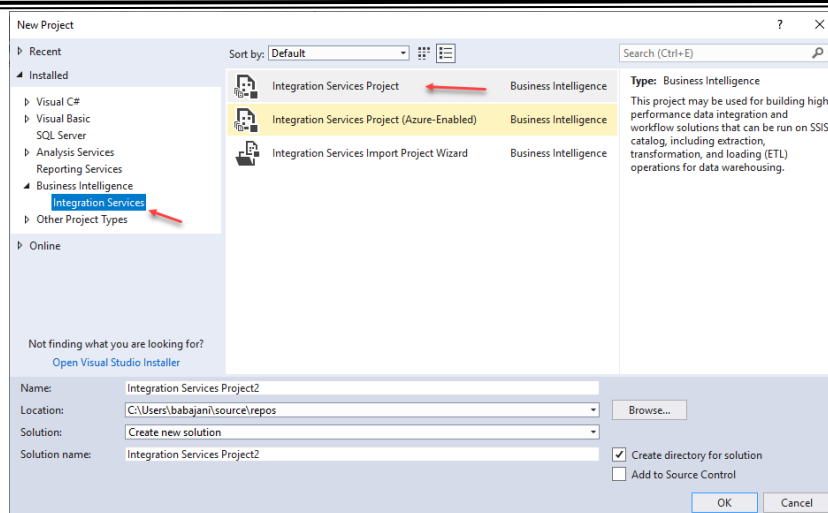
۹-۵ ایجاد پروژه SSIS

برای شروع کار نرم‌افزار Visual Studio 2017 را باز کنید و به‌مانند شکل ۹-۵ از منوی File و از قسمت New گزینه‌ی Project را انتخاب کنید.



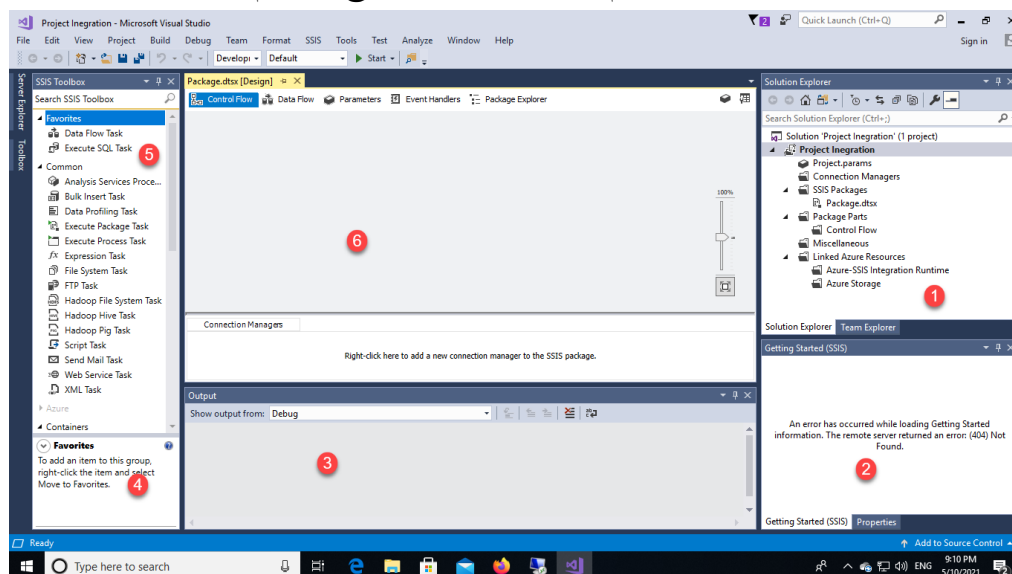
شکل ۹-۵ ایجاد پروژه

در ادامه و در شکل ۹-۶ از سمت چپ گزینه‌ی Integration Services را انتخاب کنید و در صفحه باز شده بر روی Integration Services Project را انتخاب کنید، در قسمت پایین شکل می‌توانید نام پروژه خود را مشخص و مسیر ذخیره‌سازی آن را مشخص کنید.



شکل ۶-۹ ایجاد پروژه

همان‌طور که در شکل ۷-۹ مشاهده می‌کنید نمای کلی پروژه Integration در Visual Studio باز شده است که برای درک بهتر محیط آن شماره‌گذاری کردیم و در ادامه آنها را توضیح می‌دهیم.



شکل ۷-۹

- ۱- Solution Explorer
در این قسمت ترکیبی از اطلاعات پروژه، فایل‌هایی که قرار است در پروژه استفاده شوند و کانکشن‌های مورد نظر به فایل‌های خارجی، مانند ارتباط با یک فایل اکسل.
- ۲- Properties
در این قسمت جزئیات اشیایی که انتخاب می‌کنید نمایش داده خواهد شد.
- ۳- Output
در این قسمت اطلاعاتی که در زمان اجرا پروژه نیاز است نمایش داده خواهد شد.
- ۴- Favorites
در این قسمت آیتم‌هایی که بیشتر از آنها استفاده می‌کنید نمایش داده خواهد شد.
- ۵- Toolbox

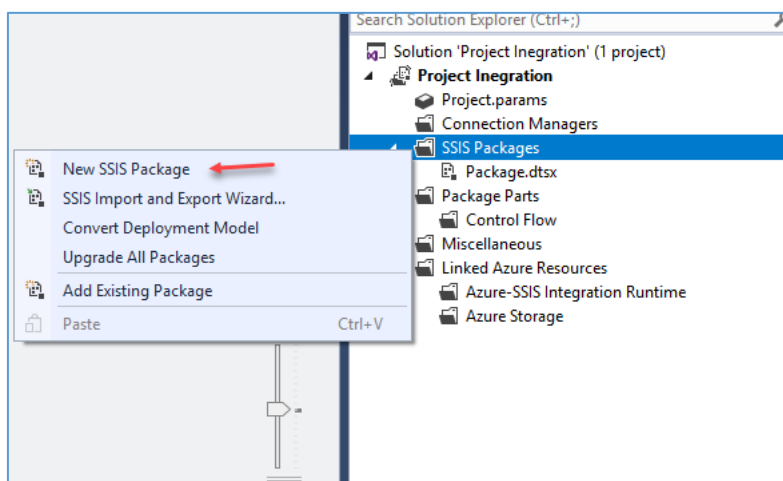
جعبه ابزار SSIS بسیاری از کارهای داخلی، کانتینرها، تحولات، منابع، مقصد و کارهای اداری را برای حل مشکلات پیچیده تجارت فراهم می‌کند. با کشیدن و رهاکردن آن کارها در محیط کار، از این ابزارهای گرافیکی SSIS استفاده کنید. این بدان معناست که برای انجام بیشتر عملیات نیازی به نوشتن یک خط کد واحد نیست.

۶- Package

در بخش Package چندین تب وجود دارد که برای طراحی پروژه کاربرد دارد و می‌توانید Taskهای مورد نظر خود را بر روی آن قرار دهید و اجرا کنید.

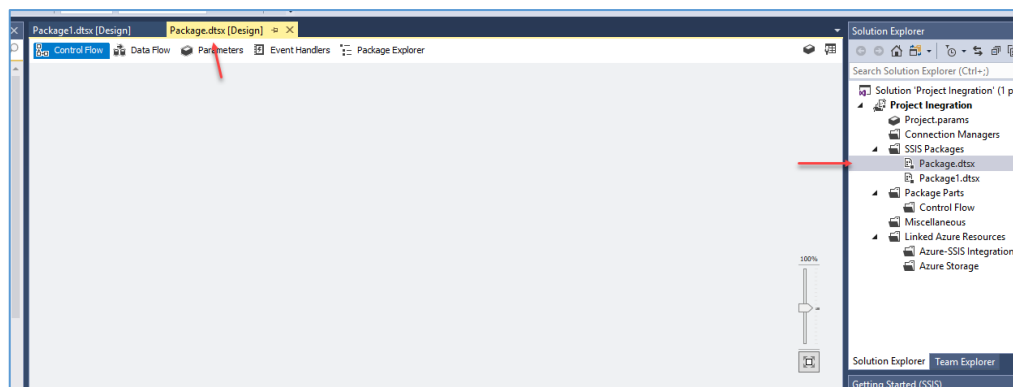
۱-۵-۹ بررسی Package در SSIS

زمانی که یک پروژه جدید ایجاد می‌کنید به صورت خودکار در قسمت SSIS Packages یک نام با عنوان Package.dtsx ایجاد می‌شود که تمام اطلاعات پروژه در آن قرار می‌گیرد و اگر بخواهید خودتان یک Package جدید ایجاد کنید باید به‌مانند شکل ۸-۹ بر روی SSIS Packages کلیک راست کنید و گزینه New SSIS Package را انتخاب کنید.



شکل ۸-۹ ایجاد Package

اگر به شکل ۹-۹ توجه کنید یک Package جدید اضافه شده است اگر دو بار بر روی هر یک از پکیج‌ها کلیک کنید اطلاعات مربوط به همان پکیج نمایش داده خواهد شد و هر پکیج کار مختص به خودش را انجام می‌دهد، توجه داشته باشید که هر Package دارای پنج تب مختلف است، یک Package باید حداقل شامل یک Control flow باشد.

