



دوره کامل آموزش MySQL (PDF)




+

لینک خرید این pdf

مشاهده دوره آنلاین



راه های ارتباط با ما 📞 📞 📞

	<a href="http://pvlearn.com">http://pvlearn.com</a>	وب سایت
	<a href="https://t.me/pvlearn">https://t.me/pvlearn</a>	آدرس کانال تلگرام
	@pvlearn	آیدی کانال تلگرام
	<a href="https://www.instagram.com/pvlearn">https://www.instagram.com/pvlearn</a>	اینستاگرام

# فهرست جلسات دوره آموزش MySQL

- جلسه ۰۱ : مقدمه ای بر MySQL
- جلسه ۰۲ : مراحل نصب MySQL
- جلسه ۰۳ : مدیریت در MySQL
- جلسه ۰۴ : MySQL در سینتکس PHP
- جلسه ۰۵ : اتصال به دیتابیس در MySQL
- جلسه ۰۶ : ایجاد دیتابیس در MySQL
- جلسه ۰۷ : حذف دیتابیس در MySQL
- جلسه ۰۸ : انتخاب دیتابیس در MySQL
- جلسه ۰۹ : انواع داده در MySQL
- جلسه ۱۰ : جداول در MySQL
- جلسه ۱۱ : حذف جداول در MySQL
- جلسه ۱۲ : کار با دستور INSERT در MySQL
- جلسه ۱۳ : کار با دستور SELECT در MySQL
- جلسه ۱۴ : دستور شرطی WHERE در MySQL
- جلسه ۱۵ : دستور UPDATE در MySQL
- جلسه ۱۶ : کار با دستور DELETE در MySQL
- جلسه ۱۷ : دستور شرطی LIKE در MySQL
- جلسه ۱۸ : مرتب سازی نتایج در MySQL
- جلسه ۱۹ : کار با دستور JOIN در MySQL
- جلسه ۲۰ : مقادیر NULL در MySQL
- جلسه ۲۱ : عملگر REGEXP در MySQL
- جلسه ۲۲ : transaction در MySQL
- جلسه ۲۳ : دستور ALTER در MySQL
- جلسه ۲۴ : INDEX ها در MySQL
- جلسه ۲۵ : جداول موقت در MySQL
- جلسه ۲۶ : جداول Clone در MySQL
- جلسه ۲۷ : اطلاعات دیتابیس در MySQL
- جلسه ۲۸ : توالی ها ( sequence) در MySQL
- جلسه ۲۹ : مدیریت موارد تکراری در MySQL
- جلسه ۳۰ : تزریق به دیتابیس ( SQL Injection) در MySQL
- جلسه ۳۱ : Export کردن دیتابیس در MySQL
- جلسه ۳۲ : Import کردن دیتابیس در MySQL

## مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. پایگاه داده یک برنامه جداگانه است که مجموعه ای از داده ها را ذخیره می کند. هر پایگاه داده دارای یک یا چند API متمایز برای ایجاد، دسترسی، مدیریت، جستجو و تکرار اطلاعات موجود در آن است. انواع مختلف دیگری از ذخیره کننده گان داده نیز می توانند مانند فایل های موجود در سیستم فایل یا جداول های هش بزرگ در حافظه مورد استفاده قرار گیرند، اما جمع آوری و نوشتن داده ها با این نوع سیستم ها چندان آسان نخواهد بود. امروزه ما از سیستم های مدیریت پایگاه داده ارتباطی (RDBMS) برای ذخیره و مدیریت حجم زیادی داده استفاده می کنیم. در ادامه ی این مبحث با ارائه ی [مقدمه ای بر MySQL](#) شما را با این سیستم آشنا می کنیم.

## مقدمه ای بر MySQL

در ادامه ی این آموزش در قالب یک مقدمه ای بر MySQL شما را با مفاهیم کلی و ویژگی های پایگاه داده MySQL آشنا می کنیم.

در ابتدا باید بدانیم که یک سیستم کامل پایگاه داده در واقع یک پایگاه داده ی رابطه ای نامیده می شود.

در پایگاه داده ی رابطه ای تمام داده ها در جداولی ذخیره می شوند که با استفاده از کلیدهای اصلی یا خارجی با هم مرتبط هستند.

یک سیستم مدیریت پایگاه داده رابطه ای (RDBMS) شامل ویژگی های زیر است :

- شما را قادر به پیاده سازی یک پایگاه داده با جداول، ستون ها و شاخص ها می کند.
- تضمین یکپارچگی ارجاع بین ردیف ها در جداول مختلف.
- بروزرسانی خودکار index ها
- یک پرس و جو SQL را تفسیر می کند و اطلاعات را از جداول مختلف ترکیب می کند.

## اصطلاحات RDBMS

قبل از اینکه به توضیح سیستم پایگاه داده MySQL بپردازیم، چندین تعاریف مرتبط با پایگاه داده را بیان می کنیم:

**Database (پایگاه داده):** یک دیتابیس مجموعه ای از جداول با داده های مرتبط است.

**Table (جدول):** جدول در واقع یک ماتریسی از داده ها است. جدول در پایگاه داده همانند صفحه گسترده به نظر می رسد.

**Column (ستون):** یک ستون شال یک یا چند داده از یک نوع است.

**Row (سطر):** یک سطر ، یک گروه از داده های ستون های جدول است.

**Redundancy (افزونگی):** ذخیره سازی دوگانه ی داده ها ، سیستم را سریع تر می کند.

**Primary Key (کلید اصلی):** کلید اصلی ستونی در جدول با مقادیر یکتا و منحصر به فرد است.

**Foreign Key (کلید خارجی):** زمانی که کلید اصلی یک جدول در جدول دیگری نیز باشد به آن کلید خارجی گفته می شود که ارتباط بین دو جدول را ممکن می سازد.

**Compound Key (کلید ترکیبی):** کلید ترکیبی از ترکیب چند ستون ایجاد می شود، چرا که گاهی یک ستون به تنهایی مقدار یکتا ندارد.

**Index :** شماره index در یک پایگاه داده همانند شماره index یک کتاب است.

**Referential Integrity (یکپارچگی ارجاع):** مطمئن می شود که مقادیر کلید خارجی همیشه به یک سطر اشاره دارند.

## پایگاه داده MySQL

سیستم های مدیریت پایگاه داده برای بسیاری از کسب و کارهای کوچک و بزرگ استفاده می شود. MySQL هم سریع بوده و هم برای استفاده کردن نیز آسان است.

پایگاه داده ی MySQL توسط MySQL AB توسعه یافته و پشتیبانی می شود. که یک شرکت سوئدی است.

MySQL به دلایلی بسیاری از جمله موارد زیر محبوب شده است :

- این سیستم برای استفاده رایگان است بنابراین نیازی برای پرداخت هیچ مبلغی ندارید.
- این پایگاه داده بسیار قدرتمند است و بسیاری از ویژگی های یک پایگاه داده ی قدرتمند را دارد.
- MySQL از فرم استاندارد شناخته شده زبان داده های SQL استفاده می کند.
- MySQL روی بسیاری از سیستم عامل ها کار می کند و در بسیاری از زبان ها از جمله C, C++, JAVA, PERL, PHP و... پشتیبانی می شود.
- MySQL حتی با وجود حجم داده های زیاد با سرعت بالایی کار می کند.
- سازگاری بسیار خوبی با زبان محبوب PHP دارد.
- از پایگاه داده های بسیار بزرگ با حداکثر ۵۰ میلیون سطر پشتیبانی می کند.
- حداکثر اندازه برای هر جدول ۴ گیگابایت است، اما شما می توانید با توجه به سیستم عامل خود آن را تا ۸ میلیون ترابایت افزایش دهید.
- MySQL یک سیستم متن باز بوده و به برنامه نویسان اجازه می دهد تا در صورت لزوم قابلیت های آن را تغییر دهند.

## پیش نیازها

قبل از شروع به یادگیری MySQL باید یک دانش پیش زمینه از HTML و PHP داشته باشید.

چراکه نمونه کدها و مثال های MySQL در طی این دوره در کدها و محیط های HTML و PHP تست و اجرا می شود.

## کلام آخر

MySQL یک پایگاه داده ی قدرتمند برای ذخیره ، بازیابی و مدیریت داده ها در محیط وب است که قبل از شروع به یادگیری این سیستم بهتر است که اطلاعاتی را در مورد این پایگاه داده و پیش نیازهای آن داشته باشید، از این رو در اولین جلسه به **مقدمه ای بر MySQL** پرداختیم.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت **پی وی لرن** ، و کاربرانی که **دوره آموزش MySQL** را دنبال می کنند. قبل از شروع به کار با سیستم پایگاه داده ی MySQL ابتدا باید مانند بسیاری از نرم افزارهای دیگر آن را در سیستم عامل خود نصب نمائید. از آنجایی که کاربران معمولا از سیستم عامل های لینوکس ، ویندوز و یا Mac OS X استفاده می کنند، از این رو در این بخش به بررسی **مراحل نصب MySQL** در این سیستم عامل ها پرداخته ایم. با دنبال کردن این مراحل ، زمینه برای کار با سیستم مدیریت پایگاه داده ی MySQL در سیستم عامل شما فراهم خواهد شد. در ادامه ی این مبحث با آموزش مراحل نصب MySQL با ما همراه باشید.

### آموزش مراحل نصب MySQL

در ادامه ی این آموزش با مراحل نصب MySQL در سیستم عامل های ویندوز ، لینوکس و تایید نصب آن آشنا خواهید شد.

### مراحل نصب MySQL در سیستم عامل های لینوکس

روش توصیه شده برای نصب دیتابیس MySQL در سیستم عامل لینوکس از طریق RPM است.

MySQL شامل بخش های زیر است:

- **MySQL**: سرور دیتابیس MySQL ، جداول و دسترسی کاربران در دیتابیس ها را مدیریت می کند.
- **MySQL-client**: برنامه ی کاربر که ممکن است با سرور ارتباط برقرار کنند.
- **MySQL-devel**: کتابخانه ها و هدر فایلی است که هنگام کامپایل دیگر برنامه های مورد استفاده در MySQL استفاده می شوند.
- **MySQL-shared**: کتابخانه های به اشتراک گذاشته شده برای MySQL client است.
- **MySQL-bench**: کارایی ابزار را برای سرور دیتابیس MySQL تست می کند.

RPM های MySQL ذکر شده در اینجا همه روی سیستم عامل لینوکس **SuSE** ساخته شده است.

اما ممکن است که بدون هیچ مشکلی روی سایر نسخه های لینوکس نیز کار کند.

در نهایت با دنبال کردن مراحل نصب زیر ، می توانید این پایگاه داده را روی سیستم عامل لینوکس خود نصب نمائید:

با استفاده از **root** کاربر به سیستم Login شوید.

به دایرکتوری حاوی RPM ها سوئیچ شوید.

با اجرای دستور زیر، سرور دیتابیس MySQL را نصب نمائید (به خاطر داشته باشید که filename بخش italics را با نام فایل RPM مورد نظر خود جایگزین کنید).

### مثال :

```
1. [root@host]# rpm -i MySQL-5.0.9-0.i386.rpm
```

دستور فوق برای نصب MySQL server طول خواهد کشید.

یک حساب کاربری MySQL ایجاد می شود ، تنظیمات ضروری ایجاد شده و MySQL server بصورت خودکار آغاز می شود.

شما می توانید باینری های مرتبط با MySQL را در مسیر /usr/bin and /usr/sbin بیابید.

تمام جداول و دیتابیس ها در مسیر /var/lib/mysql/ ایجاد خواهد شد.

کدهای زیر اختیاری بوده اما توصیه می شود که برای نصب RPM های باقی مانده انجام شود:

## مثال :

```
1. [root@host]# rpm -i MySQL-client-5.0.9-0.i386.rpm
2. [root@host]# rpm -i MySQL-devel-5.0.9-0.i386.rpm
3. [root@host]# rpm -i MySQL-shared-5.0.9-0.i386.rpm
4. [root@host]# rpm -i MySQL-bench-5.0.9-0.i386.rpm
```

## نصب MySQL روی ویندوز

نصب پیش فرض MySQL در ویندوز، برای استفاده آسان تر است. کافیت به سادگی پکیج نصب MySQL را دانلود نموده و آن را از حالت ZIP خارج کنید، سپس با اجرای فایل setup.exe مراحل نصب MySQL را دنبال می کنید.

installer پیش فرض setup.exe همه چیز را در حالت پیش فرض تحت مسیر C:\mysql خواهد کرد.

سرور را در خط فرمان تست کنید. به محل **mysqld server** که احتمالاً در مسیر C:\mysql\bin قرار دارد مراجعه کرده و کد زیر را تایپ نمائید:

## مثال :

```
1. mysqld.exe --console
```

**نکته :** در ویندوز NT به جای mysqld.exe باید mysqld-nt.exe را تایپ نمائید.

اگر همه چیز خوب پیش رفته باشد به شما پیغامی در مورد راه اندازی و InnoDB ظاهر خواهد کرد.

در غیر این صورت ممکن است محدودیت نصب در سیستم عامل خود داشته باشید.

مطمئن شوید که دایرکتوری تمام داده های شما را برای کاربران در دسترس قرار می دهد.

MySQL خود را به منوی Start ویندوز اضافه نمی کند و به هیچ وجه یک رابط کاربری خوب برای توقف سرور ندارد.

پس اگر می خواهید سرور را با دوبار کلیک روی mysqld اجرا کنید باید از mysqladmin برای توقف آن استفاده کنید.

## تایید کردن نصب MySQL

بعد از اینکه MySQL به موفقیت نصب شد، جداول پایه آنالیز شده و سرور شروع به کار می کند.

شما می توانید تایید کنید که همه ی موارد شروع به کار کند.

## استفاده از ابزار mysqladmin برای دریافت وضعیت سرور

با استفاده از باینری **mysqladmin** نسخه ی سرور را چک کنید. این باینری از مسیر `C:\mysql\bin\` در `usr/bin` on linux and in `C:\mysql\bin\` ویندوز قابل دسترسی است:

مثال :

```
1. [root@host]# mysqladmin --version
```

در لینوکس نیز نتیجه ی زیر تولید خواهد شد :

مثال :

```
1. mysqladmin Ver 8.23 Distrib 5.0.9-0, for redhat-linux-gnu on i386
```

در نهایت در پیامی گزارش وضعیت اتمام نصب را مشاهده خواهید کرد و اگر خطایی رخ داده باشد در آن پیام خواهید دید.

## اجرای ساده ی دستورات SQL با استفاده از MySQL client

شما می توانید در طول MySQL client و با استفاده از **mysql** command به MySQL server خود متصل شوید.

در این لحظه نیازی به تنظیم کلمه ی عبور (password) پیش فرض ندارید، شما می توانید فقط از کد زیر استفاده کنید :

مثال :

```
1. [root@host]# mysql
```

در حال حاضر شما به سرور SQL متصل هستید و می توانید دستورات SQL را بعد از سینتکس `mysql>` تایپ کنید.

مثال :

```
1. mysql> SHOW DATABASES;
2. +-----+
3. | Database |
4. +-----+
5. | mysql   |
6. | test    |
7. +-----+
8. 2 rows in set (0.13 sec)
```

## مراحل بعد از نصب

MySQL با `password` خالی در `root` کاربری ایجاد شده است. بعد از نصب موفقیت آمیز می توانید کلمه عبور را در `root` تنظیم کنید

مثال :

```
1. [root@host]# mysqladmin -u root password "new_password";
```

حالا برای اتصال به MySQL server باید از خط فرمان زیر استفاده کنید :

مثال :

1. [root@host]# mysql -u root -p
2. Enter password:\*\*\*\*\*

کاربران UNIX امکان ذخیره کردن مسیر کامل دایرکتوری MySQL را ذخیره کنید تا هر بار نیازی به تایپ مسیر کامل نباشد.

مثال :

1. export PATH = \$PATH:/usr/bin:/usr/sbin

## اجرای MySQL در زمان Boot

اگر شما می خواهید که MySQL server در زمان Boot اجرا شود محتوای زیر را در فایل etc/rc.local اجرا کنید:

مثال :

1. /etc/init.d/mysqld start

همچنین، شما باید mysqld باینری را در دایرکتوری /etc/init.d داشته باشید.

## کلام آخر

پس از شناخت سیستم مدیریت دیتابیس و آشنایی با قابلیت و ویژگی های MySQL و پیش نیازهای آن ، اولین گام برای استفاده از این سیستم مدیریت دیتابیس ، دنبال کردن **مراحل نصب MySQL** در سیستم عامل خود است که در این مبحث ما به آن پرداختیم.



### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بخش قبلی با روند نصب MySQL در سیستم عامل های لینوکس و ویندوز و نحوه ی اجرای آن آشنا شدیم ، بعد از طی کردن این روند سیستم مدیریت پایگاه داده ی MySQL آماده ی استفاده است. اما قبل از هر چیز باید با چگونگی تنظیم و مدیریت در MySQL در حد نیاز آشنایی داشته باشید تا بتوانید داده های خود را در قالب جداول در MySQL ذخیره و آن ها را بخوبی مدیریت کنید، برای این منظور در این بخش به چگونگی **مدیریت در MySQL** پرداخته ایم.

