# iranphp articles

عنوان مقاله : آشنايي با ماژول Mysqli در PHP 5
نگارنده : .....
آدرس پست الکترونيک : .....
تاريخ نگارش : .....

## آشنایی با ماژول Mysqli در PHP 5 :

یکی از ویژگی های جالبی که در php 5 ارائه شده است یک Extension جدید به نام ext/mysqli می باشد که همانند ext/mysql سابق به عنوان یک رابط برای کار با پایگاه داده mysql از آن استفاده می شود. برای برنامه های کاربردی در مقیاس بزرگتر ماژول جدید MySQLi به کاربران اجازه می دهد که از ویژگی های جدید MySQL 4.1 در برنامه های خود استفاده کنند که بازدهی بیشتری را نسبت به نسخه های قبلی MySQL فراهم می کند .

از اواسط دهه ۹۰ تاکنون ext/mysql نقش خود را به عنوان یک پل رابط میان PHP و MySQL به خوبی ایفا کرده است. اگرچه ext/mysql نقاط ضعف کمی داشته است ولی توانسته است که همگام با تغییرات PHP و MySQL پیش رود .

اما وقتی که PHP 5 و MySQL 4.1 پا به عرصه نهادند ext/mysql نشان داد که برای سازگاری با پیشرفت های زیاد این دو نسخه، فقدان هایی را دارد.یکی از این نقاط ضعف مربوط به برقراری ارتباط پایدار و دایمی با پایگاه داده (mysql_pconnect() بود .ویژگی ها و امکاناتی توسط MySQLارائه شد که ext/mysql هنوز قادر به سازگاری با آنها نبود. برای رفع این مشکلات Georg Richter شروع به ساخت یک extensionجدید برای PHP 5 کرد که با ویژگی های ارائه شده در MySQL 4.1 و نسخه های بالاتر همگام باشد .

### مهمترین ویژگی های ارائه شده توسط ext/mysqli

ارائه یک واسط روالی (Procedural Interface) که بسیار شبیه واسط ext/mysql می باشد

ارائه یک واسط شی گرا (Object-Oriented Interface) که در مقایسه با واسط روالی از انعطاف پذیری بیشتری برخوردار است و کار با آن راحت تر است(با توجه به امکانات شی گرایی در PHP 5 )

پشتیبانی از پروتکل باینری جدید MySQL که در نسخه ۴,۱ عرضه شد.(این پروتکل جدید در مقایسه با قبلی از بازدهی بیشتری برخوردار است و امکانات گسترده ای همچون Prepared Statement را پشتیبانی می کند .

پشتیبانی از تمام ویژگی های مجموعه کتابخانه کاربری MySQL C که شامل قابلیت تنظیم گزینه های پیشرفته اتصال از طریق mysqli_init() و توابع وابسته می شود .

به دو دلیل توصیه می شود که به این ماژول جدید switch کنید و از این به بعد با این ماژول کار کنید :

سرعت بیشتر

امنیت بیشتر

حرف i در انتهای ext/mysqli به معنای incomplete incompatible ingenious, interface, improved, می باشد !

### نحوه کار با ext/mysqli

*قبل از کدنویسی توجه داشته باشید که برای استفاده از این ماژول، شما باید PHP 5 و MySQL 4.1 را نصب کرده باشید.کدهای این مقاله از پایگاه داده Worldاستفاده می کنند که به صورت رایگان از آدرس http://www.mysql.com/documentation/index.html قابل دریافت می باشد .

برنامه ساده زیر ابتدا به پایگاه داده متصل می شود و با ارسال یک پرس و جو (Query) به Server ، دریافت نتایج و نمایش آنها و سپس بستن اتصال به کار خود خاتمه می دهد :

```php
<?php

/* Connect to a MySQL server */
$link = mysqli_connect(
            'localhost',  /* The host to connect to */
            'user',       /* The user to connect as */
            'password',   /* The password to use */
            'world');     /* The default database to query */
```

```
if (!$link) {
   printf("Can't connect to MySQL Server. Errorcode: %s\n", mysqli_connect_error());
   exit;
}

/* Send a query to the server */
if ($result = mysqli_query($link, 'SELECT Name, Population FROM City ORDER BY Population DESC LIMIT
5')) {

    print("Very large cities are:\n");

    /* Fetch the results of the query */
    while( $row = mysqli_fetch_assoc($result) ){
        printf("%s (%s)\n", $row['Name'], $row['Population']);
    }

    /* Destroy the result set and free the memory used for it */
    mysqli_free_result($result);
}

/* Close the connection */
mysqli_close($link);
?>
```

بعد از اجرای اسکریپت بالا باید خروجی را به صورت زیر مشاهده کنید:

```
Very large cities are:

Mumbai (Bombay) (10500000)
Seoul (9981619)
S?o Paulo (9968485)
Shanghai (9696300)
Jakarta (9604900
```

همانطور که در کد مشاهده می کنید در اینجا هیچ تفاوت خاصی بین ext/mysql و ext/mysqli مشاهده نمی شود جز افزوده شدن حرف i به آخر توابع !