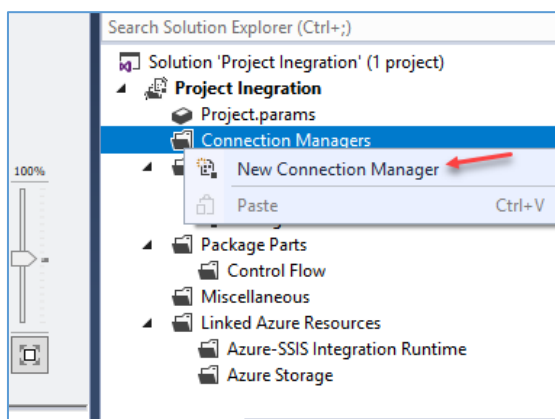


شکل ۹-۹ ایجاد Package

۹-۵-۲ بررسی Connection Manager

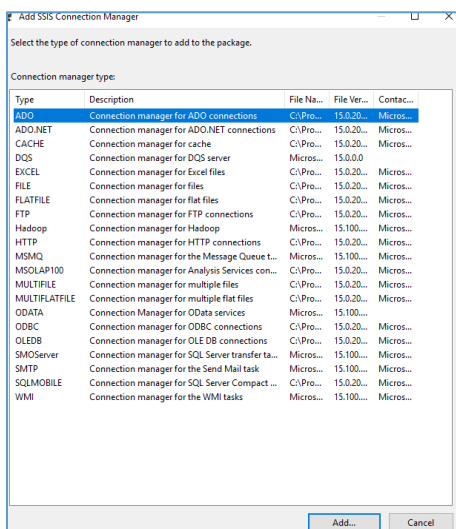
قسمت دیگری که در پروژه‌های SSIS وجود دارد، Connection Manager است، این قسمت برای انتقال داده‌ها از منابع خاص به داخل پروژه کاربرد دارد مثلاً می‌توانید یک فایل اکسل را به پروژه متصل کنید و از داده‌های آن استفاده کنید و یا اینکه یک سرور ایمیل را تنظیم کنید تا در پروژه بتوانیم از آن استفاده کنیم، مثلاً بعد از انجام کار یک ایمیل تأیید برای شما ارسال کند.

برای اینکه از گزینه‌های موجود در Connection Manager مطلع شویم، به‌مانند شکل ۹-۱۰ بر روی Connection Managers کلیک راست کنید و گزینه‌ی New Connection Manager را انتخاب کنید.



شکل ۹-۱۰ ایجاد Connection Manager

همان‌طور که در شکل ۹-۱۱ مشاهده می‌کنید چندین گزینه‌ی مختلف وجود دارد که با استفاده از آنها می‌توانید به منابع مختلف و ارتباطات گوناگون دست پیدا کنید، در



شکل ۹-۱۱ ایجاد Connection Manager

جدول ۹-۱ بررسی Connection Manager

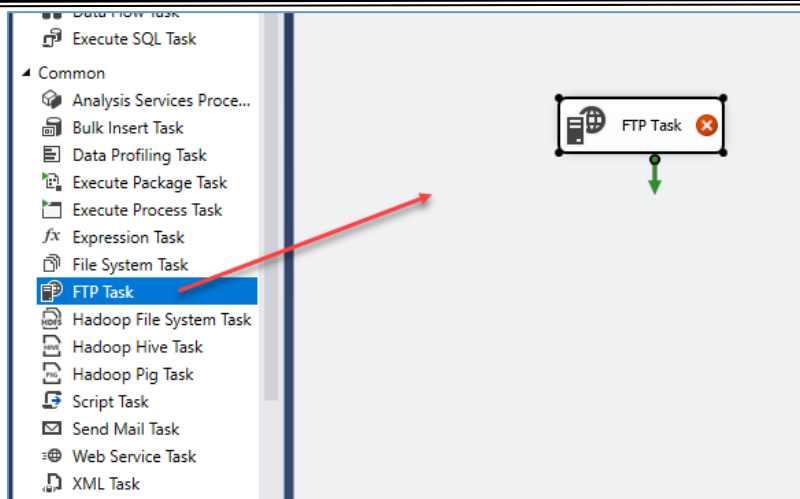
توضیحات	نوع ارتباط
این Connection Manager برای ارتباط با ActiveX Data Objects استفاده می‌شود. یا می‌توان آن را یک شی ADO نامید.	ADO

این برای اتصال به منبع داده با استفاده از یک ارائه‌دهنده دات نت استفاده می‌شود.	ADO.NET
داده‌های مربوط به جریان داده یا یک فایل حافظه نهان (.caw) را می‌خواند و می‌تواند داده‌ها را در پرونده حافظه نهان ذخیره کند.	CACHE
به یک سرور Data Quality Services و یک پایگاه داده Data Quality Services در سرور متصل می‌شود.	DQS
به یک فایل اکسل متصل می‌شود.	EXCEL
به یک فایل یا پوشه متصل می‌شود.	FILE
به یک فایل FLAT متصل می‌شود.	FLATFILE
برای ارتباط با یک سرور FTP کاربرد دارد.	FTP
برای ارتباط با Web Server طراحی شده است.	HTTP
به صف پیام متصل می‌شود.	MSMQ
به نمونه‌ای از SQL Server Analysis Services یا یک پروژه Analysis Services متصل می‌شود.	MSOLAP100
به چندین پرونده و پوشه متصل می‌شود.	MULTIFILE
به چندین پرونده داده و پوشه متصل می‌شود.	MULTIFLATFILE
با استفاده از ارائه‌دهنده OLE DB به منبع داده متصل می‌شود.	OLE DB
با استفاده از ODBC به منبع داده متصل می‌شود.	ODBC
به سرور SQL Server Management Objects (SMO) متصل می‌شود.	SMOSERVER
به سرور نامه SMTP متصل می‌شود.	SMTP
به یک پایگاه داده SQL Server Compact متصل می‌شود.	SQL MOBILE
به یک سرور متصل می‌شود و دامنه مدیریت Windows Management Instrumentation (WMI) در سرور را مشخص می‌کند.	WMI

تا به اینجای کار یک پروژه جدید ایجاد کردیم و محیط و ابزارهای آن را بررسی کردیم در ادامه می‌خواهیم با تعریف مثال‌های گوناگون نحوه کار سرویس SSIS را بررسی کنیم.

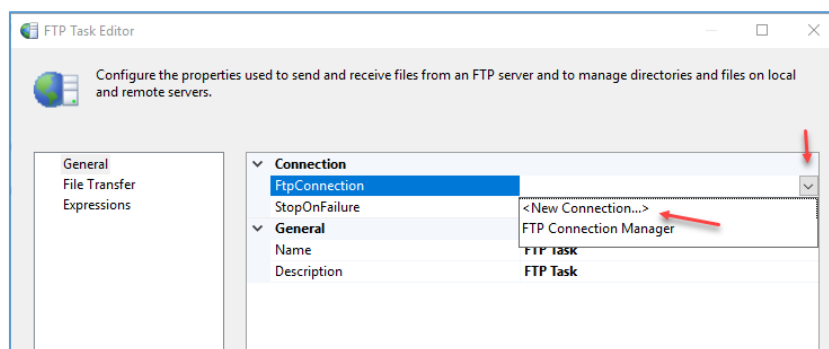
۳-۵-۹ استفاده از FTP در SSIS

یکی از امکاناتی که در SSIS وجود دارد این است که شما می‌توانید، فایل مورد نظر خود را به یک سرور FTP ارسال کنید، برای این کار به مانند شکل ۱۲-۹ ابزار FTP Task را به لیست اضافه و بر روی آن دو بار کلیک کنید.



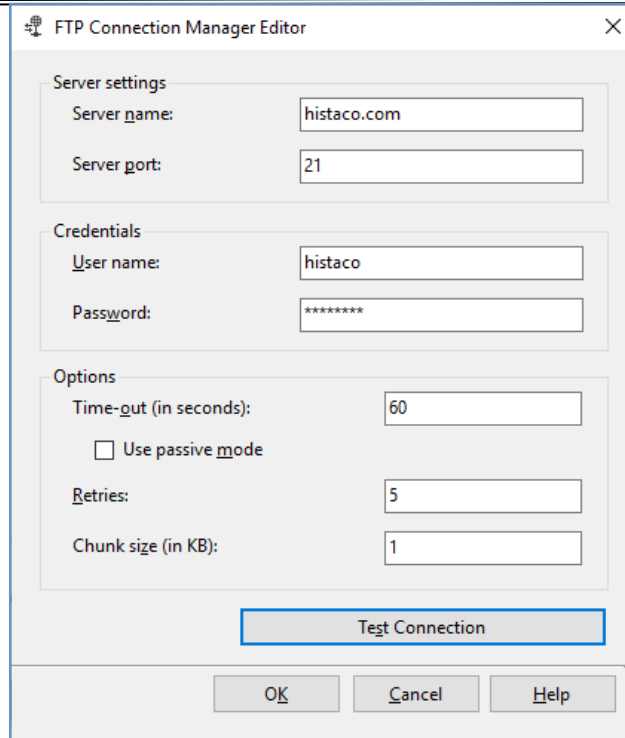
شکل ۹-۱۲

در شکل ۹-۱۳ و از قسمت FtpConnection گزینه‌ی New Connection را انتخاب کنید.



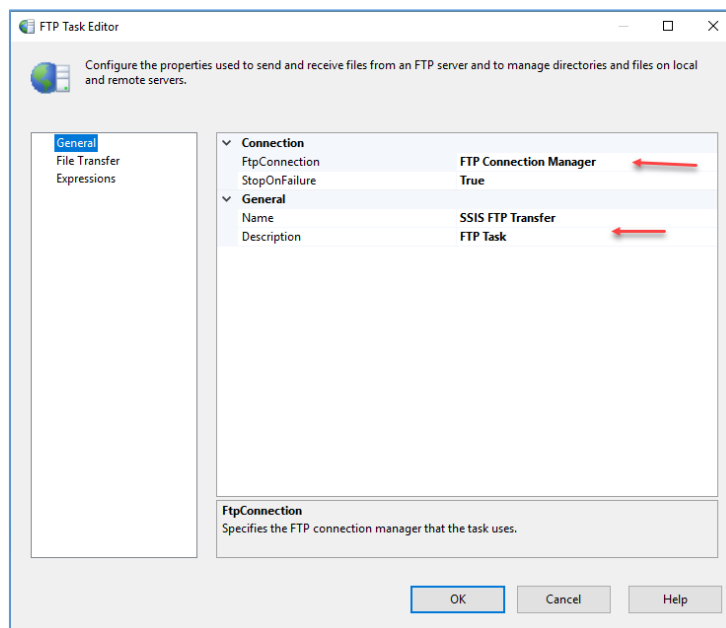
شکل ۹-۱۳ ایجاد FTP Connection

در شکل ۹-۱۴ و در قسمت Servername باید آدرس سرور FTP خود را وارد کنید، در قسمت پورت هم اگر تغییری در سرور ایجاد نکرده باشید همان پورت ۲۱ را که پورت پیش‌فرض است را وارد کنید و در قسمت Username و Password نام کاربری و رمز عبور را وارد و بر روی OK کلیک کنید، البته قبل از آن می‌توانید بر روی TestConnection کلیک کنید و ارتباط با سرور FTP را تست کنید.