### مدیریت در MySQL

در این آموزش شما با چگونگی اجرا، خروج ، تنظیمات ضروری و اقدامات اصلی در MySQL آشنا خواهید شد.

### اجرا و خروج از MySQL server

در ابتدا بررسی کنید که آیا MySQL server شما در حال اجراست یا خیر، برای این منظور می توانید از خط فرمان زیر استفاده کنید :

مثال :

```
1. ps -ef | grep mysqld
```

اگر MySQL در حال اجرا باشد ، سپس شما می توانید فرآیند MySQL لیست شده را در نتیجه ها مشاهده کنید، و اگر سرور MySQL در حال اجرا نباشد ، می توانید از طریق خط کد زیر آن را اجرا کنید:

مثال :

```
1. root@host# cd /usr/bin
2. ./safe_mysqld &
```

حالا اگر می خواهید سرور MySQL در حال اجرا را متوقف کنید می توانید از خط فرمان زیر استفاده کنید:

مثال :

```
1. root@host# cd /usr/bin
2. ./mysqladmin -u root -p shutdown
3. Enter password: *****
```

### راه اندازی حساب کاربری MySQL

برای افزودن کاربر جدید به MySQL ، شما فقط به افزودن یک ورودی جدید به جدول کاربر در بانک اطلاعاتی نیاز دارید.

برنامه ی زیر یک مثال از افزودن کاربر **guest** (مهمان) جدید با دستورات SELECT, INSERT و UPDATE با رمز عبور **guest123** می باشد.

این SQL query به شرح زیر است :

## مثال :

```
1. root@host# mysql -u root -p
2. Enter password:*****
3. mysql> use mysql;
4. Database changed
5.
6. mysql> INSERT INTO user
7.   (host, user, password,
8.    select_priv, insert_priv, update_priv)
9.   VALUES ('localhost', 'guest',
10.    PASSWORD('guest123'), 'Y', 'Y', 'Y');
11. Query OK, 1 row affected (0.20 sec)
12.
13. mysql> FLUSH PRIVILEGES;
14. Query OK, 1 row affected (0.01 sec)
15.
16. mysql> SELECT host, user, password FROM user WHERE user = 'guest';
17. +-----+-----+-----+
18. | host | user | password |
19. +-----+-----+-----+
20. | localhost | guest | 6f8c114b58f2ce9e |
21. +-----+-----+-----+
22. 1 row in set (0.00 sec)
```

همانطور که در مثال فوق هم مشاهده می کنید رمز عبور mypass با f8c114b58f2ce9e۶ رمزنگاری شده است، به کد FLUSH PRIVILEGES توجه کنید ، این به سرور اعلام می کند که جداول grant را بارگذاری کند.

اگر شما از کد فوق استفاده نکنید، قادر نخواهید بود تا زمانیکه سرور مجددا راه اندازی شود با استفاده از حساب کاربری جدید به MySQL متصل شوید.

همچنین می توانید سایر امتیازات را به کاربر جدید با مقداردهی ستون های جداول کاربری با 'Y' مشخص کنید.

همچنین می توانید آن ها را بعدا با استفاده از دستور UPDATE بروزرسانی کنید:

- Select\_priv
- Insert\_priv
- Update\_priv
- Delete\_priv
- Create\_priv
- Drop\_priv
- Reload\_priv
- Shutdown\_priv
- Process\_priv
- File\_priv
- Grant\_priv
- References\_priv
- Index\_priv

سایر روش های افزودن حساب کاربری جدید، استفاده از دستورات SQL GRANT است، مثال زیر کاربر **zara** را با رمز عبور **zara123** برای یک دیتابیس خاص با نام **TUTORIALS** اضافه می کند.

### مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use mysql;
4. Database changed
5.
6. mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
7.     -> ON TUTORIALS.*
8.     -> TO 'zara'@'localhost'
9.     -> IDENTIFIED BY 'zara123';
```

این همچنین ورودی در جدول دیتابیس MySQL با نام **user** ایجاد می کند.

**نکته :** انتهای هر دستور را با سمی کولون (;) مشخص کنید.

### تنظیم فایل /etc/my.cnf

در اغلب موارد شما نباید این فایل را دستکاری کنید، حالت پیش فرض این فایل حاوی محتوای زیر است :

### مثال :

```
1. [mysqld]
2. datadir = /var/lib/mysql
3. socket = /var/lib/mysql/mysql.sock
4.
5. [mysql.server]
6. user = mysql
7. basedir = /var/lib
8.
9. [safe_mysqld]
10. err-log = /var/log/mysqld.log
11. pid-file = /var/run/mysqld/mysqld.pid
```

در اینجا، شما می توانید دایرکتوری های مختلف برای ورود به خطا را مشخص کنید، در غیر این صورت شما نباید محتوای این جدول را تغییر دهید.

## دستور مدیریت MySQL

در اینجا لیستی از دستورات مهم مدیریت MySQL را آورده ایم، که شما از آن ها برای کار با دیتابیس MySQL استفاده خواهید کرد:

- **USE Databasename** : از این دستور برای انتخاب یک پایگاه داده در محیط کاری MySQL استفاده می شود.
- **SHOW DATABASES**: دیتابیس های قابل دسترسی در MySQL را لیست می کند.
- **SHOW TABLES** : جداول قابل دسترسی در MySQL را لیست می کند.
- **SHOW COLUMNS FROM tablename** : صفات ، انواع صفات ، اطلاعات key، پیش فرض ها و سایر اطلاعات یک جدول را نمایش می دهد.
- **SHOW INDEX FROM tablename**: تمام جزئیات index های جدول ، شامل کلید اصلی را نمایش می دهد.
- **SHOW TABLE STATUS LIKE tablename\G** : گزارش جزئیات عملکرد DBMS دیتابیس MySQL.

## کلام آخر

بعد از نصب محیط کار با دیتابیس MySQL در سیستم عامل خود ، زمینه برای کار با MySQL و استفاده از قابلیت های آن فراهم می شود، اما قبل از هر چیز باید با دستورات اصلی و مهم **مدیریت در MySQL** آشنایی داشته باشید تا بتوانید از MySQL استفاده کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. همانطور که در ابتدای دوره هم اشاره شده دیتابیس MySQL برای ذخیره ، بازیابی و مدیریت داده ها در برنامه های تحت وب طراحی و توسعه یافته است، از این رو برای استفاده از دستورات و قابلیت های این دیتابیس محبوب باید با سینتکس و اتصال به MySQL در زبان های تحت وب ، به ویژه PHP آشنایی داشته باشید، PHP یکی از محبوب ترین زبان های برنامه نویسی تحت وب است که از دیتابیس MySQL نیز پشتیبانی می کند، همچنین یک زبان متن باز و رایگان است، از این رو در طی این دوره برای آموزش کار با MySQL بیشتر سعی شده که از سینتکس و کدهای PHP استفاده شود، در ادامه ی مباحث برای [آشنایی با MySQL در سینتکس PHP](#) با ما همراه باشید.

### آشنایی با MySQL در سینتکس PHP

در ادامه ی این مبحث شما با ساختار استفاده از MySQL در سینتکس PHP آشنا خواهید شد. PHP توابع مختلف دسترسی به پایگاه داده MySQL را ارائه می دهد. PHP همچنین توابع دستکاری اطلاعات ثبت شده در داخل بانک اطلاعاتی را فراهم می کند.

### سینتکس پایه

در توابع PHP برای استفاده با MySQL ، از فرمت عمومی زیر استفاده می شود :

#### مثال :

```
1. mysql_function(value,value,...);
```

بخش دوم نام تابع معمولا عملکرد تابع را مشخص می کند، دو تابع اصلی در این آموزش ها استفاده می شود :

#### مثال :

```
1. mysqli_connect($connect);
2. mysqli_query($connect,"SQL statement");
```

مثال زیر یک سینتکس generic از PHP را برای فراخوانی هر یک از توابع MySQL را نشان می دهد :

#### مثال :

```
1. <html>
2.   <head>
3.     <title>PHP with MySQL</title>
4.   </head>
5.
6.   <body>
7.     <?php
8.       $retval = mysql_function(value, [value,...]);
9.       if( !$retval ) {
10.        die ( "Error: a related error message" );
11.      }
12.      // Otherwise MySQL or PHP Statements
13.    ?>
14.  </body>
15. </html>
```

از بخش بعدی ما تمام توابع مهم برای اتصال و مدیریت MySQL در PHP را یک به یک بررسی کرده و آموزش خواهیم داد.

## کلام آخر

سیستم قدرتمند مدیریت پایگاه داده ی MySQL برای ذخیره و مدیریت داده ها در برنامه ی های تحت وب تولید و توسعه پیدا کرده است، و از آنجایی که PHP بهترین زبان تحت وب برای اتصال و مدیریت داده های MySQL است، در این مبحث به **آشنایی با MySQL** در **سینتکس PHP** پرداختیم.

## جلسه ۵: اتصال به دیتابیس در MySQL

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#)، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بخش قبلی تا حدودی با سینتکس دستورات MySQL در زبان PHP آشنا شدیم، همچنین در انتهای بخش به چند مورد از دستورات بسیار مهم و اصلی برای کار با دیتابیس MySQL اشاره کردیم، برای کار با MySQL در کد PHP یا هر زبان دیگری در ابتدا باید به دیتابیس ایجاد شده در MySQL متصل شوید، برای اینکار دستوراتی ارائه شده است که شما را به دیتابیس مورد نظرتان در MySQL متصل می کند، که ما در ادامه ی این مبحث دستورات [اتصال به دیتابیس در MySQL](#) را بررسی خواهیم کرد.

### آشنایی با اتصال به دیتابیس در MySQL

در ادامه ی این آموزش سینتکس اصلی اتصال به دیتابیس در MySQL را به همراه کد مربوطه را ارائه خواهیم کرد، شما می توانید با استفاده از باینری **mysql** در خط فرمان دیتابیس MySQL را ایجاد کنید.

این یک مثال ساده برای اتصال به سرور از طریق خط فرمان را نشان می دهد:

#### مثال :

1. [root@host]# mysql -u root -p
2. Enter **password**:\*\*\*\*\*

بعد از دستور **mysql>** می توانید هر یک از دستورات SQL را تایپ و اجرا نمایید، این یک نتیجه از کد فوق است:

#### مثال :

1. Welcome **to** the MySQL monitor. Commands **end with** ; or \g.
2. Your MySQL **connection id is** 2854760 **to** server version: 5.0.9
- 3.
4. Type '**help**;' or '**\h**' for help. Type '**\c**' to clear the buffer.

در مثال فوق، ما از **root** به عنوان کاربر استفاده می کنیم، اما شما می توانید از هر کاربر دیگری نیز استفاده کنید، هر کاربری قادر به انجام تمام عملیات SQL که اجازه آن را داشته باشد خواهد بود.

شما می توانید با استفاده از دستور **exit** بعد از **mysql>** هر زمانی ارتباط با دیتابیس MySQL را قطع کنید.

#### مثال :

1. mysql> exit
2. Bye

### اتصال به MySQL با استفاده از PHP

زبان PHP تابع **mysql\_connect()** را برای باز کردن اتصال دیتابیس ارائه می کند، این تابع شامل ۵ پارامتر است و یک شناسه ی لینک **MySQL** را در صورت موفقیت و **FALSE** را در صورت خطا بر می گرداند.

1. `connection mysql_connect(server,user,passwd,new_link,client_flag);`

ردیف	پارامتر و توضیحات
۱	<p><b>server</b></p> <p>Optional – نام میزبان سرور پایگاه داده در حال اجرا. اگر مشخص نشده باشد، سپس مقدار پیش فرض خواهد شد: localhost:3306.</p>
۲	<p><b>user</b></p> <p>Optional – نام کاربری دسترسی به پایگاه داده، اگر مشخص نشده باشد، سپس پیش فرض نام کاربری است که فرایند سرور خواهد شد.</p>
۳	<p><b>passwd</b></p> <p>Optional – رمز عبور کاربر دسترسی به پایگاه داده. اگر مشخص نشده باشد، سپس پیش فرض یک کلمه عبور empty خواهد بود.</p>
۴	<p><b>new_link</b></p> <p>Optional – فراخوانی با تابع mysql_connect() است.</p>
۵	<p><b>client_flags</b></p> <p>Optional ترکیبی از ثابت های زیر :</p> <ul style="list-style-type: none"> <li>• MYSQL_CLIENT_SSL استفاده از رمزنگاری . SSL</li> <li>• MYSQL_CLIENT_COMPRESS استفاده از پروتکل فشرده سازی.</li> <li>• MYSQL_CLIENT_IGNORE_SPACE اجازه ی space بعد از نام توابع را می دهد.</li> <li>• MYSQL_CLIENT_INTERACTIVE ایست ثانیه ای قبل از بستن اتصال دیتابیس است.</li> </ul>



شما می توانید در هر زمانی با استفاده از تابع `mysql_close()` اتصال خود را با دیتابیس قطع کنید.

این تابع یک پارامتر را می گیرد، که یک اتصال را با تابع `mysql_connect()` باز می گرداند.

## سینتکس

مثال :

```
1. bool mysql_close ( resource $link_identifier );
```

اگر منابع مشخص نشده باشد، آخرین دیتابیس باز شده بسته می شود، این تابع در صورتی که اتصال با موفقیت بسته شده باشد `true` و در غیر این صورت `false` را برمی گرداند.

کد زیر چگونگی یک اتصال ساده به یک MySQL server را نشان می دهد :

مثال :

```
1. <html>
2.   <head>
3.     <title>Connecting MySQL Server</title>
4.   </head>
5.   <body>
6.     <?php
7.       $dbhost = 'localhost:3306';
8.       $dbuser = 'guest';
9.       $dbpass = 'guest123';
10.      $conn = mysql_connect($dbhost, $dbuser, $dbpass);
11.
12.      if(! $conn ) {
13.        die('Could not connect: ' . mysql_error());
14.      }
15.      echo 'Connected successfully';
16.      mysql_close($conn);
17.    ?>
18.  </body>
19. </html>
```

## کلام آخر

برای انجام هر گونه اقدامی در دیتابیس باید ابتدا به دیتابیس مورد نظرتان متصل شوید، از این رو باید با سینتکس اصلی **اتصال به دیتابیس در MySQL** آشنایی داشته باشید که در این بخش به آن پرداخته شد.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بخش قبلی با سینتکس اصلی و چگونگی اتصال به دیتابیس موجود در MySQL آشنا شدید ، اما این اتصال زمانی امکان پذیر خواهد بود که دیتابیس برای اتصال از قبل موجود باشد که در غیر این صورت باید ابتدا یک دیتابیس ایجاد کنید ، ایجاد دیتابیس در MySQL بسیار ساده است که فقط با یک تابع مشخص و کدهای SQL مربوط به ایجاد دیتابیس انجام می شود، در ادامه ی این مبحث چگونگی **ایجاد دیتابیس در MySQL** را بررسی کرده ایم.

### چگونگی ایجاد دیتابیس در MySQL

در ادامه ی این آموزش دو روش عمده برای ایجاد دیتابیس در MySQL را بررسی کرده ایم.

#### ایجاد دیتابیس با استفاده از mysqladmin

شما نیاز به امتیازات ویژه برای ایجاد یا حذف یک دیتابیس MySQL را دارید، بنابراین با فرض اینکه شما به root دسترسی دارید، می توانید با استفاده از mysqladmin دیتابیس ایجاد کنید.

این یک مثال ساده از ایجاد دیتابیس است که **TUTORIALS** نامیده می شود:

مثال :

1. [root@host]# mysqladmin -u root -p create TUTORIALS
2. Enter password:\*\*\*\*\*

این دیتابیس MySQL را ایجاد می کند که TUTORIALS نام دارد.

#### ایجاد دیتابیس با استفاده از PHP

زبان PHP از تابع **mysql\_query** برای ایجاد یا حذف یک دیتابیس MySQL استفاده می کند، این تابع دو پارامتر را می گیرد و در صورت موفقیت TRUE و در غیر این صورت FALSE را تولید می کند.

### سینتکس

مثال :

1. bool mysql\_query( sql, connection );

ردیف	پارامتر و توضیحات
۱	<p><b>sql</b></p> <p>Required یک پرس و جوی SQL برای ایجاد و یا حذف بانک اطلاعاتی است.</p>
۲	<p><b>connection</b></p> <p>Optional - اگر اتصال مشخص نشده باشد، آخرین اتصال باز توسط mysql_connect بسته خواهد شد.</p>

## مثال

این یک مثال ساده از ایجاد یک دیتابیس است :

### مثال :