حال اگر از واسط شی گرا استفاده کنید و برنامه بالا را مجددا بازنویسی کنید به انعطاف پذیری بیشتر و راحتی کار با این واسط جدید پی می برید :

```
<?php

/* Connect to a MySQL server */
$mysqli = new mysqli('localhost', 'user', 'password', 'world');

if (mysqli_connect_errno()) {
   printf("Can't connect to MySQL Server. Errorcode: %s\n", mysqli_connect_error());
   exit;
}

/* Send a query to the server */
if ($result = $mysqli->query('SELECT Name, Population FROM City ORDER BY Population DESC LIMIT 5'))
{

    print("Very large cities are:\n");

    /* Fetch the results of the query */
    while( $row = $result->fetch_assoc() ){
        printf("%s (%s)\n", $row['Name'], $row['Population']);
    }
```

```
    /* Destroy the result set and free the memory used for it */
    $result->close();
}

/* Close the connection */
$mysqli->close();
?>
```

همانطور که مشاهده می کنید اطلاعات مربوط به اتصال در اشیای $mysqli و $result ثبت می شود و برای اجرای دستورات نیازی به مشخص کردن این اطلاعات نداریم. همچنین وقتی در بین نتایج پرس و جو به صورت سطر به سطر توسط تابع fetch_assoc() حرکت می کنیم دیگر نیازی به ساخت یک handle برای کار با رکوردها نداریم .

### استفاده از ویژگی Prepared Statements در پرس و جوها

مزیت Prepared Statements این است که به کاربران امکان ساخت query های بسیار ایمن تر، با بازدهی بیشتر و راحت تر را می دهد. بـه دو صورت می توان از این ویژگی بهره برد bound parameter prepared statements :

bound parameter prepared statementsاین امکان را به ما می دهد که برای Query های خود قالب (Template) درست کنیم و آنها را در Server ذخیره کنیم. وقتی نیاز به اجرای Query داشتیم اطلاعات لازم را برای پر کردن Query به Server می فرسـتیم و سـپس Queryکامل را اجرا می کنیم. این کار دو مزیت دارد :

اول اینکه زمان کمتری صرف تجزیه و تحلیل Query در هنگام اجرا می شود. و دوم اینکه از حملات تزریقی SQL(SQL injection attacks) جلوگیری می شود .

در زیر یک مثال کامل از به کارگیری این ویژگی را در پرس و جوها مشاهده می کنید :

```
<?php
$mysqli = new mysqli('localhost', 'user', 'password', 'world');

/* check connection */
if (mysqli_connect_errno()) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}

$stmt = $mysqli->prepare("INSERT INTO CountryLanguage VALUES (?, ?, ?, ?)");
$stmt->bind_param('sssd', $code, $language, $official, $percent);

$code = 'DEU';
$language = 'Bavarian';
$official = "F";
$percent = 11.2;

/* execute prepared statement */
$stmt->execute();

printf("%d Row inserted.\n", $stmt->affected_rows);

/* close statement and connection */
$stmt->close();

/* Clean up table CountryLanguage */
$mysqli->query("DELETE FROM CountryLanguage WHERE Language='Bavarian'");
printf("%d Row deleted.\n", $mysqli->affected_rows);

/* close connection */
```

```
$mysqli->close();
?>
```

توجه کنید که تابع bind_param() یک رشته کوتاه را به عنوان اولین پارامتر خود می گیرد. این پارامتر String Format می باشد و نوع متغیرهای ارسالی را مشخص می کند. در مثال بالا sssd مشخص می کند که $code و $language و $official از نوع String می باشند و $percent از نوع float یا Double می باشد. مشخص کردن این نوع داده ها باعث می شود که ext/mysqli بداند که چگونه داده ها را برای فرستادن کدگذاری کند و این کار باعث بازدهی بالا می شود.جدول زیر نوع داده های قابل قبول را نشان می دهد

```
BIND TYPE
COLUMN TYPE

i
All INT types

d
DOUBLE and FLOAT

b
BLOBs

s
All other types
```

اگر با ASP.NET هم آشنایی دارید، در زیر می توانید نحوه به کارگیری این ویژگی را در ASP.NETهم مشاهده کنید و با PHP 5 مقایسه کنید :

```
Dim cnn As New SQLConnection("server=LOCALHOST;User id=SA;password=;database=Northwind")
Dim InsertCommand As SqlCommand = New SqlCommand()
InsertCommand.Connection = cnn
Dim sql As String
sql = "INSERT INTO categories (categoryName) VALUES (@newCatName)"

InsertCommand.CommandText = sql

InsertCommand.Parameters.Add("@newCatName", SqlDbType.NVarChar, 250).Value = "Category Name"
```

Bound result prepared statementsاین امکان را می دهد که مقدار فیلدهای دریافتی از پایگاه داده را به متغیر های دلخواه نسبت دهیم(چیزی شبیه تابع list که عناصر یک آرایه را به متغیر های دلخواه نسبت می دهد) .

مثال زیر استفاده از این ویژگی را نشان می دهد :

```
<?php
$mysqli = new mysqli("localhost", "user", "password", "world");

if (mysqli_connect_errno()) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}

/* prepare statement */
if ($stmt = $mysqli->prepare("SELECT Code, Name FROM Country WHERE Code LIKE ? LIMIT 5")) {

    $stmt->bind_param("s", $code);
    $code = "C%";

    $stmt->execute();

    /* bind variables to prepared statement */
    $stmt->bind_result($col1, $col2);
```

```php
    /* fetch values */
    while ($stmt->fetch()) {
        printf("%s %s\n", $col1, $col2);
    }

    /* close statement */
    $stmt->close();
}
/* close connection */
$mysqli->close();

?>
```