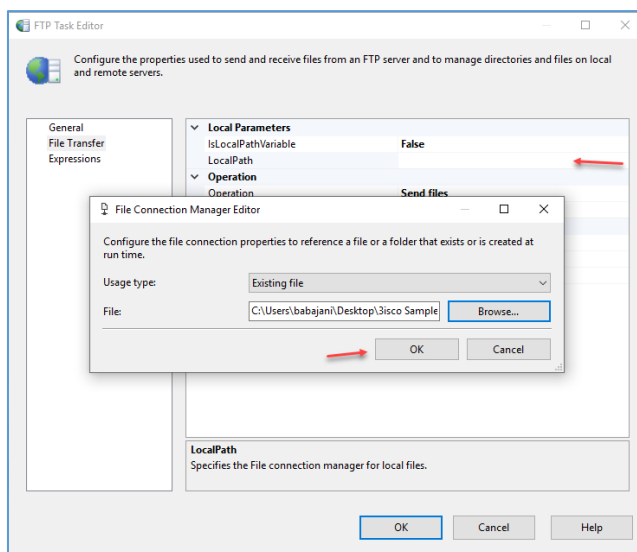
شکل ۹-۱۴ تنظیمات FTP

در شکل ۹-۱۵ بعد از اضافه کردن Connection باید نام و توضیحات دلخواه خود را وارد کنید.



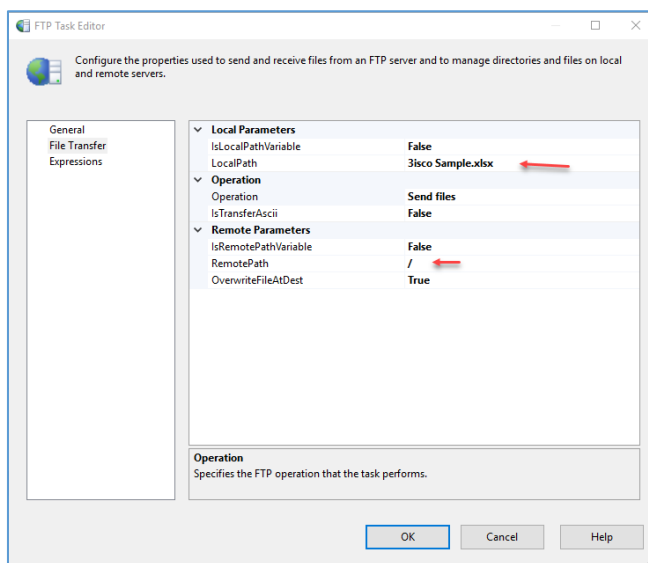
شکل ۹-۱۵ تنظیمات FTP

در شکل ۹-۱۶ باید در قسمت LocalPath بر روی New کلیک کنید و یک فایل به دلخواه از سیستم خود انتخاب کنید که در اینجا یک فایل اکسل انتخاب شده است.



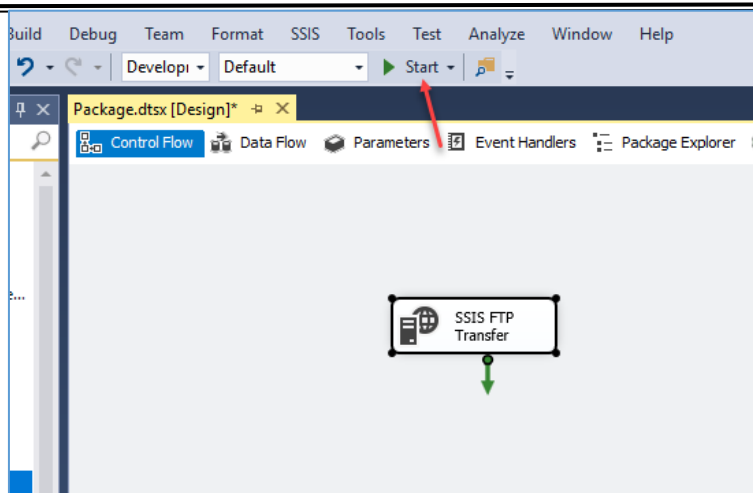
شکل ۹-۱۶ تنظیمات FTP

در شکل ۹-۱۷ بعد از معرفی فایل برای انتقال، باید در قسمت Operation نوع عملیات را مشخص کنید و در قسمت RemotePath با قرار دادن علامت "/" مشخص می‌کنید که فایل در ریشه کیپی شود.



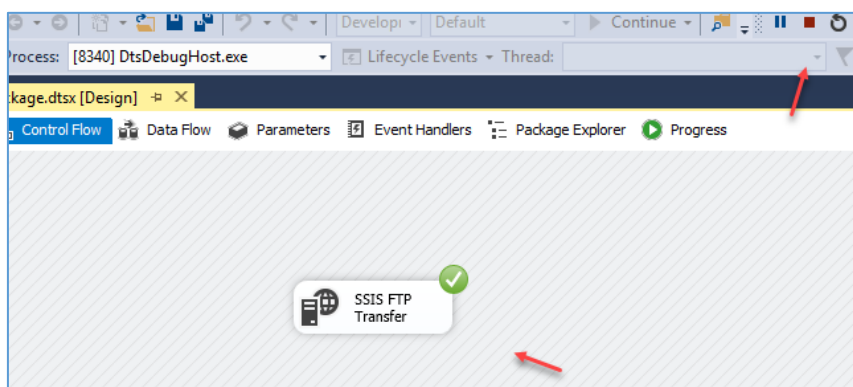
شکل ۹-۱۷ تنظیمات FTP

بعد از اینکه تنظیمات را در قسمت قبل به درستی انجام دادید به مانند شکل ۹-۱۷ بر روی دکمه Start کلیک کنید تا عملیات انتقال فایل از سیستم شما به سرور FTP از طریق SSIS انجام شود.

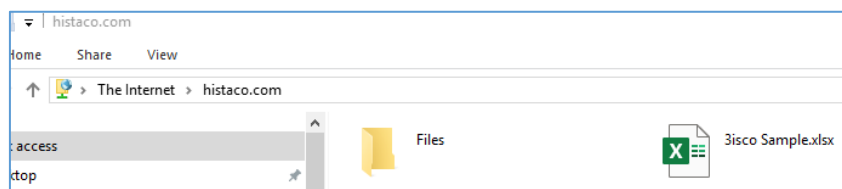


شکل ۹-۱۷ اجرای FTP

همانطور که در شکل ۹-۱۸ مشاهده می‌کنید FTP با موفقیت اجرا شده و در شکل ۹-۱۹ فایل مورد نظر در FTP کپی شده است.



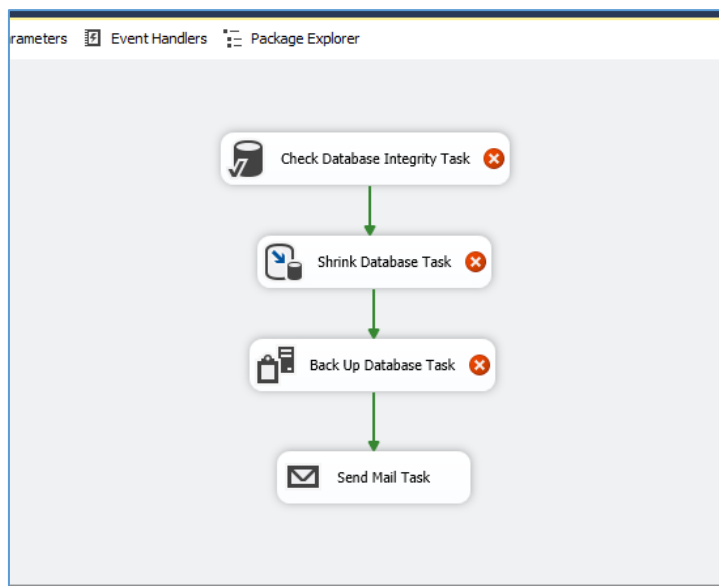
شکل ۹-۱۸ اجرای FTP



شکل ۹-۱۹ انتقال فایل FTP

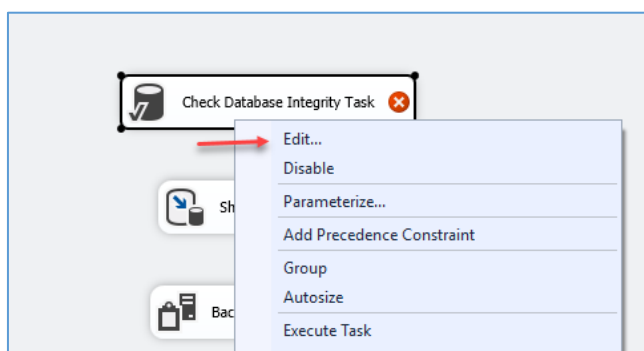
۴-۵-۹ کار با Integrity, Shrink, Backup, Email در SSIS

در سرویس SSIS شما می‌توانید دیتابیس‌های خودتان را از نظر سالم بودن (Integrity) بررسی کنید، حجم آن را کاهش دهید (Shrink)، یک پشتیبان کامل از تمام دیتابیس‌ها تهیه کنید و در آخر اگر کار به درستی انجام شد یک ایمیل هم به مدیر مورد نظر ارسال کنید، برای این کار به مانند شکل ۹-۲۰ چهار ابزار مورد نظر را به صفحه اضافه کنید و فلش‌هایی که در انتهای آنها قرار دارد را به ترتیب به هم متصل کنید، با این کار، کارها به ترتیب و پست سر هم اجرا خواهد شد.