```

1. <html>
2.   <head>
3.     <title>Creating MySQL Database</title>
4.   </head>
5.
6.   <body>
7.     <?php
8.       $dbhost = 'localhost:3036';
9.       $dbuser = 'root';
10.      $dbpass = 'rootpassword';
11.      $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12.
13.      if(! $conn ) {
14.        die('Could not connect: ' . mysql_error());
15.      }
16.      echo 'Connected successfully<br />';
17.      $sql = 'CREATE DATABASE TUTORIALS';
18.      $retval = mysql_query( $sql, $conn );
19.
20.      if(! $retval ) {
21.        die('Could not create database: ' . mysql_error());
22.      }
23.      echo "Database TUTORIALS created successfully\n";
24.      mysql_close($conn);
25.    ?>
26.   </body>
27. </html>

```

## کلام آخر

برای ذخیره ، بازبازی و مدیریت داده ها در MySQL باید یک دیتابیس در MySQL ایجاد کرده باشید تا بتوانید ایجاد جداول در آن ، امکان ذخیره داده ها را فراهم کنید، از این رو مباحث این آموزش را به **ایجاد دیتابیس در MySQL** اختصاص دادیم.

## جلسه ۷۰ : حذف دیتابیس در MySQL

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت **پی وی لرن** ، و کاربرانی که **دوره آموزش MySQL** را دنبال می کنند. در بخش قبل چگونگی ایجاد دیتابیس در MySQL بررسی شد، در مقابل ایجاد دیتابیس در MySQL ما عمل حذف دیتابیس از MySQL را داریم، گاهی به دلایل مختلفی از جمله بی استفاده بودن یک دیتابیس ممکن است تصمیم به حذف آن دیتابیس بگیرید از این رو باید با چگونگی **حذف دیتابیس در MySQL** در حد نیاز آشنایی داشته باشید. از این رو در ادامه ی این مبحث با چگونگی حذف دیتابیس در MySQL با ما همراه باشید.

### حذف دیتابیس با استفاده از mysqladmin

شما نیاز به امتیازات ویژه برای ایجاد یا حذف یک دیتابیس MySQL را دارید، بنابراین با فرض اینکه شما به root دسترسی دارید، می توانید با استفاده از mysqladmin دیتابیسی را حذف کنید.

قبل از حذف هر پایگاه داده ای در نظر داشته باشید که با حذف آن تمام اطلاعات موجود در آن را نیز از دست خواهید داد.

مثال زیر چگونگی حذف دیتابیسی که در بخش قبلی ایجاد کرده بودیم را نشان می دهد:

#### مثال :

1. [root@host]# mysqladmin -u root -p **drop** TUTORIALS
2. Enter **password:\*\*\*\*\***

اجرای کد فوق، در ابتدا به شما هشدار می دهد که آیا از حذف پایگاه داده مطمئن هستید یا خیر:

#### مثال :

1. Dropping the **database is** potentially a very bad thing **to** do.
2. Any data stored in the **database** will be destroyed.
- 3.
4. Do you really want **to drop** the 'TUTORIALS' database [y/N] **y**
5. **Database "TUTORIALS"** dropped

### حذف پایگاه داده با استفاده از اسکریپت PHP

زبان PHP از تابع **mysql\_query** برای ایجاد و یا حذف یک دیتابیس MySQL استفاده می کند، این تابع دو پارامتر می گیرد و مقادیر TRUE یا FALSE را برمی گرداند.

### سینتکس

#### مثال :

```
1. bool mysql_query( sql, connection );
```

ردیف	پارامتر و توضیحات
۱	<b>sql</b> Required - پرس و جوی ایجاد یا حذف دیتابیس mysql است.
۲	<b>connection</b> Optional - اگر اتصال مشخص نشده باشد، آخرین اتصال باز توسط mysql_connect بسته خواهد شد.

مثال زیر چگونگی حذف یک پایگاه داده را نشان می دهد:

مثال :

```
1. <html>
2.   <head>
3.     <title>Deleting MySQL Database</title>
4.   </head>
5.
6.   <body>
7.     <?php
8.       $dbhost = 'localhost:3036';
9.       $dbuser = 'root';
10.      $dbpass = 'rootpassword';
11.      $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12.
13.      if(! $conn ) {
14.        die('Could not connect: ' . mysql_error());
15.      }
16.      echo 'Connected successfully<br />';
17.      $sql = 'DROP DATABASE TUTORIALS';
18.      $retval = mysql_query( $sql, $conn );
19.
20.      if(! $retval ) {
21.        die('Could not delete database: ' . mysql_error());
22.      }
23.      echo "Database TUTORIALS deleted successfully\n";
24.      mysql_close($conn);
25.    ?>
26.  </body>
27. </html>
```

**نکته :** اگر برای حذف دیتابیس از اسکریپت PHP استفاده کنید ، اینکار در حالت عادی هیچ هشدارى را قبل از حذف دیتابیس ظاهر نمى کند.

## کلام آخر

ایجاد و حذف دیتابیس ها دو عمل اصلی در مدیریت دیتابیس ها هستند، که در MySQL نیز با آن سر و کار خواهید داشت، در بخش قبلى چگونگی ایجاد دیتابیس در MySQL بررسی شد و در این بخش در تکمیل مباحث بخش قبلى چگونگی **حذف دیتابیس در MySQL** را ارائه کردیم.

## جلسه ۸ : انتخاب دیتابیس در MySQL

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. تاکنون با روند ایجاد و حذف دیتابیس در MySQL آشنا شدید، حالا اگر چندین دیتابیس در MySQL ایجاد کرده باشید ، برای کار با هر دیتابیس یا حذف آن باید ابتدا دیتابیس مربوطه را انتخاب کنید، MySQL دستورات و توابع لازم را برای انتخاب هر یک از دیتابیس های موجود، برای دستکاری آن ها ، فراهم می کند. معمولا هر دیتابیس شامل یک نام می شود که برای دسترسی به آن استفاده می شود، در ادامه ی این مباحث برای آشنایی با چگونگی **انتخاب دیتابیس در MySQL** با ما همراه باشید.

### آشنایی با انتخاب دیتابیس در MySQL

در ادامه ی این آموزش با روش های انتخاب دیتابیس در MySQL آشنا خواهید شد، هنگامی که با سرور MySQL ارتباط برقرار می کنید، لازم است که یک پایگاه داده را برای کار با آن انتخاب کنید.

### انتخاب دیتابیس MySQL از طریق خط فرمان

این روش به سادگی با انتخاب دیتابیس بعد از فرمان `mysql>` انجام می شود، برای انتخاب یک دیتابیس از دستور **use** می توانید استفاده کنید.

این یک مثال ساده از انتخاب دیتابیسی با نام **TUTORIALS** است:

مثال :

```
1. [root@host]# mysql -u root -p
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql>
```

در کد فوق دیتابیس TUTORIALS انتخاب شده و تمام عملیات زیر مجموعه نیز روی آن انجام شده است.

**نکته :** تمام نام های پایگاه داده، نام جدول، نام فیلد جدول حساس به حروف هستند. بنابراین شما باید هنگام استفاده از دستورات SQL، از نامهای مناسب استفاده کنید.

### انتخاب پایگاه داده با استفاده از اسکریپت PHP

زبان PHP تابع `mysql_select_db` را برای انتخاب یک دیتابیس ارائه می دهد، این تابع در صورت موفقیت آمیز بودن مقدار TRUE و در غیر این صورت مقدار FALSE را برمی گرداند.

### سینتکس

مثال :

```
1. bool mysql_select_db( db_name, connection );
```

ردیف	پارامتر و توضیحات
۱	<p><b>db_name</b></p> <p>ضروری - نام دیتابیس MySQL انتخاب شده است.</p>
۲	<p><b>connection</b></p> <p>اختیاری - اگر مشخص نشده باشد، آخرین اتصال باز شده توسط mysql_connect مورد استفاده قرار خواهد گرفت.</p>

این مثال چگونگی انتخاب یک دیتابیس را نشان می دهد :

### مثال :

```

1. <html>
2.   <head>
3.     <title>Selecting MySQL Database</title>
4.   </head>
5.
6.   <body>
7.     <?php
8.       $dbhost = 'localhost:3036';
9.       $dbuser = 'guest';
10.      $dbpass = 'guest123';
11.      $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12.
13.      if(! $conn ) {
14.        die('Could not connect: ' . mysql_error());
15.      }
16.      echo 'Connected successfully';
17.      mysql_select_db( 'TUTORIALS' );
18.
19.      mysql_close($conn);
20.    ?>
21.  </body>
22. </html>

```

## کلام آخر

زمانیکه از سیستم مدیریت داده ی MySQL برای ذخیره و مدیریت داده ها استفاده می کنید، غالباً بیش از یک دیتابیس خواهید داشت، بنابراین برای کار روی هر دیتابیس ابتدا باید آن را انتخاب کنید، از این رو در این مبحث به چگونگی **انتخاب دیتابیس در MySQL** پرداخته ایم.



### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. هر دیتابیزی از جدولی تشکیل شده که هر جدول نیز شامل مجموعه ای از فیلدها یا ستون های مرتبط هستند که داده هایی را در خود ذخیره می کنند، نوع داده های فیلدها غالبا با یکدیگر متفاوت هستند، برخی فیلدها داده های عددی ، برخی رشته ها و برخی دیگر ممکن است نوع تاریخ و زمان را ذخیره کنند، مقدار فضای لازم برای ذخیره ی هر کدام از این نوع داده ها با یکدیگر متفاوت است از این رو MySQL به شما اجازه می دهد نوع داده ی فیلدهای جدول خود را تعیین کنید تا با این کار به بهینه سازی فضای هارد و عملکرد دیتابیس کمک کنید. برای این منظور بهتر است تا با **انواع داده در MySQL** آشنا شوید.

### آشنایی با انواع داده در MySQL

در این آموزش شما با انواع داده در MySQL که برای فیلدهای جدول تعریف شده آشنا خواهید شد.

به طور کلی انواع داده در MySQL در سه دسته ی کلی تقسیم می شود :

- Numeric (عددی)
- Date and Time (تاریخ و زمان)
- String Types (انواع رشته ای)

### انواع داده ی عددی

MySQL از تمام استاندارد های انواع عددی ANSI SQL استفاده می کند، بنابراین اگر از یک سیستم پایگاه داده ی دیگر به MySQL کوچ می کنید، داده های عددی برای شما تفاوت چندانی نخواهند داشت.

لیست زیر شامل انواع داده های معمول عددی در MySQL و توضیحات آن هاست:

**INT** : اندازه ی استاندارد نوع عددی integer که می تواند علامتدار یا بدون علامت باشد، اگر علامت دار باشد رنج آن از -۲۱۴۷۴۸۳۶۴۸ تا ۲۱۴۷۴۸۳۶۴۷ و در غیر این صورت از ۰ تا ۴۲۹۴۹۶۷۲۹۵ می باشد.

در این نوع شما می توانید تا ۱۱ رقم را مشخص کنید.

**TINYINT** : کوچکترین اندازه از نوع integer است، که می تواند علامتدار یا بدون علامت باشد.

اگر علامت دار باشد رنج آن از -۱۲۸ تا ۱۲۷ و در غیر این صورت از ۰ تا ۲۵۵ می باشد، در این نوع می توانید تا ۴ رقم را تعریف کنید.

**SMALLINT** : اندازه ی کوچک نوع عددی integer که می تواند علامتدار یا بدون علامت باشد.

اگر علامت دار باشد رنج آن از -۳۲۷۶۸ تا ۳۲۷۶۷ و در غیر این صورت از ۰ تا ۶۵۵۳۵ می باشد، در این نوع می توانید تا ۵ رقم را تعریف کنید.

**MEDIUMINT** : اندازه ی متوسط نوع عددی integer که می تواند علامتدار یا بدون علامت باشد.

اگر علامت دار باشد رنج آن از -۸۳۸۸۶۰۷ تا ۸۳۸۸۶۰۷ و در غیر این صورت از ۰ تا ۱۶۷۷۷۲۱۵ می باشد، در این نوع می توانید تا ۹ رقم را تعریف کنید.

**BIGINT** : اندازه ی بزرگ نوع عددی integer که می تواند علامتدار یا بدون علامت باشد.

اگر علامت دار باشد رنج آن از -۹۲۲۳۳۷۲۰۳۶۸۵۴۷۷۵۸۰۸ تا ۹۲۲۳۳۷۲۰۳۶۸۵۴۷۷۵۸۰۷ و در غیر این صورت از ۰ تا ۱۸۴۴۶۷۴۴۰۷۳۷۰۹۵۵۱۶۱۵ می باشد، در این نوع می توانید تا ۲۰ رقم را تعریف کنید.

**Float(M,D)**: اعداد اعشاری که بدون علامت نیستند و شما می توانید طول نمایش (M) و اعداد دهدهی را تعریف کنید، البته ضروری نیست.

در این نوع می توانید دقت اعشار را تا ۲۴ نقطه تعیین کنید.

**DOUBLE(M,D)**: این نوع شامل اعداد اعشاری است، که بدون علامت نیستند،

در این نوع شما می توانید طول نمایش (M) و اعداد دهدهی را نیز تعریف کنید.

در نوع double شما می توانید تا ۵۳ نقطه اعشار مشخص کنید.

**DECIMAL(M,D)**: این نوع برای ذخیره و بازیابی اعداد دهدهی استفاده می شود.

## انواع date and time (تاریخ و زمان)

انواع داده ی تاریخ و زمان در MySQL به شرح زیر است :

**DATE** : برای ذخیره تاریخ در فرمت YYYY-MM-DD و در رنج بین ۱۰۰۰-۰۱-۰۱ تا ۹۹۹۹-۱۲-۳۱ استفاده می شود، برای مثال December 30<sup>th</sup>, 1973 بصورت ۱۹۷۳-۱۲-۳۰ ذخیره می شود.

**DATETIME** : در این نوع تاریخ زمان در فرمت ترکیبی YYYY-MM-DD HH:MM:SS ذخیره می شود، این نوع در رنج بین ۱۰۰۰-۰۱-۰۱ و ۰۰:۰۰:۰۰ تا ۹۹۹۹-۱۲-۳۱ مقدار می پذیرد.

برای مثال ساعت ۳:۳۰ بعد ازظهر در تاریخ December 30<sup>th</sup>, 1973 به صورت ۱۹۷۳-۱۲-۳۰ ۱۵:۳۰:۰۰ ذخیره می شود.

**TIMESTAMP** : یک بازه ی زمانی بین نیمه شب، ۱ ژانویه ۱۹۷۰، تا سال ۲۰۳۷ می باشد.

این نوع همانند فرمت DATETIME است، فقط فاقد خطوط بین ساعت ۳:۳۰ در تاریخ December 30<sup>th</sup>, 1973 می باشد.

این نوع تاریخ فوق را بصورت ۱۹۷۳۱۲۳۰۱۵۳۰۰۰ ذخیره می کند.

**TIME** : برای ذخیره سازی زمان در فرمت HH:MM:SS استفاده می شود.

**YEAR(M)**: برای ذخیره سازی سال در فرمت ۲ یا ۴ رقمی استفاده می شود.

اگر دو رقمی باشد می تواند در سالهای بین ۱۹۷۰ تا ۲۰۶۹ (رنج ۷۰ تا ۶۹) مقدار پذیرد.

اگر ۴ رقمی باشد نیز می تواند شامل سال های بین ۱۹۰۱ تا ۲۱۵۵ باشد. (طول پیش فرض ۴ رقم است).

## انواع رشته ای

اگر چه در طول کار با دیتابیس با داده های عددی زیاد سر و کار دارید ، اما در اکثر مواقع نیاز به ذخیره مقادیر رشته ای دارید.

این لیست انواع داده ای رشته ای می باشد که در MySQL پشتیبانی می شود :

**( CHAR(M) )**: رشته ها را با طول ثابت بین ۱ تا ۲۵۵ کاراکتر را می پذیرد ، که با تراز راست به چپ و با فضایی به طول مشخص شده ذخیره می شوند.

در این نوع تعیین طول رشته ضروری نیست اما به پیش فرض آن ۱ می باشد.

**( VARCHAR(M) )**: رشته ها را با طول متغیر بین ۱ تا ۲۵۵ کاراکتر را می پذیرد.

برای مثال در **VARCHAR(25)** شما باید یک طول را در زمان تعریف فیلد **VARCHAR** ، تعیین کنید.

**BLOB or TEXT**: یک فیلد با حداکثر طول ۶۵۵۳۵ کاراکتر می باشد، که **BLOB** ها در واقع آبجکت های بزرگ دودویی هستند.

**BLOB** ها برای ذخیره داده های دودویی بزرگ نظیر **image** ها استفاده می شود.

فیدهایی که **TEXT** تعریف می شود همچنین داده های بزرگی را ذخیره می کنند.

تفاوت بین این دو نوع: **BLOB** به حروف بزرگ و کوچک حساس بوده در حالیکه نوع **TEXT** نسبت به حروف حساس نیست.

**TINYBLOB or TINYTEXT**: ستون های **BLOB** یا **TEXT** با حداکثر ۲۵۵ کاراکتر می باشد که در این نوع ها نیازی به تعیین طول نیست.

**MEDIUMBLOB or MEDIUMTEXT**: ستون های **BLOB** یا **TEXT** را با حداکثر ۱۶۷۷۷۲۱۵ کاراکتر را تعریف می کند.

در این دو نوع نیز تعریف طول رشته ها ضروری نیست.

**LOB or LONGTEXT**: ستون های **BLOB** یا **TEXT** را با حداکثر ۴۲۹۴۹۶۷۲۹۵ کاراکتر را تعریف می کند.

در این دو نوع نیز تعریف طول رشته ها ضروری نیست.

**ENUM**: یک ستون شمارشی را ایجاد می کند، هنگام تعریف **ENUM**، شما لیستی از موارد قابل انتخاب را ایجاد می کنید.

به عنوان مثال برای تنظیم فیلد **ENUM** که شامل موارد "A" یا "B" یا "C" باشد، باید به صورت ('A', 'B', 'C') تعریف کنید.

## کلام آخر

در هر یک از دیتابیس های قدرتمند در دنیا انواع داده از جمله **انواع داده در MySQL** برای مشخص کردن نوع داده ای که قرار است در هر فیلد ذخیره شود ارائه گردیده است ، مشخص کردن نوع داده برای هر فیلد در جداول دیتابیس، به بهبود عملکرد برنامه و بهینه سازی فضای هارد سیستم کمک می کند.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بخش های قبلی سینتکس و مراحل ایجاد دیتابیس در MySQL را بررسی کردیم و می دانیم که داده ها در قالب ستون های جداول در دیتابیس ذخیره می شوند، پس گام بعدی ایجاد جداول مورد نیاز به همراه فیلدهای آن ها در دیتابیس می باشد، جداول در واقع ماتریس های دو بعدی هستند که در دیتابیس اطلاعات ذخیره شده ی خود را همانند ساختار صفحات گسترده (Excel) نشان می دهد. در ادامه ی این مبحث با ما در چگونگی [ایجاد جداول در MySQL](#) همراه باشید.

### ایجاد جداول در MySQL

در این آموزش شما با سینتکس پایه و روش های ایجاد جداول در MySQL آشنا خواهید شد، قبل از هر چیز باید بدانید که برای ایجاد جداول باید ابتدا جزئیات زیر را مشخص کنید :

- نام هر جدول
- نام فیلدها
- تعاریف برای هر کدام از فیلدها

### سینتکس

این سینتکس عمومی ایجاد جداول در MySQL می باشد :

مثال :

```
1. CREATE TABLE table_name (column_name column_type);
```

حالا برای نمونه یک جدول در دیتابیس **TUTORIALS** ایجاد می کنیم :

مثال :

```
1. create table tutorials_tbl(  
2.     tutorial_id INT NOT NULL AUTO_INCREMENT,  
3.     tutorial_title VARCHAR(100) NOT NULL,  
4.     tutorial_author VARCHAR(40) NOT NULL,  
5.     submission_date DATE,  
6.     PRIMARY KEY ( tutorial_id )  
7. );
```

در اینجا چند مورد نیاز به توضیح دارند :

- صفت **NOT NULL** را باری فیلدهای جدول تعریف کرده ایم ، چرا که نمی خواهیم فیلدها مقدار Null بگیرند.

بنابراین اگر کاربر رکوردی را با یک مقدار خالی یا NULL بخواهد ذخیره کند، MySQL خطا خواهد داد.

- فیلدی که با صفت **AUTO\_INCREMENT** مشخص شده در واقع آن فیلد را تبدیل به یک شمارنده ی سطر برای جدول می کند.

- کلمه ی کلیدی **PRIMARY KEY** برای یکی از فیلدها مشخص شده که آن فیلد را به عنوان کلید اصلی جدول تعریف می کند.

## ایجاد جدول با استفاده از خط فرمان (Command Prompt)

ایجاد جدول در خط فرمان به راحتی با نوشتن دستور جلوی `mysql>` انجام می شود، برای این منظور باید از دستور **CREATE TABLE** برای ایجاد جدول استفاده کنید.

### مثال

این یک مثال از ایجاد جدول **tutorials\_tbl** است :

### مثال :

```
1. root@host# mysql -u root -p
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> CREATE TABLE tutorials_tbl(
6.     -> tutorial_id INT NOT NULL AUTO_INCREMENT,
7.     -> tutorial_title VARCHAR(100) NOT NULL,
8.     -> tutorial_author VARCHAR(40) NOT NULL,
9.     -> submission_date DATE,
10.    -> PRIMARY KEY ( tutorial_id )
11.    -> );
12. Query OK, 0 rows affected (0.16 sec)
13. mysql>
```

**نکته :** برای مشخص کردن انتهای دستورات در MySQL باید از سمی کولون (;) استفاده کنید.

## ایجاد جدول با استفاده از اسکریپت PHP

برای ایجاد یک جدول جدید در هر کدام از دیتابیس های موجود با استفاده از کدهای PHP باید از تابع `mysql_query()` استفاده کنید، شما باید پارامتر دوم را با دستور SQL مناسب برای ایجاد جدول ارسال کنید.

برنامه ی زیر یک مثال از ایجاد جدول با استفاده از اسکریپت PHP را نشان می دهد :

### مثال :

```
1. <html>
2.   <head>
3.     <title>Creating MySQL Tables</title>
4.   </head>
5.
6.   <body>
7.     <?php
8.       $dbhost = 'localhost:3036';
9.       $dbuser = 'root';
10.      $dbpass = 'rootpassword';
11.      $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12.
13.      if(! $conn ) {
14.        die('Could not connect: ' . mysql_error());
15.      }
16.      echo 'Connected successfully<br />';
```

```

17.      $sql = "CREATE TABLE tutorials_tbl( ".
18.          "tutorial_id INT NOT NULL AUTO_INCREMENT, ".
19.          "tutorial_title VARCHAR(100) NOT NULL, ".
20.          "tutorial_author VARCHAR(40) NOT NULL, ".
21.          "submission_date DATE, ".
22.          "PRIMARY KEY ( tutorial_id )); ";
23.      mysql_select_db( 'TUTORIALS' );
24.      $retval = mysql_query( $sql, $conn );
25.
26.      if( ! $retval ) {
27.          die('Could not create table: ' . mysql_error());
28.      }
29.      echo "Table created successfully\n";
30.      mysql_close($conn);
31.      ?>
32.      </body>
33. </html>

```

## کلام آخر

یکی از ارکان ذخیره و مدیریت داده ها در پایگاه های داده، جداول هستند که انواع داده را در قالب سطری و ستونی ذخیره می کنند ، برای هر گونه ذخیره و مدیریت داده ها در MySQL نیز با با چگونگی **ایجاد جداول در MySQL** نیز آشنایی داشته باشید.

## جلسه ۱۱ : حذف جداول در MySQL

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در مقابل عملیات ایجاد جداول در MySQL با عملیات حذف جداول در MySQL را نیز سر و کار خواهیم داشت، یک دیتابیس می تواند شامل جداول متعددی باشد که به نوعی با هم مرتبط بوده و یا به صورت یک جدول مستقل عمل می کنند، گاهی به دلایل مختلفی از جمله بی استفاده بودن جداول ممکن است تصمیم به حذف یک یا چند جدول بگیرید. برای این منظور باید با چگونگی **حذف جداول در MySQL** آشنایی داشته باشید، که ما در ادامه ی مباحث این بخش به آن پرداخته ایم.

### آشنایی با حذف جداول در MySQL

در این آموزش شما با سینتکس اصلی و روش های حذف جداول در MySQL آشنا خواهید شد.

### سینتکس

سینتکس عمومی SQL برای حذف جداول در MySQL بصورت زیر است :

مثال :

```
1. DROP TABLE table_name ;
```

### حذف جداول با استفاده از خط فرمان

برای حذف جداول با استفاده از command prompt باید از دستور DROP TABLE جوی >mysql استفاده کنید.

در برنامه ی زیر جدول **tutorials\_tbl** حذف می شود :

مثال :

```
1. root@host# mysql -u root -p
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> DROP TABLE tutorials_tbl
6. Query OK, 0 rows affected (0.8 sec)
7. mysql>
```

## حذف جداول با استفاده از اسکریپت PHP

برای حذف جداول موجود در دیتابیس با استفاده از PHP باید از تابع `mysql_query()` استفاده کنید.

شما باید پارامتر دوم را با دستور SQL مناسب برای حذف جدول ارسال کنید.

مثال :

```
1. <html>
2.   <head>
3.     <title>Creating MySQL Tables</title>
4.   </head>
5.
6.   <body>
7.     <?php
8.       $dbhost = 'localhost:3036';
9.       $dbuser = 'root';
10.      $dbpass = 'rootpassword';
11.      $conn = mysql_connect($dbhost, $dbuser, $dbpass);
12.
13.      if(! $conn ) {
14.        die('Could not connect: ' . mysql_error());
15.      }
16.      echo 'Connected successfully<br />';
17.      $sql = "DROP TABLE tutorials_tbl";
18.      mysql_select_db( 'TUTORIALS' );
19.      $retval = mysql_query( $sql, $conn );
20.
21.      if(! $retval ) {
22.        die('Could not delete table: ' . mysql_error());
23.      }
24.      echo "Table deleted successfully\n";
25.      mysql_close($conn);
26.    ?>
27.  </body>
28. </html>
```

## کلام آخر

در بخش قبلی چگونگی ایجاد جداول و فیلدهای آن را در دیتابیس های موجود در را شرح دادیم ؛ در این بخش نیز در تکمیل مباحث بخش قبلی به چگونگی **حذف جداول در MySQL** پرداختیم.



### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. تاکنون با مباحث مهم و پایه در MySQL از جمله ایجاد و حذف دیتابیس و جداول آشنا شدیم، تمام این عملیات برای فراهم کردن زمینه ی ذخیره و مدیریت داده ها در جداول است، پس بعد از آشنایی با چگونگی ایجاد جداول در MySQL، باید با طریقه ی درج رکورد و داده ها در جداول آشنا شوید تا بتوانید داده های مرتبط با هر جدول را در فیلدهای مربوطه ذخیره کنید، برای این هدف باید با ساختار اصلی کار با **دستور INSERT در MySQL** و سینتکس اصلی آن آشنایی داشته باشید که در این خصوص از **دستور INSERT INTO در MySQL** برای درج داده ها استفاده می شود. در ادامه ی مباحث برای آشنایی با این دستور با ما همراه باشید.

### کار با دستور INSERT در MySQL

مطالب این بخش شما را با سینتکس و چگونگی کار با دستور INSERT در MySQL آشنا می کند.

### سینتکس

سینتکس عمومی SQL برای دستور INSERT INTO در MySQL به منظور درج داده در جداول MySQL به صورت زیر است :

#### مثال :

```
1. INSERT INTO table_name ( field1, field2,...fieldN )
2.   VALUES
3.   ( value1, value2,...valueN );
```

برای درج داده های رشته ای حتما باید مقادیر را در تک یا دابل کوتیشن ("value" مانند) قرار دهید.

### درج داده با استفاده از خط فرمان (command prompt)

برای درج داده با استفاده از خط فرمان نیز باید از دستور INSERT INTO در MySQL استفاده کنید.

مثال زیر سه رکورد را در جدول **tutorials\_tbl** ایجاد می کند :

#### مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5.
6. mysql> INSERT INTO tutorials_tbl
7.   ->(tutorial_title, tutorial_author, submission_date)
8.   ->VALUES
9.   ->("Learn PHP", "John Poul", NOW());
10. Query OK, 1 row affected (0.01 sec)
11.
12. mysql> INSERT INTO tutorials_tbl
13.   ->(tutorial_title, tutorial_author, submission_date)
14.   ->VALUES
15.   ->("Learn MySQL", "Abdul S", NOW());
16. Query OK, 1 row affected (0.01 sec)
17.
18. mysql> INSERT INTO tutorials_tbl
```

```

19. ->(tutorial_title, tutorial_author, submission_date)
20. ->VALUES
21. ->("JAVA Tutorial", "Sanjay", '2007-05-06');
22. Query OK, 1 row affected (0.01 sec)
23. mysql>

```

**نکته :** توجه داشته باشید که علامت های (->) بخشی از دستور SQL نیستند، این علامت هنگام زدن کلید Enter و در خط بعدی بصورت خودکار ایجاد می شود.

در کد فوق برای فیلد tutorial\_id مقداری در نظر نگرفته ایم چرا که این فیلد صفت AUTO\_INCREMENT را داراست که برای هر سطر جدید یک شماره اضافه کرده و در این فیلد ذخیره می کند، در واقع این فیلد یک شمارنده ی سطر است.

## درج داده با استفاده از اسکرپت PHP

در اسکرپت های PHP نیز ساختار دستور INSERT در MySQL برای درج داده ها استفاده می شود.

برای این منظور شما می توانید از همان دستور INSERT INTO در MySQL داخل تابع **mysql\_query()** در PHP برای درج داده استفاده کنید.

این مثال سه پارامتر را از کاربر می گیرد و آن ها را در جدول MySQL ذخیره می کند :

مثال :

```

1. <html>
2.
3. <head>
4. </head>
5.
6. <body>
7. <?php
8.     if(isset($_POST['add'])) {
9.         $dbhost = 'localhost:3036';
10.        $dbuser = 'root';
11.        $dbpass = 'rootpassword';
12.        $conn = mysql_connect($dbhost, $dbuser, $dbpass);
13.
14.        if(! $conn ) {
15.            die('Could not connect: ' . mysql_error());
16.        }
17.
18.        if(! get_magic_quotes_gpc() ) {
19.            $tutorial_title = addslashes ($_POST['tutorial_title']);
20.            $tutorial_author = addslashes ($_POST['tutorial_author']);
21.        } else {
22.            $tutorial_title = $_POST['tutorial_title'];
23.            $tutorial_author = $_POST['tutorial_author'];
24.        }
25.
26.        $submission_date = $_POST['submission_date'];
27.
28.        $sql = "INSERT INTO tutorials_tbl ".
29.            "(tutorial_title,tutorial_author, submission_date) ". "VALUES ".
30.            "('$tutorial_title','$tutorial_author','$submission_date')";
31.        mysql_select_db('TUTORIALS');
32.        $retval = mysql_query( $sql, $conn );
33.
34.        if(! $retval ) {

```

```

35.         die('Could not enter data: ' . mysql_error());
36.     }
37.
38.     echo "Entered data successfully\n";
39.     mysql_close($conn);
40. } else {
41.     ?>
42.
43.     <form method = "post" action = "<?php $_PHP_SELF ?>">
44.         <table width = "600" border = "0" cellspacing = "1" cellpadding = "2">
45.             <tr>
46.                 <td width = "250">Tutorial Title</td>
47.                 <td>
48.                     <input name = "tutorial_title" type = "text" id = "tutorial_title">
49.                 </td>
50.             </tr>
51.
52.             <tr>
53.                 <td width = "250">Tutorial Author</td>
54.                 <td>
55.                     <input name = "tutorial_author" type = "text" id = "tutorial_author">
56.                 </td>
57.             </tr>
58.
59.             <tr>
60.                 <td width = "250">Submission Date [   yyyy-mm-dd   ]</td>
61.                 <td>
62.                     <input name = "submission_date" type = "text" id = "submission_date">
63.                 </td>
64.             </tr>
65.
66.             <tr>
67.                 <td width = "250"> </td>
68.                 <td> </td>
69.             </tr>
70.
71.             <tr>
72.                 <td width = "250"> </td>
73.                 <td>
74.                     <input name = "add" type = "submit" id = "add" value = "Add Tutorial">
75.                 </td>
76.             </tr>
77.         </table>
78.     </form>
79. <?php
80. }
81. ?>
82. </body>
83. </html>

```

زمانیکه در حال داده ها درج می کنید بهتر است که با استفاده از تابع **get\_magic\_quotes\_gpc ()** تنظیمات جاری magic quote را چک کنید، اگر تابع فوق مقدار false برگرداند، سپس از تابع **addslashes ()** برای افزودن اسلش قبل از کوتیشن ها استفاده کنید.

## کلام آخر

مباحث گفته شده در بخش های قبلی از جمله روش های ایجاد دیتابیس ، جداول و فیلدهای آن در MySQL برای این انجام می شود تا زمینه برای درج داده ها در دیتابیس فراهم شود، به همین خاطر در این بخش به بررسی طرز کار با **دستور INSERT INTO** در **MySQL** برای درج داده ها پرداختیم.



### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. تاکنون با روش های درج داده ها در فیلدهای جداول آشنا شدیم ، زمانیکه داده هایی در جداول دیتابیس شما درج شود، برای هر گونه عملیات روی داده های موجود در فیلدها، ابتدا باید رکورد یا رکوردهای موجود در جدول مربوطه را انتخاب نموده و سپس عملیات مورد نظر خود را که معمولا شامل عملیات حذف ،ویرایش، بروزرسانی و ... می باشد را اعمال نمائید. برای این منظور در ادامه ی این مباحث چگونگی انتخاب رکوردهای جداول را با استفاده از کار با دستور **SELECT در MySQL** شرح داده ایم.