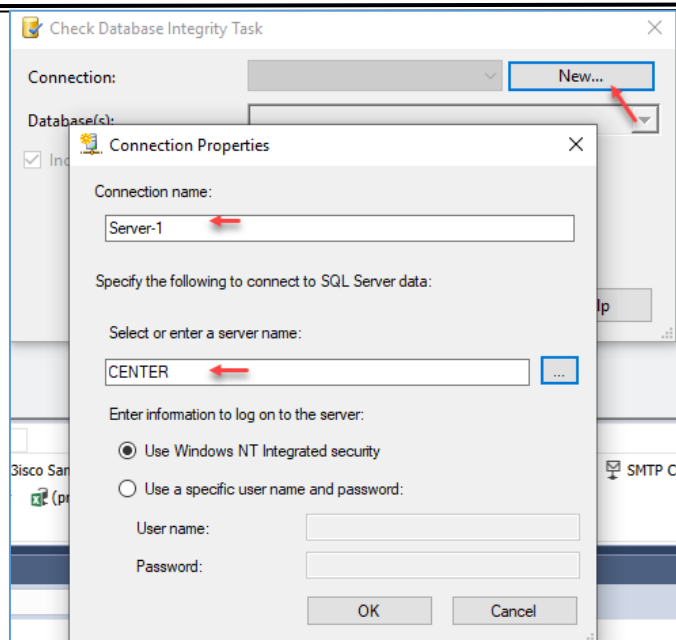
شکل ۹-۲۰ اجرای چهار مرحله

برای اینکه بتوانیم هر یک از ابزارها را تنظیم کنید به مانند شکل ۹-۲۱ بر روی آنها کلیک راست و گزینه‌ی Edit را انتخاب کنید.



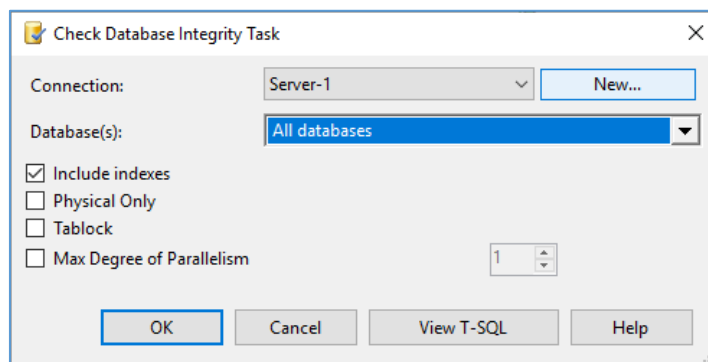
شکل ۹-۲۱ تنظیم Integrity

در شکل ۹-۲۲ باید بر روی New کلیک کنید و سروری که SQL بر روی آن نصب شده است را وارد کنید.



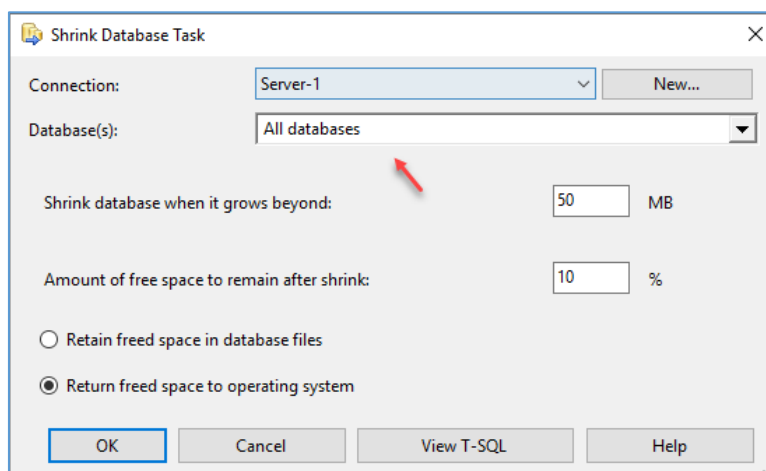
شکل ۹-۲۲ ارتباط با سرور SQL

در شکل ۹-۲۳ باید در قسمت Database گزینه‌ی All databases را انتخاب کنید و بر روی OK کلیک کنید.



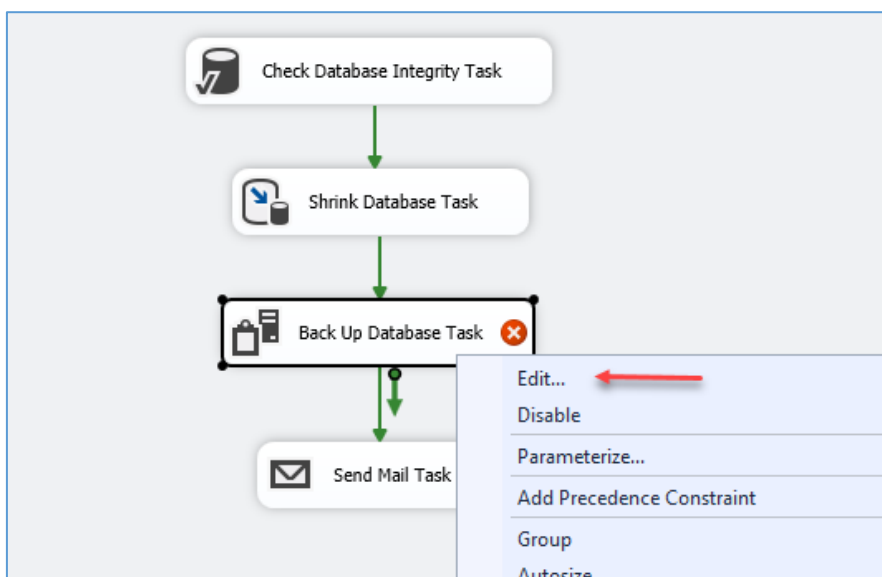
شکل ۹-۲۳ انتخاب دیتابیس

در قسمت دوم یعنی Sharink باید به مانند شکل ۹-۲۴ گزینه‌ی All databases را انتخاب و بر روی OK کلیک کنید.



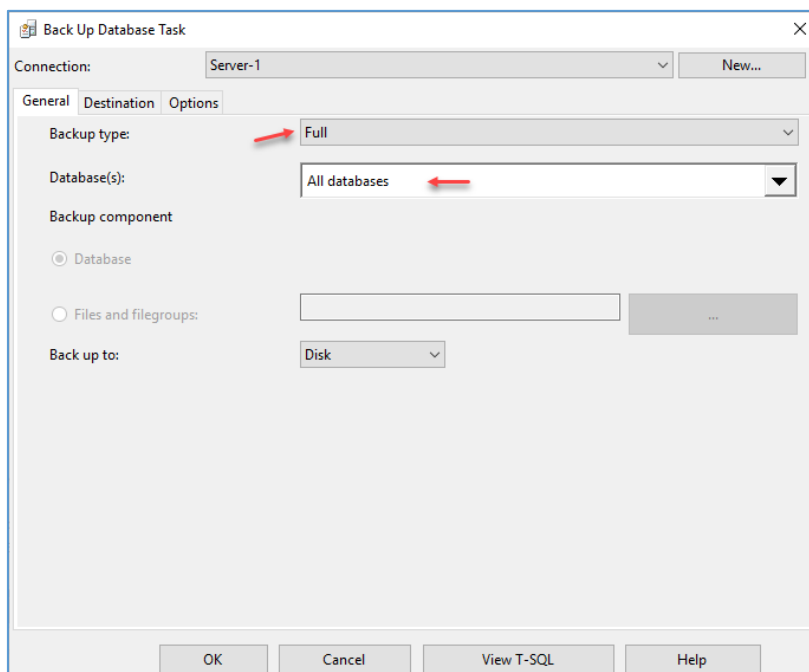
شکل ۹-۲۴ انتخاب دیتابیس

در ادامه کار به مانند شکل ۹-۲۵ بر روی Back up کلیک راست کنید و گزینه‌ی Edit را انتخاب کنید:



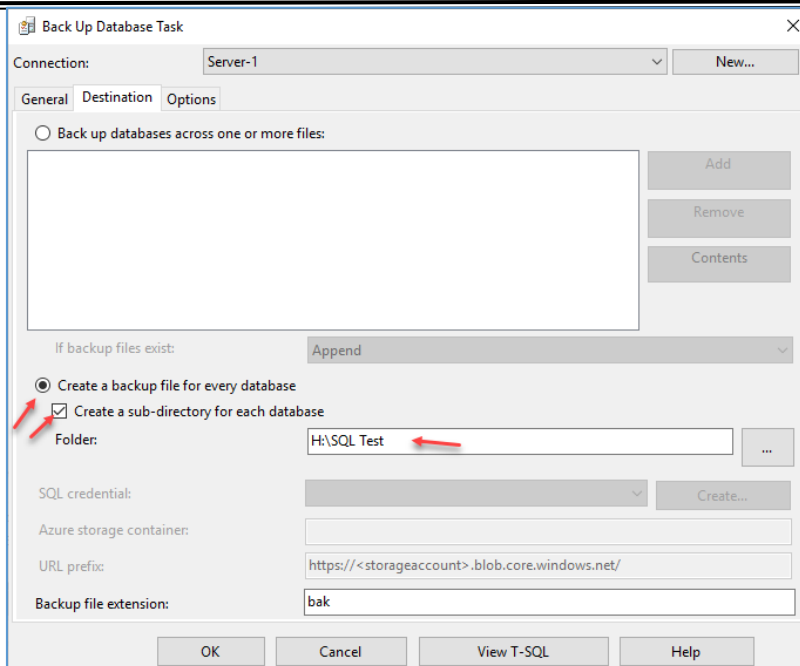
شکل ۹-۲۵ تنظیم Back up

در شکل ۹-۲۶ باید از قسمت Backup Type نوع پشتیبان را Full انتخاب کنید و در قسمت Database کل دیتابیس را انتخاب کنید و در ادامه کار وارد تب Destination شوید.



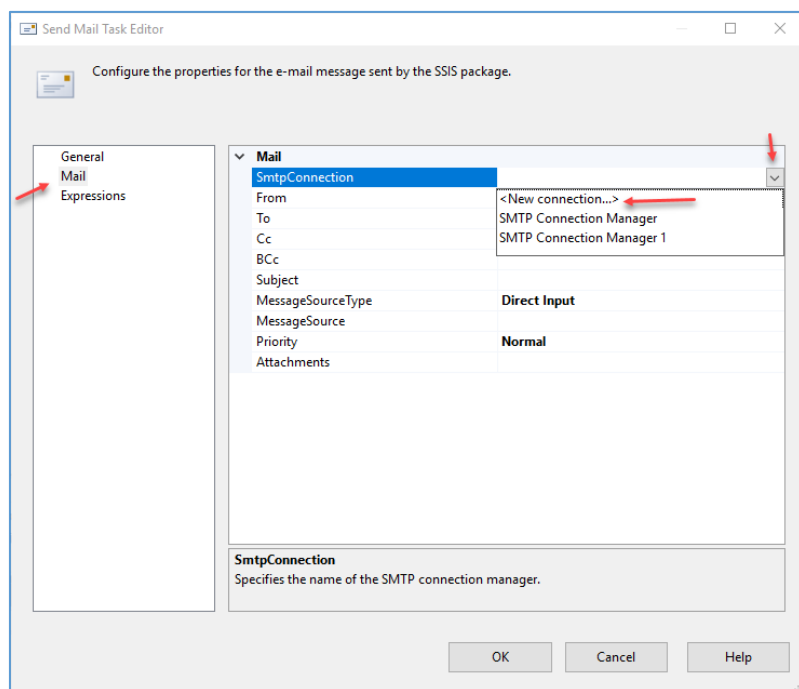
شکل ۹-۲۶ تنظیم Back up

به مانند شکل ۹-۲۷ باید مسیر مورد نظر برای کپی کردن فایل پشتیبان را مشخص کنید.



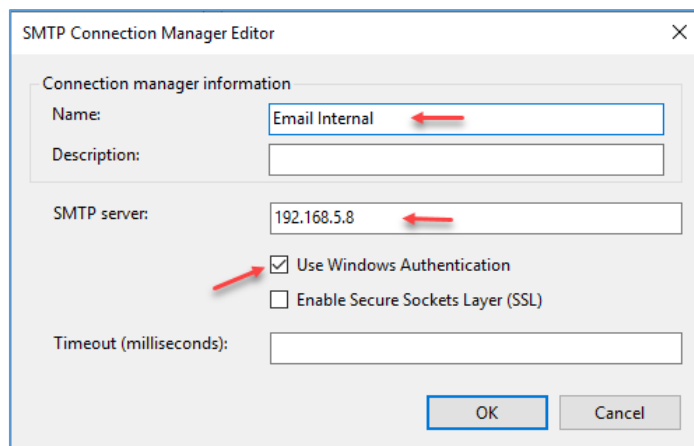
شکل ۹-۲۷ تنظیم Back up

در مرحله آخر باید ایمیل را تنظیم کنیم برای این کار به مانند شکل ۹-۲۸ وارد تب mail شوید و از قسمت SmtpConnection گزینه New connection را انتخاب کنید.



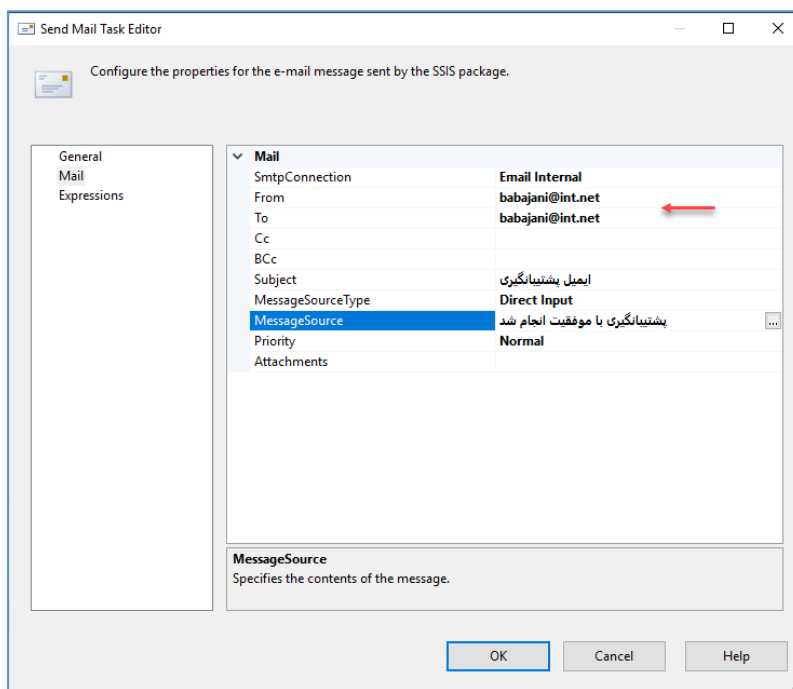
شکل ۹-۲۸ تنظیم Email

در شکل ۹-۲۹ نام به دلخواه وارد و آدرس سرور Email خود را وارد کنید، اگر تیک گزینه‌ی مورد نظر را انتخاب کنید از همان نام کاربری و رمز عبور شما که وارد ویندوز شده‌اید استفاده خواهد شد.



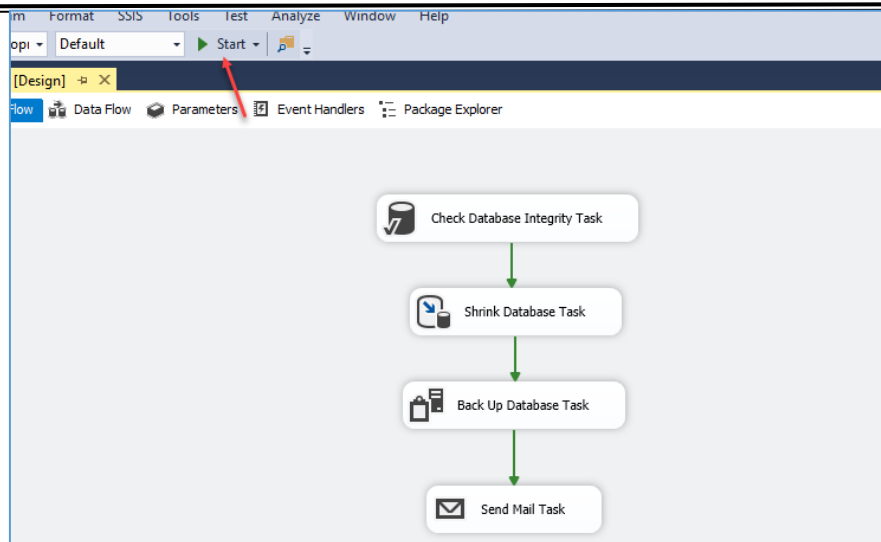
شکل ۹-۲۹ تنظیم Email

در شکل ۹-۳۰ و در قسمت From و To ایمیل مورد نظر خود را وارد کنید و در قسمت Subject عنوان ایمیل و در قسمت MessageSource متن ایمیل را وارد و بر روی OK کلیک کنید.



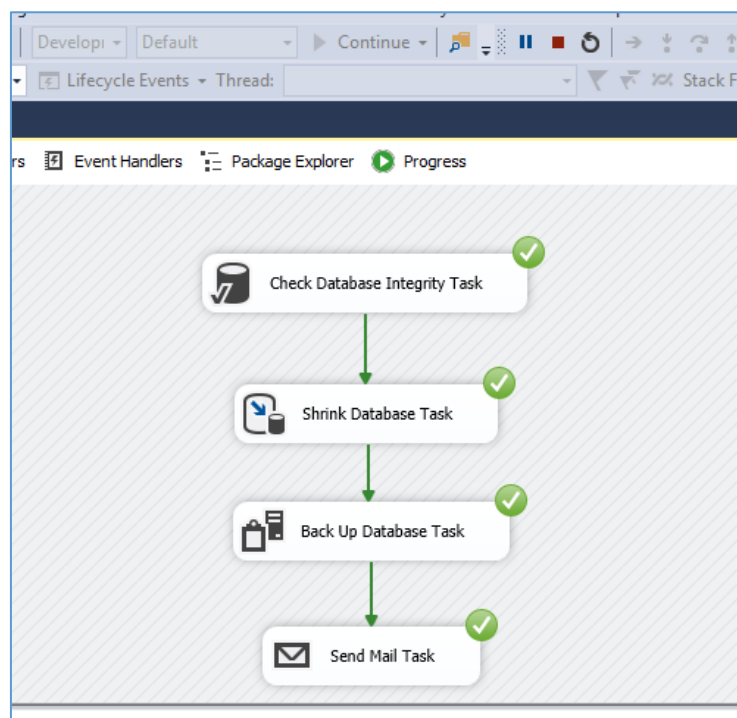
شکل ۹-۳۰ تنظیم ایمیل

بعد از تکمیل موارد بالا به مانند شکل ۹-۳۱ بر روی دکمه Start کلیک کنید تا عملیات آغاز شود.

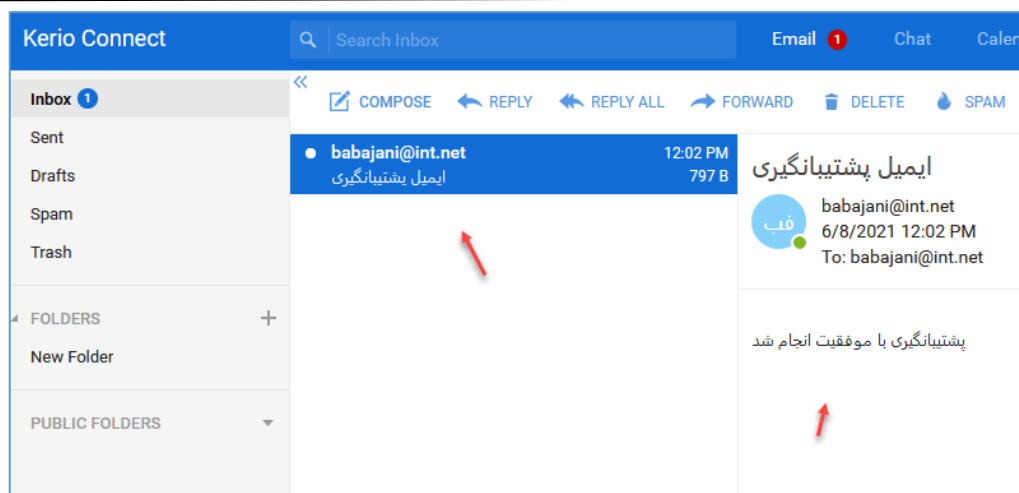


شکل ۹-۳۱ اجرای عملیات

همانطور که در شکل ۹-۳۲ مشاهده می‌کنید عملیات با موفقیت اجرا شده است و در شکل ۹-۳۳ هم ایمیل تایید ارسال شده است.