### دستور SELECT در MySQL

در این آموزش با سینتکس دستور SELECT در MySQL و چگونگی واکشی داده ها با استفاده از این دستور آشنا خواهید شد، دستور SQL SELECT برای جمع آوری داده ها از پایگاه داده MySQL استفاده می شود.

شما می توانید از این دستور در `mysql <prompt` و همچنین در هر اسکریپتی مانند PHP استفاده کنید.

### سینتکس

سینتکس عمومی SQL در دستور SELECT برای واکشی داده ها از جداول MySQL به صورت زیر است :

#### مثال :

```
1. SELECT field1, field2,...fieldN
2. FROM table_name1, table_name2...
3. [WHERE Clause]
4. [OFFSET M ][LIMIT N]
```

- در سینتکس فوق، شما می توانید بیش از یک جدول را با کاما از هم تفکیک کنید، تا بتوانید شرط های مختلفی را اعمال کنید.
- در یک دستور SELECT شما می توانید یک یا بیشتر از یک فیلد را را واکشی کنید.
- شما می توانید به جای نام فیلدها از (\*) استفاده کنید، با این اقدام تمام فیلدهای جدول انتخاب می شوند.
- با استفاده از WHERE می توانید شروط مختلفی را روی واکشی فیلدها اعمال کنید.
- می توانید با استفاده از **OFFSET** یک آفست را از جایی که شروع به بازگرداندن رکوردها می کند را مشخص کنید.

در حالت پیش فرض آفست از صفر شروع می شود.

- با استفاده از صفت **LIMIT** می تواند تعداد نتایج بازگشتی را محدود کنید.

## واکشی داده با استفاده از خط فرمان

واکشی داده ها در MySQL با استفاده از سینتکس دستور SELECT در MySQL انجام می شود.

مثال زیر تمامی رکوردهای جدول **tutorials\_tbl** را برمی گرداند:

مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> SELECT * from tutorials_tbl
6. +-----+-----+-----+-----+
7. | tutorial_id | tutorial_title | tutorial_author | submission_date |
8. +-----+-----+-----+-----+
9. |          1 | Learn PHP      | John Poul      | 2007-05-21      |
10. |          2 | Learn MySQL    | Abdul S        | 2007-05-21      |
11. |          3 | JAVA Tutorial  | Sanjay         | 2007-05-21      |
12. +-----+-----+-----+-----+
13. 3 rows in set (0.01 sec)
14.
15. mysql>
```

## واکشی داده ها با استفاده از اسکریپت PHP

شما می توانید از دستور SELECT داخل تابع **mysql\_query()** در PHP استفاده کنید، تابع فوق برای اجرای دستورات SQL بکار می رود، و سپس با استفاده از تابع **mysql\_fetch\_array()** می توانید داده ها را واکشی کنید.

تابع فوق رکورد یا سطر را به صورت یک آرایه ی associative مرتبط برمی گرداند، سپس شما می توانید با استفاده از آرایه ی associative با استفاده از نام فیلدها به جای index به آن دسترسی داشته باشید.

زبان PHP توابع دیگری از جمله **mysql\_fetch\_assoc()** را ارائه می کند که یک آرایه ی associative را برمی گرداند.

مثال زیر تمام رکوردهای جدول tutorial\_tb را با استفاده از تابع **mysql\_fetch\_assoc()** نمایش می دهد :

مثال :

```
1. <?php
2. $dbhost = 'localhost:3036';
3. $dbuser = 'root';
4. $dbpass = 'rootpassword';
5. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7. if(! $conn ) {
8.     die('Could not connect: ' . mysql_error());
9. }
10.
11. $sql = 'SELECT tutorial_id, tutorial_title, tutorial_author, submission_date
12.     FROM tutorials_tbl';
13.
14. mysql_select_db('TUTORIALS');
15. $retval = mysql_query( $sql, $conn );
16.
17. if(! $retval ) {
18.     die('Could not get data: ' . mysql_error());
19. }
```

```

20.
21. while($row = mysql_fetch_assoc($retval)) {
22.     echo "Tutorial ID :{$row['tutorial_id']} <br> ".
23.         "Title: {$row['tutorial_title']} <br> ".
24.         "Author: {$row['tutorial_author']} <br> ".
25.         "Submission Date : {$row['submission_date']} <br> ".
26.         "-----<br>";
27. }
28. echo "Fetched data successfully\n";
29. mysql_close($conn);
30. ?>

```

شما همچنین می توانید از ثابت **MYSQL\_NUM** به عنوان پارامتر دوم در تابع `mysql_fetch_array()` استفاده کنید، این باعث می شود که تابع فوق یک آرایه را با index های عددی باز کند.

حالا مثال قبلی را با استفاده از ثابت **MYSQL\_NUM** به عنوان آرگومان دوم تابع `mysql_fetch_array()` تکمیل می کنیم:

### مثال :

```

1. <?php
2. $dbhost = 'localhost:3036';
3. $dbuser = 'root';
4. $dbpass = 'rootpassword';
5. $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7. if(! $conn ) {
8.     die('Could not connect: ' . mysql_error());
9. }
10.
11. $sql = 'SELECT tutorial_id, tutorial_title, tutorial_author, submission_date
12.     FROM tutorials_tbl';
13.
14. mysql_select_db('TUTORIALS');
15. $retval = mysql_query( $sql, $conn );
16.
17. if(! $retval ) {
18.     die('Could not get data: ' . mysql_error());
19. }
20.
21. while($row = mysql_fetch_array($retval, MYSQL_NUM)) {
22.     echo "Tutorial ID :{$row[0]} <br> ".
23.         "Title: {$row[1]} <br> ".
24.         "Author: {$row[2]} <br> ".
25.         "Submission Date : {$row[3]} <br> ".
26.         "-----<br>";
27. }
28. echo "Fetched data successfully\n";
29. mysql_close($conn);
30. ?>

```

با استفاده از تابع `mysql_free_result()` در PHP می توانید فضای اشغال شده از حافظه را آزاد کنید.

مثال :

```
1. <?php
2.     $dbhost = 'localhost:3036';
3.     $dbuser = 'root';
4.     $dbpass = 'rootpassword';
5.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7.     if(! $conn ) {
8.         die('Could not connect: ' . mysql_error());
9.     }
10.
11.     $sql = 'SELECT tutorial_id, tutorial_title, tutorial_author, submission_date
12.           FROM tutorials_tbl';
13.
14.     mysql_select_db('TUTORIALS');
15.     $retval = mysql_query( $sql, $conn );
16.
17.     if(! $retval ) {
18.         die('Could not get data: ' . mysql_error());
19.     }
20.
21.     while($row = mysql_fetch_array($retval, MYSQL_NUM)) {
22.         echo "Tutorial ID :{$row[0]} <br> ".
23.             "Title: {$row[1]} <br> ".
24.             "Author: {$row[2]} <br> ".
25.             "Submission Date : {$row[3]} <br> ".
26.             "-----<br>";
27.     }
28.     mysql_free_result($retval);
29.     echo "Fetched data successfully\n";
30.     mysql_close($conn);
31. ?>
```

هنگام واکنشی داده شما می توانید یک کد پیچیده بنویسید، اما ساختار کلی همانند قبل خواهد بود.

## کلام آخر

بعد از درج داده ها در جداول موجود در دیتابیس، برای انجام هر گونه عملیاتی روی داده های ذخیره شده از جمله ویرایش ، حذف ، بروزرسانی و ... باید ابتدا رکوردهای مورد نظر را انتخاب کنید که این کار با استفاده از **دستور SELECT در MySQL** انجام می شود.



### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بخش قبلی طرز استفاده از دستور SELECT را برای استخراج داده ها از دیتابیس را مرور کردیم، همچنین نمونه کدهایی از استخراج تمام داده های یک جدول را بررسی کردیم، اما این ساده ترین حالت است و در بسیاری موارد شما نیاز دارید که فقط بخشی از داده های یک یا چند جدول مرتبط را استخراج کنید، برای این منظور شما نیاز با استفاده از یک دستور شرطی در ادامه ی ساختار SELECT دارید، در این مبحث شما یاد خواهید گرفت که چه طور از **دستور شرطی WHERE در MySQL** برای استخراج رکوردهای مورد نیاز خود استفاده کنید.

### دستور شرطی WHERE در MySQL

در ادامه ی این آموزش سینتکس دستور شرطی WHERE در MySQL و مثال هایی از آن را ارائه کرده ایم.

### سینتکس دستور شرطی WHERE در MySQL

ساختار عمومی دستور SELECT به همراه دستور شرطی WHERE در MySQL به صورت زیر است :

مثال :

1. **SELECT** field1, field2,...fieldN table\_name1, table\_name2...
2. [**WHERE** condition1 [**AND** [**OR**]] condition2.....

- شما می توانید از یک یا چند جدول جدا شده با کاما استفاده کنید تا شامل شرایط مختلف با استفاده از دستور WHERE باشد.
- شما می توانید هر شرطی را با استفاده از دستور WHERE مشخص کنید.
- با استفاده از عملگرهای **AND** یا **OR** می توانید بیش از یک شرط را در دستور WHERE تعریف کنید.
- از دستور شرطی WHERE علاوه بر دستور SELECT می توان در دستورات DELETE یا UPDATE نیز برای تعریف شرط در SQL استفاده کرد.

دستور **WHERE** همانند ساختار شرطی if که در بسیاری از زبان های برنامه نویسی استفاده می شود عمل می کنند، این دستور برای مقایسه مقدار داده شده با مقدار فیلد موجود در یک جدول MySQL استفاده می شود.

اگر مقدار داده شده در دستور WHERE با مقدار فیلد موجود در جدول مربوطه برابر باشد، MySQL آن ردیف را باز می گرداند.

## کاربرد عملگرها در دستور WHERE

لیستی از عملگرهایی که می توان به همراه دستور WHERE استفاده کرد به شرح زیر است ، فرض بر این است که مقدار فیلد A برابر با ۱۰ و مقدار فیلد B نیز برابر با ۲۰ است :

مثال	توضیحات	عملگر
A = B) is not true)	اگر مقدار دو فیلد با هم برابر باشند این عملگر true را برمی گرداند.	=
(A != B) is true)	اگر مقدار دو فیلد با هم برابر نباشند این عملگر true را برمی گرداند.	!=
< A) B) is not true)	اگر مقدار فیلد سمت چپ عملگر از سمت راست آن بیشتر باشد، این عملگر true را برمی گرداند.	<
> A) B) is true)	اگر مقدار فیلد سمت چپ عملگر از سمت راست آن کمتر باشد، این عملگر true را برمی گرداند.	>
A) =< B) is not true)	اگر مقدار فیلد سمت چپ عملگر از سمت راست آن بیشتر یا مساوی آن باشد، این عملگر true را برمی گرداند.	=<

A) => B) is true)	اگر مقدار فیلد سمت چپ عملگر از سمت راست آن کمتر یا مساوی آن باشد، این عملگر true را برمی گرداند.	=>
----------------------------	--	----

دستور WHERE برای وقتی که می خواهید ردیف های انتخاب شده را از یک جدول جدا کنید، بسیار مفید است، مخصوصاً زمانی که از MySQL Join استفاده می کنید.

این یک روش معمول برای جستجوی رکوردها با استفاده از کلید اصلی برای جستجوی سریعتر است، اگر شرط داده شده با هیچ رکوردی در جدول مطابقت نداشته باشد، پس از آن پرس و جو هیچ ردیفی بازگردانده نمی شود.

## واکشی داده ها با استفاده از خط فرمان

در این بخش یک نمونه کد از دستور SELECT به همراه شرط WHERE ارائه شده است.

مثال زیر تمام رکوردها را از جدول **tutorials\_tbl** که فیلد name آن ها با **Sanjay** برابر باشد را بازمی گرداند:

مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> SELECT * from tutorials_tbl WHERE tutorial_author = 'Sanjay';
6. +-----+-----+-----+-----+
7. | tutorial_id | tutorial_title | tutorial_author | submission_date |
8. +-----+-----+-----+-----+
9. | 3 | JAVA Tutorial | Sanjay | 2007-05-21 |
10. +-----+-----+-----+-----+
11. 1 rows in set (0.01 sec)
12.
13. mysql>
```

به استثنای دستور **LIKE** در سایر موارد مقایسه در SQL، بزرگی یا کوچکی حروف مهم نیست، البته شما می توانید با استفاده از کلید **BINARY** بصورت زیر، حساسیت به حروف را در Query خود اعمال کنید:

مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> SELECT * from tutorials_tbl \
6. WHERE BINARY tutorial_author = 'sanjay';
7. Empty set (0.02 sec)
8.
9. mysql>
```

## واکشی داده ها با استفاده از اسکرپت PHP

شما می توانید از دستورات SQL به همراه دستور شرطی WHERE در تابع `mysql_query()` در PHP استفاده کنید، این تابع برای اجرای دستورات SQL استفاده می شود و برای واکشی داده ها باید از تابع `mysql_fetch_array()` استفاده کرد.

تابع فوق یک سطر از جدول را در قالب آرایه ی وابسته ، آرایه ی عددی یا هر دو مورد بر می گرداند، اگر سطرهای بیشتری وجود نداشته باشد این تابع مقدار FALSE را برمی گرداند.

مثال زیر تمام رکوردها را از جدول `tutorials_tbl` که نام آن ها با **Sanjay** برابر است را برمی گرداند:

مثال :

```
1. <?php
2.     $dbhost = 'localhost:3036';
3.     $dbuser = 'root';
4.     $dbpass = 'rootpassword';
5.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7.     if(! $conn ) {
8.         die('Could not connect: ' . mysql_error());
9.     }
10.
11.     $sql = 'SELECT tutorial_id, tutorial_title,
12.           tutorial_author, submission_date
13.           FROM tutorials_tbl
14.           WHERE tutorial_author = "Sanjay"';
15.
16.     mysql_select_db('TUTORIALS');
17.     $retval = mysql_query( $sql, $conn );
18.
19.     if(! $retval ) {
20.         die('Could not get data: ' . mysql_error());
21.     }
22.
23.     while($row = mysql_fetch_array($retval, MYSQL_ASSOC)) {
24.         echo "Tutorial ID :{$row['tutorial_id']} <br> ".
25.             "Title: {$row['tutorial_title']} <br> ".
26.             "Author: {$row['tutorial_author']} <br> ".
27.             "Submission Date : {$row['submission_date']} <br> ".
28.             "-----<br>";
29.     }
30.
31.     echo "Fetched data successfully\n";
32.     mysql_close($conn);
33. ?>
```

## کلام آخر

دستور SELECT به تنهایی فقط تمام داده های موجود در یک جدول یا چند را باز می گرداند، در بسیاری از موارد شما نیاز خواهید داشت که فقط تعدادی از رکوردها بازگردانده شود، برای این منظور **دستور شرطی WHERE در MySQL** شما را به هدفتان می رساند.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت **پی وی لرن** ، و کاربرانی که **دوره آموزش MySQL** را دنبال می کنند. در بخش قبلی کاربرد دستور شرطی مهم WHERE را بررسی کردیم، و اشاره کردیم که این دستور در ساختار UPDATE نیز بسیار استفاده می شود، دستور UPDATE در MySQL برای ویرایش و بروزرسانی فیلدهای انتخابی از جداول در MySQL استفاده می شود. یکی از استفاده های رایج این دستور، ایجاد قابلیت ویرایش برخی از اطلاعات برای کاربران است. در ادامه ی این بخش ما شما را با چگونگی کار با **دستور UPDATE در MySQL** آشنا خواهیم کرد.

### دستور UPDATE در MySQL

در ادامه ی این آموزش سینتکس دستور UPDATE در MySQL و مثال هایی از آن را ارائه کرده ایم.

### سینتکس دستور UPDATE در MySQL

ساختار عمومی سینتکس دستور UPDATE در MySQL به صورت زیر است :

مثال :

1. **UPDATE** table\_name **SET** field1 = new-value1, field2 = new-value2
2. [**WHERE** Clause]

- شما می توانید یک یا چند فیلد را به طور کامل به روز کنید.
- شما می توانید هر شرطی را با استفاده از دستور WHERE در ساختار UPDATE مشخص کنید.
- شما می توانید تمام مقادیر را در یک جدول تنها در یک زمان به روز کنید.

دستور WHERE برای زمانی که می خواهید فقط بخشی از رکوردهای یک جدول را بروزرسانی کنید بسیار کاربرد دارد.

### بروزرسانی داده ها با استفاده از خط فرمان

در این بخش یک نمونه کد از دستور UPDATE به همراه شرط WHERE ارائه شده است، در مثال زیر فیلد **tutorial\_title** در یک رکوردی که **tutorial\_id** آن مقدار ۳ دارد آپدیت می شود:

مثال :