شکل ۹-۳۲ اجرای عملیات



شکل ۹-۳۳ تایید ایمیل

۹-۵-۵ گرفتن خروجی تصادفی از جداول با SSIS

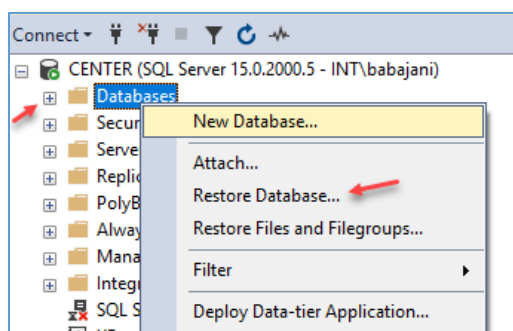
بعضی اوقات تعداد زیادی ردیف در جداول پایگاه داده داریم. ما می‌خواهیم محاسبات مربوط به داده‌ها را انجام دهیم، اما به تمام داده‌های جدول نیازی نداریم.

می‌توانید آن را مانند آزمایش خون در آزمایشگاه پزشکی درک کنید. لازم است شما یک نمونه از خون خود را بدهید تا آنها بتوانند آزمایش آزمایشگاهی را روی آن انجام دهند و به شما نتیجه بدهند. فرض کنید ما رکورد فروش مشتری داریم و شامل میلیون‌ها رکورد است، شما می‌خواهید روی آن داده‌ها محاسبات مورد نظر خود را انجام دهید. اگر بخواهید تمام آن داده‌ها را استفاده کنید، منابع زیادی از سیستم مصرف می‌شود و اجرای بسته نیز زمان‌بر است. ما مکانیزمی می‌خواهیم که باید ۲۰٪ رکورد را از جدول دریافت کند و می‌تواند از آن برای کارهای بیشتر استفاده شود. مثال دیگری که می‌توانید در نظر بگیرید سیستم قرعه‌کشی است. فرض کنید می‌خواهید ۲۰ مشتری را از کل مجموعه داده‌ها به طور تصادفی انتخاب کنید. ما می‌توانیم برای تحقق هدف خود از نمونه برداری داده در SSIS استفاده کنیم. برای انجام این کار می‌خواهیم یک پایگاه داده‌عظیم را به SQL Server اضافه کنیم و از طریق SSIS یک خروجی به دلخواه از آن بگیریم.

برای شروع از لینک زیر پایگاه داده‌ی AdventureWorks2019 را از طریق لینک زیر دانلود کنید:

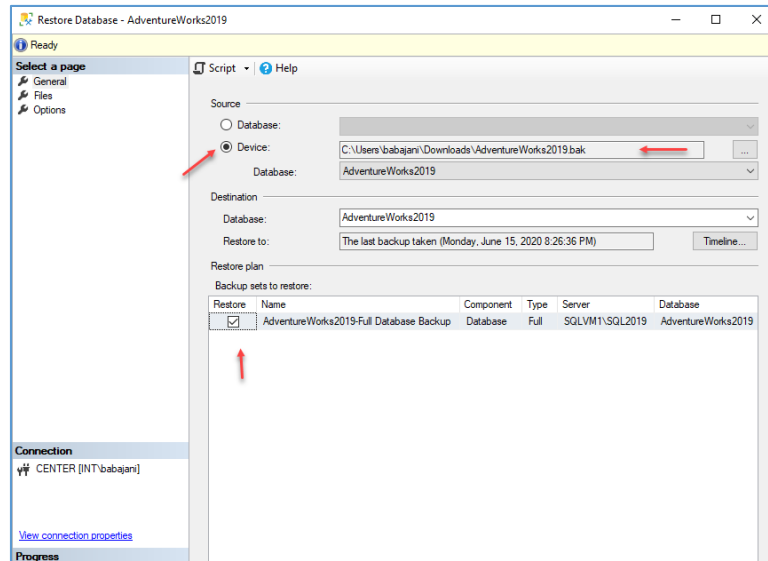
<https://github.com/Microsoft/sql-server-samples/releases/download/adventureworks/AdventureWorks2019.bak>

بعد از دانلود فایل مورد نظر باید آن را در SQL Restore کنید، برای این کار به مانند شکل ۹-۳۴ وارد SQL Server شوید و بر روی Databases کلیک راست کنید و گزینه‌ی Restore Database را انتخاب کنید.



شکل ۹-۳۴ Restore Database

به مانند شکل ۳۵-۹ باید گزینه‌ی Devices را انتخاب کنید و فایل مورد نظر را در مسیر مشخص شده انتخاب کنید و بعد بر روی OK کلیک کنید تا فایل مورد نظر به SQL اضافه شود تا بتوانیم در ادامه از این دیتابیس استفاده کنیم.



شکل ۳۵-۹ Restore Database

بعد از اضافه کردن دیتابیس می‌توانیم از طریق دستور زیر خروجی کلی از جدول Sales.Customer را مشاهده کنید.

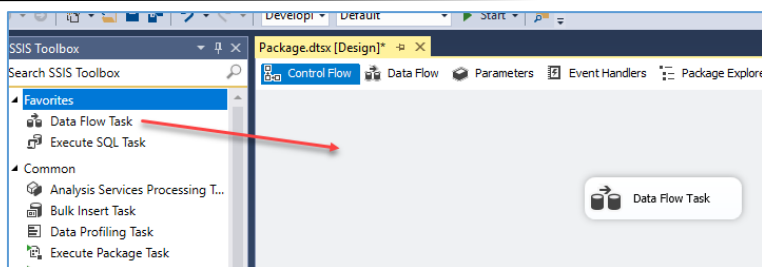
```
SELECT [CustomerID]
, [PersonID]
, [StoreID]
, [TerritoryID]
, [AccountNumber]
, [rowguid]
, [ModifiedDate]
FROM [AdventureWorks2019].[Sales].[Customer]
```

همانطور که در شکل ۳۶-۹ مشاهده می‌کنید، خروجی جدول Sales.Customer دارای 19820 سطر هست که در ادامه می‌خواهیم آن را محدودتر کنیم و در خروجی نمایش دهیم.

rowid	CustomerID	PersonID	StoreID	TerritoryID	AccountNumber	rowguid	ModifiedDate
7	7	NULL	930	1	AW00000006	1A02D288F92467D6B034FC82847F0D7	2014-09-12 11:15:07.263
8	8	NULL	1024	5	AW00000007	03E9273E-6193-448E-9823-FE0C44AED78	2014-09-12 11:15:07.263
9	9	NULL	620	5	AW00000008	80136881-4323-4BFA-8BEA-5B5B1E4BD4A0	2014-09-12 11:15:07.263
10	10	NULL	928	6	AW00000009	B900987F-23C3-481D-80DA-C45A48D6F772	2014-09-12 11:15:07.263
11	11	NULL	1022	6	AW00000010	CD86690D-2FF1-4E8A-8F22-68AD1D11DA8D	2014-09-12 11:15:07.263
12	12	NULL	622	6	AW00000011	750F3495-59C4-48A0-80E1-E37EC6E7709	2014-09-12 11:15:07.263
13	13	NULL	434	7	AW00000012	947BCAF1-1F32-44F3-89C3-0011F98FE5E4	2014-09-12 11:15:07.263
14	14	NULL	1020	8	AW00000013	B0FA5854-2511-4398-47AC-59C9C4608175	2014-09-12 11:15:07.263
15	15	NULL	624	9	AW00000014	2F968BDC-723D-468F-834B-B2B8AE79C849	2014-09-12 11:15:07.263
16	16	NULL	432	10	AW00000015	0340707B-D4FA-4795-93AA-CAE863718CF9	2014-09-12 11:15:07.263
17	17	NULL	1018	5	AW00000016	C3381589-D31C-4E5E-8978-8D3449E81F0F	2014-09-12 11:15:07.263
18	18	NULL	1332	3	AW00000017	34DB417F-1E0B-4408-9FF6-987E59D0D73C	2014-09-12 11:15:07.263
19	19	NULL	430	1	AW00000018	C04D6B4D-94C6-4C5C-A44C-B449C0AC1B45	2014-09-12 11:15:07.263
20	20	NULL	1016	1	AW00000019	69AE5D43-318E-4B76-BFBB-5A23C4788B8C	2014-09-12 11:15:07.263
21	21	NULL	1334	4	AW00000020	E010C10A-F1C3-48BA-81CA-A7E083350400	2014-09-12 11:15:07.263
22	22	NULL	494	3	AW00000021	564E0842-4609-43DE-9881-914D433D82D	2014-09-12 11:15:07.263
23	23	NULL	1014	4	AW00000022	9774AED6-D673-412D-B481-2573E470B478	2014-09-12 11:15:07.263
24	24	NULL	1336	4	AW00000023	733F8250-3251-4C2A-8C85-C2858B76B764	2014-09-12 11:15:07.263
25	25	NULL	1014	4	AW00000024	C111E51D-178D-40BD-86FF-F1CCBF4B1AAD	2014-09-12 11:15:07.263

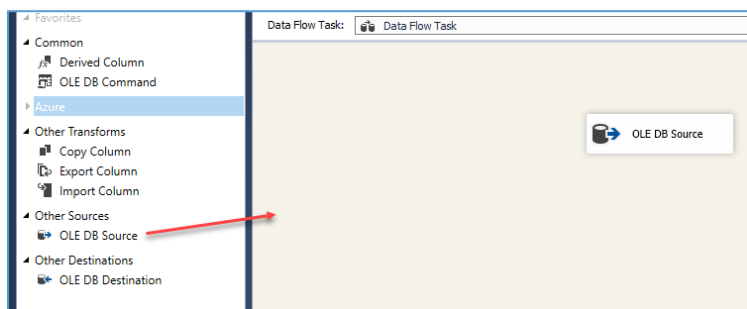
شکل ۳۶-۹ خروجی جدول

برای شروع کار به مانند شکل ۳۷-۹ وارد SSIS شوید و Data Flow Task را وارد قسمت Control Flow کنید و بعد از آن بر روی آن دو بار کلیک کنید تا وارد تب Data Flow شویم.



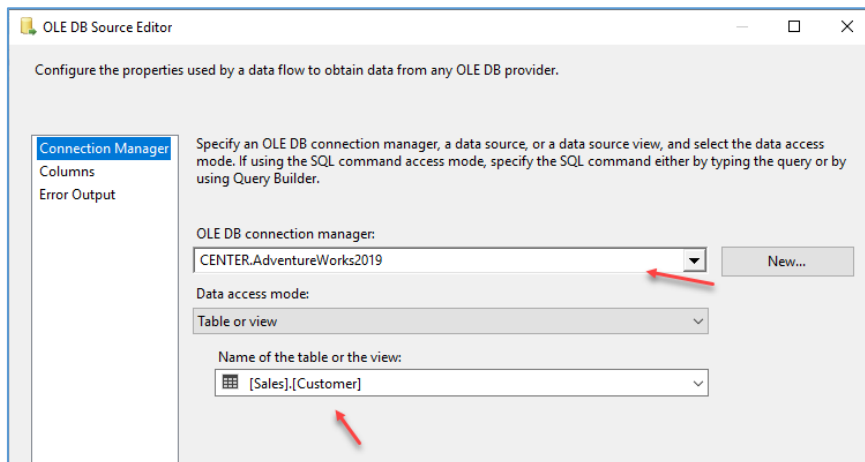
شکل ۹-۳۷ Data Flow

به مانند شکل ۹-۳۸ ابزار OLE DB Source را وارد صفحه Data Flow کنید و بر روی آن کلیک راست کنید و گزینه Edit را انتخاب کنید.



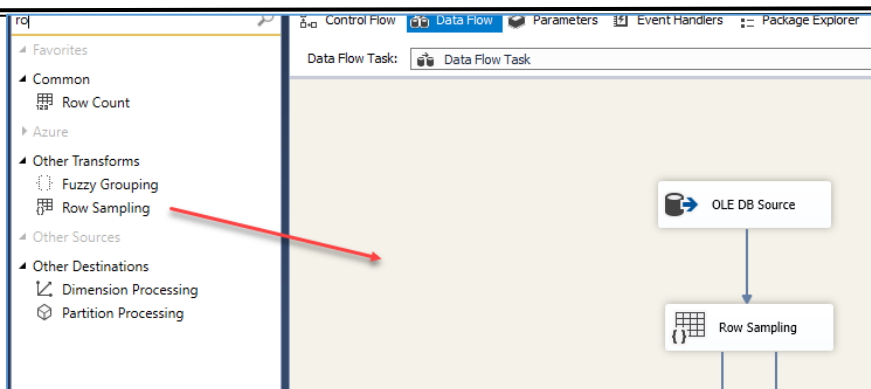
شکل ۹-۳۸ ابزار OLE DB Source

در شکل ۹-۳۹ باید بر روی New کلیک کنید و آدرس سرور به همراه دیتابیس مورد نظر را انتخاب کنید، بعد از این کار باید جدول Sales.Customers را انتخاب کنید و بر روی OK کلیک کنید.



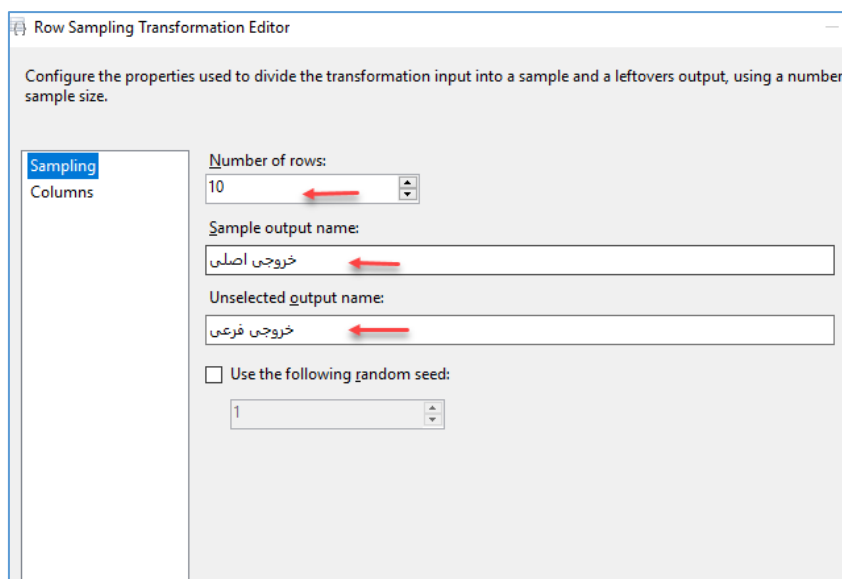
شکل ۹-۳۹ انتخاب جدول

در ادامه کار به مانند شکل ۹-۴۰ از ابزارهای موجود، ابزار Row Sampling را به صفحه اضافه کنید و بر روی آن دو بار کلیک کنید.



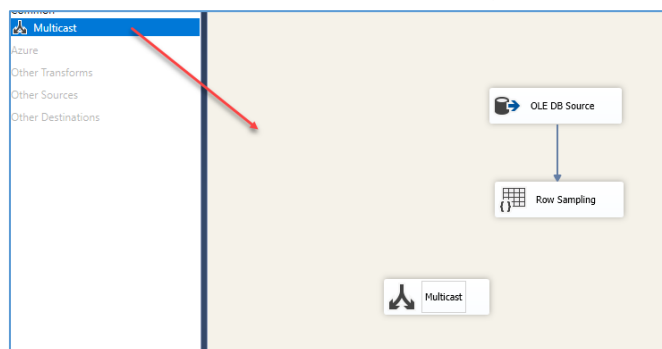
شکل ۹-۴۰ ابزار Row Sampling

در شکل ۹-۴۱ باید در قسمت اول مشخص کنید که چند سطر را می‌خواهید در خروجی نمایش دهید، همانطور که در اول این بخش گفتیم ۱۹۸۲۰ سطر داریم که شما می‌توانید هر تعداد به دلخواه را در خروجی نمایش دهید که البته خروجی به صورت تصادفی می‌باشد، در این قسمت عدد ۱۰ را وارد می‌کنیم و در قسمت Sample output name نام "خروجی اصلی" و در قسمت Unselected output name نام "خروجی فرعی" را وارد می‌کنیم، توجه داشته باشید که در تب Columns می‌توانید مشخص کنید که چه ستون‌هایی در خروجی نمایش داده شوند.



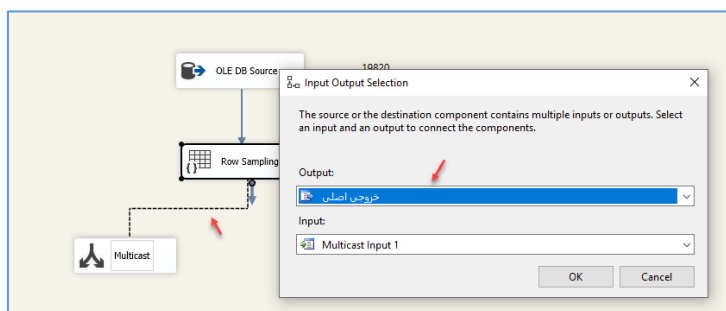
شکل ۹-۴۱ Row Sampling

در ادامه برای اینکه خروجی را نمایش دهیم به مانند شکل ۹-۴۲ دو بار ابزار Multicast را در صفحه قرار دهید



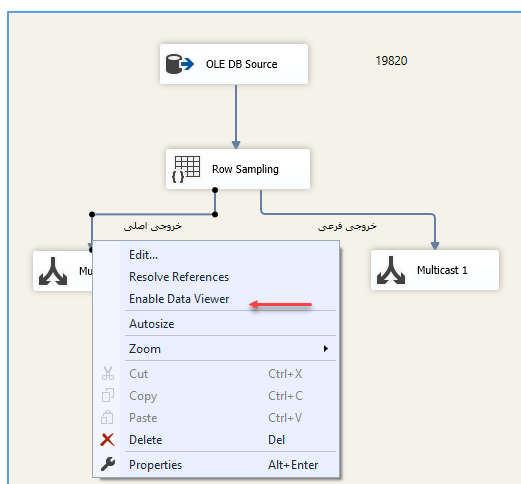
شکل ۹-۴۲ ابزار Multicast

و برای ارتباط با ابزار Row Sampling فلش آبی رنگ را به آن متصل کنید و بعد از اضافه کردن شکل ۹-۴۳ ظاهر می‌شود که دارای دو خروجی است، برای این قسمت گزینه‌ی "خروجی اصلی" را انتخاب کنید و برای گزینه‌ی دیگر "خروجی فرعی" را انتخاب کنید.



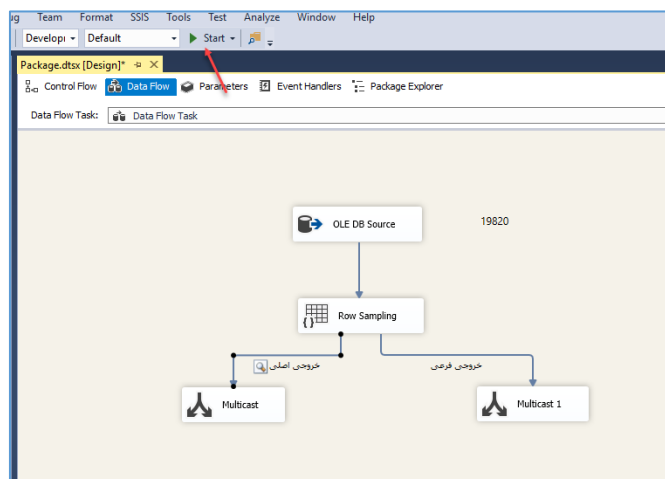
شکل ۹-۴۳ ارتباط بین دو ابزار

در ادامه و بعد از مشخص کردن ارتباط به مانند شکل ۹-۴۴ بر روی لینک ارتباطی "خروجی اصلی" کلیک راست کنید و گزینه‌ی Enable Data Viewer را انتخاب کنید.