- ```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3.
4. mysql> use TUTORIALS;
5. Database changed
6.
7. mysql> UPDATE tutorials_tbl
8.     -> SET tutorial_title = 'Learning JAVA'
9.     -> WHERE tutorial_id = 3;
10. Query OK, 1 row affected (0.04 sec)
11. Rows matched: 1  Changed: 1  Warnings: 0
12.
13. mysql>
```

## بروزرسانی داده ها با استفاده از یک اسکرپت PHP

شما می توانید از دستور UPDATE به همراه WHERE یا بدون آن در تابع `mysql_query()` در PHP استفاده کنید، تابع فوق ، دستورات SQL را همانند روش استفاده از خط فرمان `mysql>` اجرا می کند.

در مثال زیر فیلد `tutorial_title` در یک رکوردی که `tutorial_id` آن مقدار ۳ دارد آپدیت می شود:

مثال :

```
1. <?php
2.     $dbhost = 'localhost:3036';
3.     $dbuser = 'root';
4.     $dbpass = 'rootpassword';
5.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7.     if(! $conn ) {
8.         die('Could not connect: ' . mysql_error());
9.     }
10.
11.     $sql = 'UPDATE tutorials_tbl
12.         SET tutorial_title="Learning JAVA"
13.         WHERE tutorial_id=3';
14.
15.     mysql_select_db('TUTORIALS');
16.     $retval = mysql_query( $sql, $conn );
17.
18.     if(! $retval ) {
19.         die('Could not update data: ' . mysql_error());
20.     }
21.     echo "Updated data successfully\n";
22.     mysql_close($conn);
23. ?>
```

## کلام آخر

داده های درج شده در جداول MySQL، قطعا در مواردی باید امکان ویرایش و تغییر توسط کاربران را داشته باشند، که چگونگی ایجاد این قابلیت برای کاربران، با استفاده از **دستور UPDATE در MySQL** در مباحث فوق ارائه شد.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت **پی وی لرن**، و کاربرانی که **دوره آموزش MySQL** را دنبال می کنند. همانطور که در بخش قبلی هم اشاره کردیم، داده های درج شده در جداول MySQL قطعا در برخی موارد باید توسط برنامه نویس ویرایش شده و یا توسط کاربران مختلف قابلیت ویرایش را داشته باشند، اما در مواردی نیز این داده ها هیچ استفاده ای ندارند و یا به دلایلی دیگری از جمله حذف برخی از قابلیت های برنامه ها باید از دیتابیس نیز حذف شوند. برای حذف داده های موجود در جداول MySQL، می توان از کار با **دستور DELETE در MySQL** استفاده کرد، که در ادامه ی مباحث به آن پرداخته ایم.

### دستور DELETE در MySQL

در ادامه ی این آموزش سینتکس دستور DELETE در MySQL و مثال هایی از آن را ارائه کرده ایم.

### سینتکس دستور DELETE در MySQL

ساختار عمومی سینتکس دستور DELETE در MySQL به صورت زیر است :

مثال :

1. **DELETE FROM** table\_name [**WHERE** Clause]

- اگر شرط WHERE مشخص نشده باشد، تمام رکوردها از جدول داده MySQL حذف می شوند.
- شما می توانید هر شرطی را با استفاده از دستور WHERE مشخص کنید.
- شما می توانید سوابق را در یک جدول تنها در یک زمان حذف کنید.

دستور WHERE برای زمانی که می خواهید فقط بخشی از رکوردهای یک جدول را حذف کنید بسیار کاربرد دارد.

### حذف داده با استفاده از خط فرمان (Command Prompt)

در این بخش یک نمونه کد از دستور DELETE به همراه شرط WHERE ارائه شده است، در مثال زیر یک رکورد از جدول tutorial\_tbl که فیلد tutorial\_id در آن رکورد مقدار ۳ دارد، حذف می شود.

مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3.
4. mysql> use TUTORIALS;
5. Database changed
6.
7. mysql> DELETE FROM tutorials_tbl WHERE tutorial_id=3;
8. Query OK, 1 row affected (0.23 sec)
9.
10. mysql>
```

## حذف داده با استفاده از اسکرپت PHP

شما می توانید از دستور DELETE زبان SQL به همراه دستور شرطی WHERE یا بدون آن در تابع `mysql_query()` در PHP استفاده کنید.

تابع فوق ، دستورات SQL را همانند روش استفاده از خط فرمان `mysql>` اجرا می کند، در مثال زیر یک رکورد از جدول `tutorial_tbl` که فیلد `tutorial_id` در آن رکورد مقدار ۳ دارد، حذف می شود:

مثال :

```
1. <?php
2.     $dbhost = 'localhost:3036';
3.     $dbuser = 'root';
4.     $dbpass = 'rootpassword';
5.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7.     if(! $conn ) {
8.         die('Could not connect: ' . mysql_error());
9.     }
10.
11.     $sql = 'DELETE FROM tutorials_tbl WHERE tutorial_id = 3';
12.
13.     mysql_select_db('TUTORIALS');
14.     $retval = mysql_query( $sql, $conn );
15.
16.     if(! $retval ) {
17.         die('Could not delete data: ' . mysql_error());
18.     }
19.     echo "Deleted data successfully\n";
20.     mysql_close($conn);
21. ?>
```

## کلام آخر

قطعا همانطور که داده های موجود در جداول در مواردی باید ویرایش و بروزرسانی شوند، در برخی موارد نیز داده هایی باید از برخی جداول حذف شوند و یا اینکه قابلیت حذف آن ها برای کاربران ایجاد شود، که برای اینگونه موارد بهترین راهکار استفاده از **دستور DELETE در MySQL** است.



### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بخش های قبلی با دستور شرطی WHERE در MySQL آشنا شدیم که امکان انتخاب بخشی از رکوردهای یک یا چند جدول را در MySQL را فراهم می کند، اما در برخی موارد ما نیاز داریم که انتخاب دقیق تری انجام دهیم، به عنوان مثال شاید بخواهیم که رکوردهایی که ابتدای آن با یک یا چند حرف مشخص شروع شده و یا شامل آن ها باشد را انتخاب کنیم، برای اینگونه موارد می توانیم از **دستور شرطی LIKE در MySQL** استفاده کنیم، که ادامه ی این مبحث طرز استفاده از **دستور LIKE در MySQL** را بررسی می کنیم.

### دستور شرطی LIKE در MySQL

در ادامه ی این آموزش سینتکس دستور LIKE در MySQL و مثال هایی از آن را ارائه کرده ایم، دستور WHERE عملکرد عملگرهای از جمله = را پیاده سازی می کند، و تمام مقادیر یک فیلد را مطابقت می دهد.

اما دستور شرطی LIKE در MySQL برای فیلتر کردن یک جستجو بسیار مناسب است، استفاده از دستور LIKE در MySQL به همراه کاراکتر % کار فیلتر جستجوها را انجام می دهد.

### سینتکس دستور LIKE در MySQL

ساختار عمومی سینتکس دستور DELETE در MySQL برای استخراج داده ها از جداول به صورت زیر است :

مثال :

```
1. SELECT field1, field2,...fieldN table_name1, table_name2...
2. WHERE field1 LIKE condition1 [AND [OR]] filed2 = 'somevalue'
```

- شما می توانید هر شرطی را با استفاده از دستور WHERE مشخص کنید.
- شما می توانید از دستور LIKE به همراه شرط WHERE استفاده کنید.
- شما می توانید از عبارت LIKE به جای **equals to** استفاده کنید.
- زمانیکه از دستور LIKE به همراه کاراکتر % استفاده شود، این همانند جستجوی کاراکتر متا عمل می کند.
- با استفاده از عملگرهای **AND** یا **OR** می توانید بیش از یک شرط را در دستور WHERE تعریف کنید.
- یک ساختار شرطی WHERE...LIKE را می توان در دستورات DELETE یا UPDATE برای مشخص کردن شرط استفاده کرد.

### استفاده از دستور LIKE در خط فرمان (Command Prompt)

در این بخش یک نمونه کد از دستور شرطی LIKE در MySQL به همراه شرط WHERE ارائه شده است، در مثال زیر تمام رکوردها از جدول **tutorials\_tbl** که فیلد name آن ها با **jay** پایان می یابد، انتخاب می شوند:

مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> SELECT * from tutorials_tbl
6. -> WHERE tutorial_author LIKE '%jay';
7. +-----+-----+-----+-----+
8. | tutorial_id | tutorial_title | tutorial_author | submission_date |
```

```

9. +-----+-----+-----+-----+
10. |      3      | JAVA Tutorial | Sanjay      | 2007-05-21 |
11. +-----+-----+-----+-----+
12. 1 rows in set (0.01 sec)
13.
14. mysql>

```

## استفاده از دستور LIKE در اسکریپت PHP

شما می توانید از سینتکس WHERE...LIKE در تابع `mysql_query()` در زبان PHP استفاده کنید.

تابع فوق برای اجرای دستورات SQL استفاده می شود و از تابع `mysql_fetch_array()` برای استخراج تمام داده ها استفاده می شود.

اگر ساختار WHERE ... LIKE با دستور DELETE یا UPDATE مورد استفاده قرار بگیرد، پس هیچ تابع فراخوانی PHP دیگری مورد نیاز است.

در مثال زیر تمام رکوردها از جدول `tutorials_tbl` که فیلد name آن ها شامل **jay** باشد، انتخاب می شوند:

### مثال :

```

1. <?php
2.     $dbhost = 'localhost:3036';
3.     $dbuser = 'root';
4.     $dbpass = 'rootpassword';
5.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7.     if(! $conn ) {
8.         die('Could not connect: ' . mysql_error());
9.     }
10.    $sql = 'SELECT tutorial_id, tutorial_title,
11.           tutorial_author, submission_date
12.           FROM tutorials_tbl
13.           WHERE tutorial_author LIKE "%jay%";
14.
15.    mysql_select_db('TUTORIALS');
16.    $retval = mysql_query( $sql, $conn );
17.
18.    if(! $retval ) {
19.        die('Could not get data: ' . mysql_error());
20.    }
21.
22.    while($row = mysql_fetch_array($retval, MYSQL_ASSOC)) {
23.        echo "Tutorial ID :{$row['tutorial_id']} <br> ".
24.            "Title: {$row['tutorial_title']} <br> ".
25.            "Author: {$row['tutorial_author']} <br> ".
26.            "Submission Date : {$row['submission_date']} <br> ".
27.            "-----<br>";
28.    }
29.    echo "Fetched data successfully\n";
30.    mysql_close($conn);
31. ?>

```

## کلام آخر

برای جستجوهای کلی در جداول MySQL، دستور WHERE گزینه ی مناسبی است، اما برای جستجوهای دقیق و جزئی تر می توان از دستور شرطی **LIKE در MySQL** استفاده کرد.

## جلسه ۱۸ : مرتب سازی نتایج در MySQL

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. تا الان دستوراتی را بررسی کردیم که امکان انتخاب ، ویرایش ، بروزرسانی و حذف داده های جداول MySQL را فراهم می کند. اما در حالت عادی نتایج در این دستورات بدون ترتیب خاصی برگردانده می شوند، زمانیکه شما می خواهید نتایج یک Query را در یک ترتیب خاص دریافت کنید، می توانید از دستور **ORDER BY در MySQL** استفاده کنید. در ادامه ی این مبحث چگونگی **مرتب سازی نتایج در MySQL** را با استفاده از دستور فوق توضیح داده ایم.

### سینتکس دستور ORDER BY در MySQL

سینتکس عمومی دستور ORDER BY برای مرتب سازی نتایج در MySQL برای استخراج داده ها از جداول به صورت زیر است :

#### مثال :

```
1. SELECT field1, field2,...fieldN table_name1, table_name2...
2. ORDER BY field1, [field2...] [ASC [DESC]]
```

- شما می توانید نتیجه بازگشتی را در هر فیلد مرتب کنید، در صورتی که این فیلد لیست شده باشد.
- شما می توانید نتیجه را در بیش از یک فیلد مرتب کنید.
- شما می توانید از کلمه کلیدی ASC یا DESC استفاده کنید تا نتیجه را به ترتیب صعودی یا نزولی بدست آورید. به صورت پیش فرض، این ترتیب صعودی است.
- شما می توانید از ساختار شرطی WHERE ... LIKE نیز استفاده کنید.

### کار با دستور ORDER BY برای مرتب سازی نتایج در MySQL در خط فرمان

در این بخش یک نمونه کد از دستور ORDER BY در MySQL به همراه شرط WHERE ارائه شده است، مثال زیر تمام نتایج را به صورت صعودی مرتب سازی می کند :

#### مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> SELECT * from tutorials_tbl ORDER BY tutorial_author ASC
6. +-----+-----+-----+-----+
7. | tutorial_id | tutorial_title | tutorial_author | submission_date |
8. +-----+-----+-----+-----+
9. | 2 | Learn MySQL | Abdul S | 2007-05-24 |
10. | 1 | Learn PHP | John Poul | 2007-05-24 |
11. | 3 | JAVA Tutorial | Sanjay | 2007-05-06 |
12. +-----+-----+-----+-----+
13. 3 rows in set (0.42 sec)
14.
15. mysql>
```

## استفاده از دستور ORDER BY در اسکرپت PHP

شما می توانید از دستور ORDER BY در تابع `mysql_query()` در زبان PHP استفاده کنید، برای استخراج داده ها نیز می توانید از دستور `mysql_fetch_array()` استفاده کنید.

اینبار مثال قبلی را به صورت نزولی مرتب می کنیم :

مثال :

```
1. <?php
2.     $dbhost = 'localhost:3036';
3.     $dbuser = 'root';
4.     $dbpass = 'rootpassword';
5.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7.     if(! $conn ) {
8.         die('Could not connect: ' . mysql_error());
9.     }
10.    $sql = 'SELECT tutorial_id, tutorial_title,
11.           tutorial_author, submission_date
12.           FROM tutorials_tbl
13.           ORDER BY tutorial_author DESC';
14.
15.    mysql_select_db('TUTORIALS');
16.    $retval = mysql_query( $sql, $conn );
17.
18.    if(! $retval ) {
19.        die('Could not get data: ' . mysql_error());
20.    }
21.
22.    while($row = mysql_fetch_array($retval, MYSQL_ASSOC)) {
23.        echo "Tutorial ID :{$row['tutorial_id']} <br> ".
24.            "Title: {$row['tutorial_title']} <br> ".
25.            "Author: {$row['tutorial_author']} <br> ".
26.            "Submission Date : {$row['submission_date']} <br> ".
27.            "-----<br>";
28.    }
29.    echo "Fetched data successfully\n";
30.    mysql_close($conn);
31. ?>
```

## کلام آخر

نتایجی که از پرس و جویهای مختلف در بدست می آورید، در حالت پیش فرض به صورت صعودی مرتب شده اند ، اما در صورتی که شما قصد تغییر این حالت را به نزولی داشته باشید ، به راحتی می توانید از دستور **ORDER BY در MySQL** استفاده کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در مثال های بخش های قبلی، ما داده ها را فقط از یک جدول استخراج کرده و یا دستکاری می کردیم. در حالت عادی و در یک دستور SQL فقط می توانیم روی داده های موجود در یک جدول کار کنیم، اما برای انتخاب و یا دستکاری همزمان داده های چند جدول در یک دستور SQL یک راهکار ساده و پرمکاربرد وجود دارد و آن استفاده از دستور JOIN در MySQL است، این دستور به شما امکان تعریف چندین جدول در یک خط کد SQL را برای اعمال مختلف می دهد، که در ادامه ی مبحث به کار با **دستور JOIN در MySQL** آشنا خواهیم شد.

### دستور JOIN در MySQL

در این آموزش شما را با طرز کار و روش های استفاده از دستور JOIN در MySQL آشنا خواهیم کرد، شما می توانید از دستور JOIN در MySQL در دستورات SELECT, UPDATE و DELETE برای دستکاری همزمان چند جدول استفاده کنید.