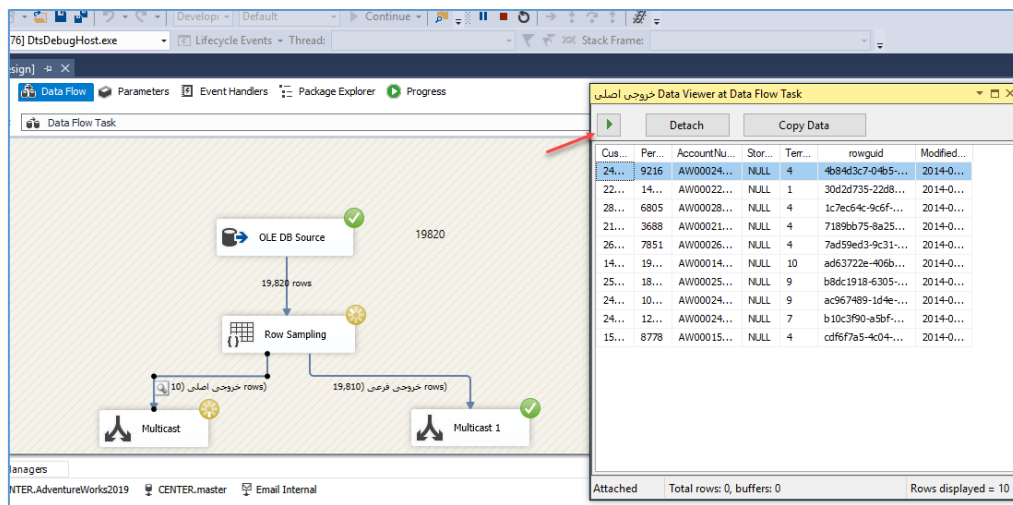
شکل ۹-۴۴ بررسی لینک ارتباطی

اگر به شکل ۹-۴۵ توجه کنید، بعد از اینکه ابزارها را به درستی تنظیم کردید می‌توانید بر روی دکمه start کلیک کنید تا خروجی را مشاهده کنیم.



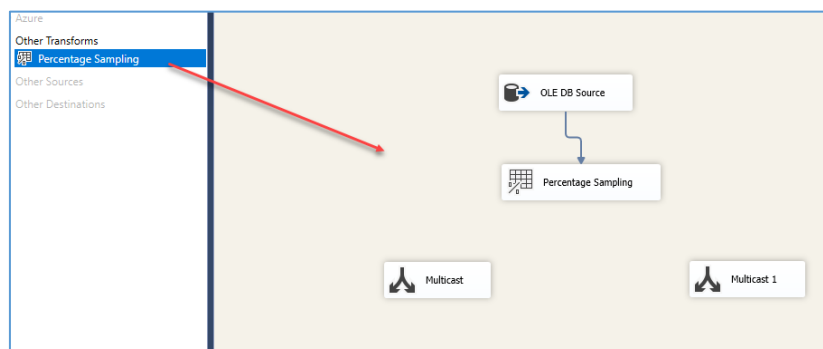
شکل ۹-۴۵ اجرای پروژه

همانطور که در شکل ۴۶-۹ مشاهده می‌کنید، خروجی با موفقیت نمایش داده شده و ۱۰ سطر که مشخص کرده بودیم مشخص شده است، برای اینکه پروژه را به اتمام برسانید باید بر روی فلش سبز رنگ کلیک کنید، به لینک‌های ارتباطی توجه کنید که ۱۰ خروجی در "خروجی اصلی" قرار گرفته و ۱۹۸۱۰ خروجی در "خروجی فرعی".



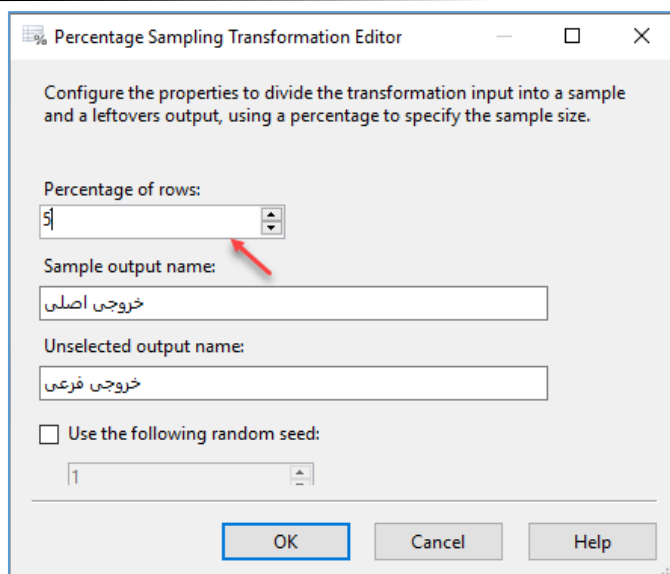
شکل ۴۶-۹ اجرای پروژه

همانطور که مشاهده کردید توانستیم با استفاده از تعداد سطرها خروجی را کنترل و نمایش دهیم، ولی اگر بخواهیم خروجی را به صورت درصد نمایش دهیم نحوه کار تغییر خواهد کرد، برای این کار به مانند شکل ۴۷-۹ ابزار Percentage Sampling را به لیست اضافه کنید.



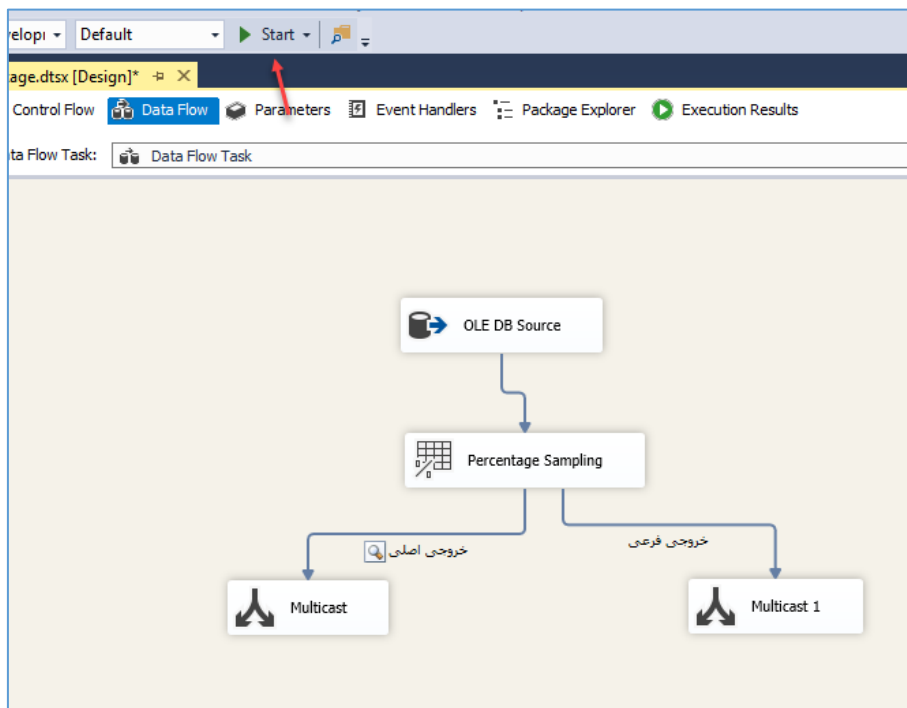
شکل ۴۷-۹ ابزار Percentage Sampling

همانطور که در شکل ۴۸-۹ مشاهده می‌کنید باید درصد مورد نظر خود را وارد کنید که در اینجا ۵ درصد وارد شده است و ورودی اصلی و فرعی مشخص شده است.



شکل ۹-۴۸ تنظیم درصد خروجی

بعد از تکمیل کار باید بر روی دکمه Start به مانند شکل ۹-۴۹ کلیک کنید.



شکل ۹-۴۹ اجرای پروژه

همانطور که در شکل ۹-۵۰ مشاهده می‌کنید خروجی کار مشخص شده است.

The screenshot displays a SQL Server Data Tools interface during the execution of a Data Flow Task. The task flow is as follows:

- OLE DB Source:** Outputs 19,820 rows.
- Percentage Sampling:** Outputs 18,896 rows (فرعی) and 924 rows (اصلی).
- Multicast:** Receives the 924 rows (اصلی).
- Multicast 1:** Receives the 18,896 rows (فرعی).

The Data Viewer on the right shows the following data:

Cus...	Per...	Stor...	Terr...	AccountNu...	rowguid	Modified...
29...	1339	1340	6	AW00029...	8cac7b4-d851-...	2014-0...
29...	1349	1350	5	AW00029...	6a6c0a44-b870-...	2014-0...
30...	1431	1432	8	AW00030...	797e4fb5-ec0f-...	2014-0...
30...	1441	1442	7	AW00030...	c152c07f-f202-...	2014-0...
30...	1447	1448	3	AW00030...	1e4996b9-004d-...	2014-0...
30...	1463	1464	9	AW00030...	c04e5e6c-ebce-...	2014-0...
30...	1815	1816	6	AW00030...	f1141256-0a49-...	2014-0...
30...	1817	1818	9	AW00030...	138ed7b3-65bc-...	2014-0...
30...	1831	1832	10	AW00030...	ba59c532-d2ba-...	2014-0...
30...	1877	1878	2	AW00030...	4f4af2ad-352f-...	2014-0...

شکل ۵۰-۹ اجرای پروژه

منابع

- Pro SQL Server 2019 Administration, 2nd Edition – Peter A. Carter
- Microsoft SQL Server 2012 – William R. Stanek
- SQL Server 2012 T-SQL Recipes – Jason Brimhall – David Dye – Jonathan Gennick – Andy Roberts – Wayne Sheffield
- Microsoft SQL Server 2012 BIBLE – Patrick LeBlanc, Jose Chinchilla, Aaron Nelson, Jorge Segarra , Adam Jorgensen
- SQL Server Integration Services (SSIS) Step by Step Tutorials – Karthikeyan Anbarsan
- <https://www.sqlshack.com/>
- <https://dba.tosinso.com> – مریم حیات رضانی
- <http://www.hpkclasses.ir> – هاله کسمائی
- <https://mug.ac> - سایت تکنولوژی و برنامه نویسی ماگ از مجموعه وب سایت های آموزشی موسسه آموزش عالی - آزاد دانش پژوهان
- <https://www.sqlservertutorial.net/>

تماس با ما

برای ارتباط با ما می توانید سوالات و پیشنهادات خود را به آدرس ایمیل Farshidbabajani@outlook.com و یا Frshidbabajani@gmail.com ارسال کنید.

آدرس تلگرام : @Farshidbabajani

آدرس کانال تلگرامی : @Ciscopress