### استفاده از دستور JOIN در خط فرمان

ما فرض می کنیم که دو جدول **tcount\_tbl** و **tutorials\_tbl** را در TUTORIALS داریم، مثال زیر را در نظر بگیرید:

مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> SELECT * FROM tcount_tbl;
6. +-----+-----+
7. | tutorial_author | tutorial_count |
8. +-----+-----+
9. | mahran         | 20             |
10. | mahnaz         | NULL           |
11. | Jen            | NULL           |
12. | Gill           | 20             |
13. | John Poul      | 1              |
14. | Sanjay         | 1              |
15. +-----+-----+
16. 6 rows in set (0.01 sec)
17. mysql> SELECT * from tutorials_tbl;
18. +-----+-----+-----+-----+
19. | tutorial_id | tutorial_title | tutorial_author | submission_date |
20. +-----+-----+-----+-----+
21. | 1          | Learn PHP     | John Poul      | 2007-05-24      |
22. | 2          | Learn MySQL   | Abdul S        | 2007-05-24      |
23. | 3          | JAVA Tutorial | Sanjay         | 2007-05-06      |
24. +-----+-----+-----+-----+
25. 3 rows in set (0.00 sec)
26. mysql>
```

حالا ما می توانیم یک query از SQL بنویسیم که دو جدول فوق را به هم پیوند، این query تمام نویسندگان (author) را از جدول **tutorials\_tbl** انتخاب کرده و tutorial مرتبط با آن ها را از جدول **tcount\_tbl** انتخاب می کند:

مثال :

```
1. mysql> SELECT a.tutorial_id, a.tutorial_author, b.tutorial_count
2.     -> FROM tutorials_tbl a, tcount_tbl b
3.     -> WHERE a.tutorial_author = b.tutorial_author;
4. +-----+-----+-----+
5. | tutorial_id | tutorial_author | tutorial_count |
6. +-----+-----+-----+
7. |          1 |      John Poul |              1 |
8. |          3 |       Sanjay   |              1 |
9. +-----+-----+-----+
10. 2 rows in set (0.01 sec)
11. mysql>
```

## استفاده از دستور JOIN در اسکریپت PHP

همانطور که قبلا هم اشاره کردیم برای اجرای هر یک از queryهای SQL در PHP از تابع **mysql\_query()** استفاده می شود و سپس می توانیم از توابع دیگری مانند **mysql\_fetch\_array** نتایج را استخراج کنیم.

مثال زیر را در نظر بگیرید:

مثال :

```
1. <?php
2.     $dbhost = 'localhost:3036';
3.     $dbuser = 'root';
4.     $dbpass = 'rootpassword';
5.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7.     if(! $conn ) {
8.         die('Could not connect: ' . mysql_error());
9.     }
10.
11.     $sql = 'SELECT a.tutorial_id, a.tutorial_author, b.tutorial_count
12.           FROM tutorials_tbl a, tcount_tbl b
13.           WHERE a.tutorial_author = b.tutorial_author';
14.
15.     mysql_select_db('TUTORIALS');
16.     $retval = mysql_query( $sql, $conn );
17.
18.     if(! $retval ) {
19.         die('Could not get data: ' . mysql_error());
20.     }
21.
22.     while($row = mysql_fetch_array($retval, MYSQL_ASSOC)) {
23.         echo "Author:{$row['tutorial_author']} <br> ".
24.             "Count: {$row['tutorial_count']} <br> ".
25.             "Tutorial ID: {$row['tutorial_id']} <br> ".
26.             "-----<br>";
27.     }
28.     echo "Fetched data successfully\n";
29.     mysql_close($conn);
30. ?>
```

## دستور LEFT JOIN در MySQL

یک دستور MySQL LEFT JOIN با یک دستور join ساده فرق دارد، این دستور به جدولی که در سمت چپ است بیشتر توجه می کند.

اگر ما از دستور **LEFT JOIN** استفاده کنیم، تمام رکوردهای دو جدول که با هم مطابقت دارند استخراج می شود و در نهایت رکوردهای باقی مانده از جدول سمت چپ که مطابقت نداشته اند نیز به نتایج اضافه می شوند.

مثال زیر درک شما را از عملکرد **LEFT JOIN** راحت تر می کند :

مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3. mysql> use TUTORIALS;
4. Database changed
5. mysql> SELECT a.tutorial_id, a.tutorial_author, b.tutorial_count
6.     -> FROM tutorials_tbl a LEFT JOIN tcount_tbl b
7.     -> ON a.tutorial_author = b.tutorial_author;
8. +-----+-----+-----+
9. | tutorial_id | tutorial_author | tutorial_count |
10. +-----+-----+-----+
11. | 1          | John Poul      | 1             |
12. | 2          | Abdul S       | NULL          |
13. | 3          | Sanjay        | 1             |
14. +-----+-----+-----+
15. 3 rows in set (0.02 sec)
```

## کلام آخر

در اغلب موارد که چندین جدول مرتبط در یک دیتابیس داریم ، اغلب پیش خواهد آمد که نیاز دارید از داده های دو یا چند جدول هزمان در یک خروجی استفاده کنید برای اینگونه موارد می توانیم از **دستور JOIN در MySQL** به صورتی که در مباحث فوق اشاره شد، استفاده کنیم.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت **پی وی لرن** ، و کاربرانی که **دوره آموزش MySQL** را دنبال می کنند. در بخش های قبلی چگونگی استخراج داده ها از جداول MySQL را بررسی کردیم و روش انجام اینکار آشنا شدیم، اما زمانیکه شما از دستورات شرطی مانند WHERE برای استخراج بخشی از رکوردها از جداول استفاده می کنید، اگر مقادیر رکوردها NULL باشد ممکن است نتایج بازگشتی بدرستی انجام نشده باشد. برای جلوگیری از اینگونه مشکلات باید با چگونگی مدیریت **مقادیر NULL در MySQL** آشنایی داشته باشید که ما در ادامه ی این آموزش به آن پرداخته ایم.

### مقادیر NULL در MySQL

در ادامه ی این مبحث شما را با دستورات کنترل مقادیر NULL در MySQL آشنا خواهیم کرد.

برای مدیریت مقادیر NULL در MySQL سه عملگر ارائه شده است :

- **IS NULL** : این عملگر در صورتی که فیلد مقدار NULL داشته باشد، true برمی گرداند.
- **IS NOT NULL** : این عملگر در صورتی که فیلد مقدار NULL نداشته باشد، true برمی گرداند.
- **<=>** : مقادیر دو فیلد را با هم مقایسه می کند و در صورتی که هر دو مقدار NULL داشته باشند، true برمی گرداند.

دقت کنید که برای تست NULL بودن یک فیلد نمی توان از روش هایی مانند **NULL =** یا **NULL !=** استفاده کرد و اگر استفاده کنید با خطا مواجه می شوید.

### استفاده از مقادیر NULL در خط فرمان

فرض می کنیم که یک جدول داریم که **tcount\_tbl** نامیده می شود و در دیتابیس TUTORIALS قرار دارد، همچنین جدول فوق شامل دو فیلد به نام های **tutorial\_author** و **tutorial\_count** می باشد.

مثال زیر را در نظر بگیرید:

#### مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3.
4. mysql> use TUTORIALS;
5. Database changed
6.
7. mysql> create table tcount_tbl
8.   -> (
9.     -> tutorial_author varchar(40) NOT NULL,
10.    -> tutorial_count INT
11.    -> );
12. Query OK, 0 rows affected (0.05 sec)
13.
14. mysql> INSERT INTO tcount_tbl
15.   -> (tutorial_author, tutorial_count) values ('mahran', 20);
16.
17. mysql> INSERT INTO tcount_tbl
18.   -> (tutorial_author, tutorial_count) values ('mahnaz', NULL);
19.
20. mysql> INSERT INTO tcount_tbl
21.   -> (tutorial_author, tutorial_count) values ('Jen', NULL);
```



```

22.
23. mysql> INSERT INTO tcount_tbl
24.     -> (tutorial_author, tutorial_count) values ('Gill', 20);
25.
26. mysql> SELECT * from tcount_tbl;
27. +-----+-----+
28. | tutorial_author | tutorial_count |
29. +-----+-----+
30. | mahran         | 20             |
31. | mahnaz         | NULL           |
32. | Jen            | NULL           |
33. | Gill           | 20             |
34. +-----+-----+
35. 4 rows in set (0.00 sec)
36.
37. mysql>

```

می توانید ببینید که عملگرهای = و != برای تست مقادیر NULL کارایی ندارند :

### مثال :

```

1. mysql> SELECT * FROM tcount_tbl WHERE tutorial_count = NULL;
2. Empty set (0.00 sec)
3.
4. mysql> SELECT * FROM tcount_tbl WHERE tutorial_count != NULL;
5. Empty set (0.01 sec)

```

برای پیدا کردن رکوردهایی که در فیلد tutorial\_count مقدار NULL را دارا باشند و یا نباشند، می توانید برنامه را به صورت زیر بنویسید:

### مثال :

```

1. mysql> SELECT * FROM tcount_tbl
2.     -> WHERE tutorial_count IS NULL;
3. +-----+-----+
4. | tutorial_author | tutorial_count |
5. +-----+-----+
6. | mahnaz         | NULL           |
7. | Jen            | NULL           |
8. +-----+-----+
9. 2 rows in set (0.00 sec)
10. mysql> SELECT * from tcount_tbl
11.     -> WHERE tutorial_count IS NOT NULL;
12. +-----+-----+
13. | tutorial_author | tutorial_count |
14. +-----+-----+
15. | mahran         | 20             |
16. | Gill           | 20             |
17. +-----+-----+
18. 2 rows in set (0.00 sec)

```

## مدیریت مقادیر NULL در اسکریپت PHP

برای مدیریت مقادیر NULL در اسکریپت PHP می توانید از ساختار شرطی **if...else** استفاده کنید، مثال زیر tutorial\_count را از بیرون می گیرد و سپس آن را با مقدار موجود در جدول مقایسه می کند:

مثال :

```
1. <?php
2.     $dbhost = 'localhost:3036';
3.     $dbuser = 'root';
4.     $dbpass = 'rootpassword';
5.     $conn = mysql_connect($dbhost, $dbuser, $dbpass);
6.
7.     if(! $conn ) {
8.         die('Could not connect: ' . mysql_error());
9.     }
10.
11.     if( isset($tutorial_count) ) {
12.         $sql = 'SELECT tutorial_author, tutorial_count
13.             FROM tcount_tbl
14.             WHERE tutorial_count = $tutorial_count';
15.     } else {
16.         $sql = 'SELECT tutorial_author, tutorial_count
17.             FROM tcount_tbl
18.             WHERE tutorial_count IS $tutorial_count';
19.     }
20.
21.     mysql_select_db('TUTORIALS');
22.     $retval = mysql_query( $sql, $conn );
23.
24.     if(! $retval ) {
25.         die('Could not get data: ' . mysql_error());
26.     }
27.
28.     while($row = mysql_fetch_array($retval, MYSQL_ASSOC)) {
29.         echo "Author:{$row['tutorial_author']} <br> ".
30.             "Count: {$row['tutorial_count']} <br> ".
31.             "-----<br>";
32.     }
33.     echo "Fetched data successfully\n";
34.     mysql_close($conn);
35. ?>
```

## کلام آخر

فیلدهایی که در هنگام درج داده ها در دیتابیس مقدار نمی گیرند، مقدار پیش فرض NULL را خواهند گرفت، پس بهتر است در هنگام استخراج داده ها از جداول دیتابیس با استفاده از دستورات مدیریتی **مقادیر NULL در MySQL** ابتدا از NULL بودن با نبودن فیلدها اطمینان حاصل کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بخش های قبلی با طرز استفاده از عملگر شرطی `%...LIKE` برای جستجوهای دقیق تر و فیلتر بیشتر در جداول با استفاده از الگوهای تطبیقی مانند `%` آشنا شدیم، با استفاده از این الگوها می توانید به عنوان مثال فیلدهایی را که با یک مقدار خاص شروع شده و یا شامل یک مقداری می شوند و ... را استخراج کنید، اما گاهی ممکن است که به فیلترهای دقیق تر و استفاده از الگوهای دیگری برای استخراج داده ها نیاز داشته باشید، برای اینگونه موارد می توانید از **عملگر REGEXP در MySQL** استفاده کنید.

### عملگر REGEXP در MySQL

در ادامه ی این مبحث شما را با طرز کار با عملگر REGEXP در MySQL آشنا می کنیم، اگر شما با زبان هایی مانند PHP یا PERL کار کرده باشید ، احتمالاً با عملگر REGEXP آشنا هستید.

جدول زیر شامل الگوهای است که می توان در عملگر **REGEXP** از آن ها استفاده کرد :

| مکان انطباق الگو                                    | Pattern  |
|-----------------------------------------------------|----------|
| شروع رشته                                           | ^        |
| انتهای رشته                                         | \$       |
| هر کاراکتری                                         | .        |
| تمام کاراکترهایی که در براکت مربع شکل لیست شده اند. | [...]    |
| هر کاراکتری که در براکت مربعی لیست نشده باشد.       | [...^]   |
| یکی از الگوهای p1، p2 یا p3 را در نظر می گیرد.      | p1 p2 p3 |

|                                     |       |
|-------------------------------------|-------|
| صفر یا نمونه های بیشتر از عنصر قبلی | *     |
| یک یا نمونه های بیشتر از عنصر قبلی  | +     |
| n نمونه از عنصر قبلی                | {n}   |
| m تا n نمونه از عنصر قبلی           | {m,n} |

#### مثال ها

حالا بر اساس جدول فوق، شما می توانید انواع مختلفی از پرس و جو های SQL را بر اساس انطباق با الگویی خاص استخراج کنید، توجه داشته باشید که ما یک جدول به نام **person\_tbl** داریم که دارای فیلدی با نام **name** است:

یک Query برای پیدا کردن تمام رکوردهایی که name آن ها با 'st' شروع می شوند :

#### مثال :

```
1. mysql> SELECT name FROM person_tbl WHERE name REGEXP '^st';
```

یک Query برای پیدا کردن تمام رکوردهایی که name آن ها با 'ok' خاتمه می یابد :

#### مثال :

```
1. mysql> SELECT name FROM person_tbl WHERE name REGEXP 'ok$';
```

یک Query برای پیدا کردن تمام رکوردهایی که name آن ها شامل 'mar' می باشد :

#### مثال :

```
1. mysql> SELECT name FROM person_tbl WHERE name REGEXP 'mar';
```

یک Query برای پیدا کردن تمام رکوردهایی که name آن ها با یک حرف صدا دار شروع شده و با 'ok' خاتمه می یابد :

#### مثال :

```
1. mysql> SELECT name FROM person_tbl WHERE name REGEXP '^[aeiou]|ok$';
```

## کلام آخر

قبلا برای استخراج رکوردهایی از جداول بر اساس یک الگوی خاص ، با دستوراتی نظیر %... LIKE آشنا شدیم، اما با استفاده از **عملگر REGEXP در MySQL** می توانید الگوهای بیشتری را برای استخراج داده ها از جداول استفاده کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. transaction در MySQL یک سری عملیات ترتیبی دستکاری پایگاه داده است که به صورت یک واحد یکتا کار می کند، به عبارتی دیگر باید هر یک از فرآیندهای دستکاری داده های دیتابیس به درستی انجام شود تا نتیجه ی کلی و نهایی نیز بدون خطا انجام شود، در غیر این صورت اگر هر یک از این فرآیندهایی که در یک گروه در قالب یک transaction تعریف شده به نتیجه نرسیده و یا با خطا مواجه شود نتیجه ی نهایی نیز با خطا مواجه خواهد بود، در ادامه ی این مبحث شما را بیشتر با **ویژگی transaction در MySQL** آشنا خواهیم کرد.

### آشنایی با transaction در MySQL

در ادامه ی این آموزش با ویژگی transaction در MySQL و انواع آن آشنا خواهید شد.

### خصوصیات یک ویژگی transaction در MySQL

transaction ها شامل ۴ خصوصیت استاندارد می باشند، که در مخفف ACID خلاصه می شود:

- **Atomicity** : این تضمین می کند که تمام عملیات در واحد کار با موفقیت انجام شده است، در غیر این صورت خطا رخ می دهد.
- **Consistency** : این تضمین می کند که وضعیت دیتابیس به درستی طی این عملیات دستکاری تغییر کرده می کند.
- **Isolation** : این transaction ها را قادر می سازد تا به صورت مستقل از یکدیگر عمل کنند.
- **Durability** : این اطمینان حاصل می کند که نتیجه یک transaction مهم در صورت خرابی سیستم ادامه می یابد.

در MySQL وضعیت transaction ها با **BEGIN WORK** شروع شده و با **COMMIT** یا **ROLLBACK** پایان می یابد.

### COMMIT و ROLLBACK

دو کلمه ی کلیدی **Commit** و **Rollback** به طور عمده برای Transaction های MySQL استفاده می شود.

- هنگامی که یک عملیات موفق انجام شود، دستور **COMMIT** باید صادر شود تا تغییرات در همه جداول مرتبط در عملیات انجام شود.
- اگر یک خطا اتفاق بیفتد، یک دستور **ROLLBACK** باید صادر شود تا هر جدولی که در عملیات به حالت قبلی رجوع می شود را نشان دهد.

شما می توانید رفتار یک transaction را با تنظیم متغیر جلسه به نام **AUTO COMMIT** کنترل کنید.

اگر **AUTO COMMIT** با مقدار ۱ (به طور پیش فرض) تنظیم شده باشد، پس هر دستور (SQL در یک معامله یا نه) یک Transaction کامل است و وقتی به پایان رسید، به صورت پیش فرض انجام می شود.

وقتی که **AUTO COMMIT** مقدار ۰ داشته باشد (با تنظیم **SET AUTOCOMMIT = 0**) سری بعدی از دستورات مثل یک Transaction عمل می کند و هیچ فعالیتی تا زمانی که دستور **COMMIT** اجرا نشود، فعال نمی شود.

شما می توانید این Query ها را با استفاده از تابع **(mysql\_query)** در PHP اجرا کنید.

## یک مثال عمومی از Transaction

این مجموعه ی رویدادها به زبان برنامه نویسی مورد استفاده بستگی دارد، شروع transaction با اجرای دستور **BEGIN WORK** در دستورات SQL است.

اجرای یک یا بیشتر دستورات SQL نظیر SELECT, INSERT, UPDATE یا DELETE.

بررسی کنید که آیا هیچ خطایی وجود ندارد و همه چیز مطابق نیاز شماست، اگر خطایی وجود داشته باشد، یک دستور ROLLBACK صادر می کند، در غیر این صورت دستور COMMIT را اجرا می کند.

## انواع Transaction-Safe در MySQL

شما نمی توانید مستقیماً از transaction استفاده کنید، اما برای exception ها می توانید اینکار را انجام دهید.

با این حال، آنها امن و تضمین شده نیستند. اگر شما قصد استفاده از transaction در MySQL را دارید، باید جدول های خود را به روش خاصی ایجاد کنید.

انواع مختلفی از جداول وجود دارد که از transaction ها پشتیبانی می کنند اما محبوب ترین InnoDB است، پشتیبانی از جداول InnoDB نیاز به یک پارامتر کامپایل خاص هنگام کامپایل MySQL از منبع دارد.

شما باید MySQL سازگار با جداول InnoDB را داشته باشید، در غیر این صورت باید آن را دانلود نمایید، اگر MySQL از InnoDB پشتیبانی می کند، به سادگی کافیست تعریف **TYPE = InnoDB** را به دستور ایجاد جدول اضافه کنید.

برای مثال کد زیر یک جدول InnoDB را ایجاد می کند که **count\_tbl** نام دارد :

مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3.
4. mysql> use TUTORIALS;
5. Database changed
6.
7. mysql> create table tcount_tbl
8.   -> (
9.     -> tutorial_author varchar(40) NOT NULL,
10.    -> tutorial_count INT
11.    -> ) TYPE = InnoDB;
12. Query OK, 0 rows affected (0.05 sec)
```

همچنین می توانید از جداول **BDB GEMINI** نیز استفاده کنید که البته این به نسخه ی نصب MySQL شما بستگی دارد.

## کلام آخر

اگر می خواهید چنیدن QUERY مختلف را در MySQL در قالب یک فرآیند واحد انجام دهید، استفاده از **ویژگی transaction در MySQL** به شما توصیه می شود، البته قبل از هر کاری از اینکه نسخه ی MySQL نصب شده ی شما از انواع transaction پشتیبانی می کند اطمینان حاصل کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. گاهی ممکن است به دلایلی لازم شود که نام یک جدول و یا نام یک فیلد یا فیلدهایی از جدول را تغییر دهید. همچنین شاید خواسته باشید یکی از ستون های موجود در جدول را حذف کرده و یا اینکه ستونی را به آن اضافه نمائید، برای تغییرات نام جداول و فیلدها و یا افزودن ستونی به جدول و یا حذف یک ستون موجود ، می توان از دستور ALTER در MySQL استفاده نمود ، در ادامه ی این آموزش شما را با چگونگی استفاده از **دستور ALTER در MySQL** آشنا خواهیم کرد.

### دستور ALTER در MySQL

در ادامه ی این مباحث با چگونگی و چند نمونه از موارد استفاده از دستور ALTER در MySQL آشنا خواهید شد، دستور ALTER در MySQL برای هر گونه تغییر در نام جداول یا ستون ها و یا افزودن و حذف ستون ها کاربرد دارد.

ابتدا برای نمونه جدولی با نام **testalter\_tbl** را ایجاد می کنیم :

#### مثال :

```
1. root@host# mysql -u root -p password;
2. Enter password:*****
3.
4. mysql> use TUTORIALS;
5. Database changed
6.
7. mysql> create table testalter_tbl
8.   -> (
9.     -> i INT,
10.    -> c CHAR(1)
11.    -> );
12. Query OK, 0 rows affected (0.05 sec)
13. mysql> SHOW COLUMNS FROM testalter_tbl;
14. +-----+-----+-----+-----+-----+-----+
15. | Field | Type  | Null | Key | Default | Extra |
16. +-----+-----+-----+-----+-----+-----+
17. | i     | int(11) | YES  |     | NULL    |       |
18. | c     | char(1) | YES  |     | NULL    |       |
19. +-----+-----+-----+-----+-----+-----+
20. 2 rows in set (0.00 sec)
```

### حذف ، افزودن و یا تغییر موقعیت یک ستون در جدول

به منظور اینکه ستون ا را از جدول MySQL حذف کنید باید از دستور **DROP** پس از **ALTER** بصورت زیر استفاده کنید:

#### مثال :

```
1. mysql> ALTER TABLE testalter_tbl DROP i;
```



**نکته:** اگر جدول تنها شامل یک ستون باشد، دستور **DROP** عمل نخواهد کرد.

برای افزودن ستون می توان از دستور **ADD** در تعریف ستون بعد از **ALTER** استفاده کرد، نمونه کد زیر ستون **i** را به جدول **testalter\_tbl** اضافه می کند :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl ADD i INT;
```

پس از افزودن مجدد ستون **i** به جدول **testalter\_tbl** ساختار آن مانند قبل نخواهد بود چراکه در ابتدا ستون **i** اولین ستون این جدول بود اما با حذف و افزودن مجدد آن به جدول، ستونی که به تازگی اضافه شده به انتهای جدول اضافه می شود.

مثال :

```
1. mysql> SHOW COLUMNS FROM testalter_tbl;
2. +-----+-----+-----+-----+-----+
3. | Field | Type  | Null | Key | Default | Extra |
4. +-----+-----+-----+-----+-----+
5. | c     | char(1) | YES  |     | NULL    |       |
6. | i     | int(11) | YES  |     | NULL    |       |
7. +-----+-----+-----+-----+-----+
8. 2 rows in set (0.00 sec)
```

حالا اگر می خواهید ستونی که اضافه می کنید به ابتدای جدول اضافه شود در انتهای **ALTER** از کلمه کلیدی **FIRST** استفاده کنید.

اگر می خواهید ستون جدید بعد از یک ستون مشخص در جدول قرار گیرد نام آن ستون را پس از **AFTER** و نام ستون جدید را قبل از آن قرار دهید.

در مثال زیر ستون **i** یکبار به عنوان اولین ستون به جدول اضافه می شود و بار دیگر پس از ستون **c** به جدول اضافه می شود:

مثال :

```
1. ALTER TABLE testalter_tbl DROP i;
2. ALTER TABLE testalter_tbl ADD i INT FIRST;
3. ALTER TABLE testalter_tbl DROP i;
4. ALTER TABLE testalter_tbl ADD i INT AFTER c;
```

**نکته:** کلمات کلیدی **FIRST** و **AFTER** فقط با دستور **ADD** کار می کنند.

این به این معنی است که اگر می خواهید موقعیت یک ستون را در جدول تغییر دهید ابتدا باید آن را حذف کرده و سپس موقعیت جدید آن را با **ADD** تعیین کنید.

## تغییر دادن یک تعریف ستون یا نام

برای تغییر دادن تعریف ستون از کلمات کلیدی **MODIFY** یا **CHANGE** در دستور **ALTER** استفاده کنید، برای مثال، برای تغییر دادن ستون **c** از **CHAR(1)** به **CHAR(10)** می توانید از دستور زیر استفاده کنید :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl MODIFY c CHAR(10);
```

با **CHANGE**، سینتکس یک بیت متفاوت است. پس از کلمه کلیدی **CHANGE** نام ستون را برای تغییر قرار دهید، سپس تعریف جدیدی را تعریف می کنید که شامل نام جدید است، مثال زیر را در نظر بگیرید :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl CHANGE i j BIGINT;
```

برای تغییر **j** از **BIGINT** به **INT** بدون تغییر نام ستون، کد باید به صورت زیر باشد:

مثال :

```
1. mysql> ALTER TABLE testalter_tbl CHANGE j j INT;
```

هنگام تغییر یا تغییر یک ستون، می توانید مشخص کنید که آیا ستون می تواند مقادیر **NULL** یا مقادیر پیش فرض آن را داشته باشد یا خیر، در واقع، اگر شما این کار را نکنید، MySQL به طور خودکار مقادیر برای این ویژگی ها را اختصاص می دهد، در نمونه کد زیر ستونی که **NOT NULL** باشد، مقدار پیش فرض ۱۰۰ را داراست :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl  
2.    -> MODIFY j BIGINT NOT NULL DEFAULT 100;
```

اگر از کد فوق استفاده نکنید ، MySQL تمام ستون ها را با مقادیر **NULL** پر خواهد کرد.

## تغییر دادن مقادیر پیش فرض یک ستون

شما می توانید یک مقدار پیش فرض را برای هر ستونی با استفاده از **ALTER** تعریف کنید، مثال زیر را در نظر داشته باشید :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl ALTER i SET DEFAULT 1000;  
2. mysql> SHOW COLUMNS FROM testalter_tbl;  
3. +-----+-----+-----+-----+-----+  
4. | Field | Type  | Null | Key | Default | Extra |  
5. +-----+-----+-----+-----+-----+  
6. | c     | char(1) | YES  |     | NULL    |       |  
7. | i     | int(11) | YES  |     | 1000    |       |  
8. +-----+-----+-----+-----+-----+  
9. 2 rows in set (0.00 sec)
```

شما می توانید پیش فرض ها را از ستون ها با استفاده از DROP به همراه ALTER حذف کنید.

مثال :

```
1. mysql> ALTER TABLE testalter_tbl ALTER i DROP DEFAULT;
2. mysql> SHOW COLUMNS FROM testalter_tbl;
3. +-----+-----+-----+-----+-----+
4. | Field | Type  | Null | Key | Default | Extra |
5. +-----+-----+-----+-----+-----+
6. | c     | char(1) | YES  |     | NULL    |       |
7. | i     | int(11) | YES  |     | NULL    |       |
8. +-----+-----+-----+-----+-----+
9. 2 rows in set (0.00 sec)
```

## تغییر نوع یک جدول

شما می توانید از یک نوع جدول استفاده کنید، به این صورت که از **TYPE** در طول استفاده از ALTER استفاده کنید، در مثال زیر جدول **testalter\_tbl** را به نوع **MYISAM** تغییر می دهیم.

برای پیدا کردن **TYPE** جاری جداول می توان از **SHOW TABLE STATUS** استفاده کرد:

مثال :

```
1. mysql> ALTER TABLE testalter_tbl TYPE = MYISAM;
2. mysql> SHOW TABLE STATUS LIKE 'testalter_tbl'\G
3. ***** 1. row *****
4.      Name: testalter_tbl
5.      Type: MyISAM
6.      Row_format: Fixed
7.      Rows: 0
8.      Avg_row_length: 0
9.      Data_length: 0
10. Max_data_length: 25769803775
11.      Index_length: 1024
12.      Data_free: 0
13. Auto_increment: NULL
14.      Create_time: 2007-06-03 08:04:36
15.      Update_time: 2007-06-03 08:04:36
16.      Check_time: NULL
17. Create_options:
18.      Comment:
19. 1 row in set (0.00 sec)
```

## تغییر نام یک جدول

برای تغییر نام جداول از گزینه ی **RENAME** از ساختار **ALTER TABLE** استفاده می کنیم، در مثال زیر نام جدول از **testalter\_tbl** به **alter\_tbl** تغییر می کند :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl RENAME TO alter_tbl;
```

شما می توانید از دستور ALTER برای ایجاد و رها کردن دستور INDEX در فایل MySQL استفاده کنید.

## کلام آخر

پس از تعریف و ایجاد ساختار جداول و ستون ها در دیتابیس های MySQL ، به دلایل مختلفی ممکن است بخواهید نام جداول و فیلدها یا ترتیب قرار گیری ستونها را تغییر داده و یا آن ها را حذف و اضافه کنید، در چنین مواردی می توان از **دستور ALTER در MySQL** استفاده نمود.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. INDEX ها در MySQL یک ساختار داده است که سرعت عملیات در یک جدول را بهبود می بخشد. INDEX ها در MySQL را می توان با استفاده از یک یا چند ستون ایجاد کرد، مبنای دودویی را برای جستجوی سریع و دسترسی به داده ها را فراهم می کند. هنگام ایجاد INDEX ، باید توجه داشت که تمام ستون ها برای ساخت پرس وجوهای SQL و ایجاد یک یا چند INDEX در آن ستون ها مورد استفاده قرار می گیرند. در واقع، INDEX نیز نوعی جدول هستند که فیلد index را نگه می دارند و یک اشاره گر به هر یک از رکوردهای جداول موجود دارد. در ادامه ی این مباحث برای آشنایی با **INDEX ها در MySQL** با ما همراه باشید.

### INDEX ها در MySQL

در ادامه ی این آموزش با انواع INDEX و نحوه ی کار با آن ها آشنا می شوید، در نظر داشته باشید که کاربران INDEX ها را نمی بینند. از INDEX ها در MySQL فقط به منظور سرعت بخشیدن به نمایش داده ها استفاده می شوند.

دستورات INSERT و UPDATE از دستورات SELECT در SQL کند تر انجام می شود، چراکه در INSERT و UPDATE مقدار INDEX ها نیز وارد شده یا بروز می شود.

### index ساده و یکتا

شما می توانید یک index یکتا روی یک جدول ایجاد کنید.

یک index یکتا به این معنی است که دو ردیف نمی توانند یک مقدار شاخص را داشته باشند.

سینتکس ایجاد یک Index روی یک جدول به صورت زیر است :

- `CREATE UNIQUE INDEX index_name`
- `ON table_name ( column1, column2)...;`

شما می توانید از یک یا چند ستون برای ایجاد یک index استفاده کنید.

برای مثال ما می توانیم یک index روی جدول **tutorials\_tbl** با استفاده از **tutorial\_author** ایجاد کنیم :

- `CREATE UNIQUE INDEX AUTHOR_INDEX`
- `ON tutorials_tbl (tutorial_author)`

همچنین می توانید با حذف کلمه ی کلیدی **UNIQUE** از query یک index ساده ایجاد کنید.

index ساده اجازه می دهد تا مقادیر در یک جدول تکرار شوند.

اگر می خواهید مقادیر یک ستون را به ترتیب نزولی نشان دهید، می توانید نام DESC را پس از نام ستون ذخیره شده قرار دهید:

- `CREATE UNIQUE INDEX AUTHOR_INDEX <mysql`
- `ON tutorials_tbl (tutorial_author DESC)`

## استفاده از دستور ALTER به منظور افزودن و یا حذف index

۴ نوع دستور برای افزودن index ها به یک جدول وجود دارد :

- ALTER TABLE tbl\_name ADD PRIMARY KEY (column\_list : )

این دستور یک کلید اصلی را به جدول اضافه می کند، این به این معنی است که index این فیلد باید یکتا باشد.

- ALTER TABLE tbl\_name ADD UNIQUE index\_name (column\_list :)

یک index را ایجاد می کند که ارزش ها در آن باید منحصر به فرد باشند (به جز مقادیر NULL که ممکن است چندین بار ظاهر شوند).

- ALTER TABLE tbl\_name ADD INDEX index\_name (column\_list :)

این یک index عادی را اضافه می کند که هر مقدار ممکن است بیش از یک بار ظاهر شود.

- ALTER TABLE tbl\_name ADD FULLTEXT index\_name (column\_list :)

این یک شاخص ویژه FULLTEXT که برای اهداف جستجوی متن استفاده می شود را ایجاد می کند، کد زیر یک نمونه برنامه برای افزودن index در یک جدول موجود است :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl ADD INDEX (c);
```

شما می توانید هر یک از INDEX های موجود را با استفاده از **DROP** در قالب دستور ALTER حذف کنید، مثال زیر index ایجاد شده در مثال فوق را حذف می کند :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl DROP INDEX (c);
```

## استفاده از ALTER برای افزودن یا حذف کلید اصلی

شما می توانید یک کلید اصلی را به همین روش ایجاد کنید، اما اطمینان حاصل کنید که کلید اصلی روی ستون هایی کار می کند که NULL نیستند.

مثال زیر برای افزودن کلید اصلی به جداول موجود در دیتابیس است، این ابتدا ستون NULL را ایجاد می کند و سپس آن را به عنوان کلید اصلی اضافه می کند:

مثال :

```
1. mysql> ALTER TABLE testalter_tbl MODIFY i INT NOT NULL;  
2. mysql> ALTER TABLE testalter_tbl ADD PRIMARY KEY (i);
```

استفاده از ALTER برای حذف یک کلید اصلی :

مثال :

```
1. mysql> ALTER TABLE testalter_tbl DROP PRIMARY KEY;
```

برای حذف یک index که کلید اصلی نیست ، شما باید نام index را مشخص کنید.

## نمایش دادن اطلاعات index

می توانید با استفاده از دستور **SHOW INDEX** تمام index های موجود در یک جدول را لیست کنید، می توانید با استفاده از کاراکتر \G تمام نتایج را به صورت عمودی مرتب سازی کنید.

مثال :

```
1. mysql> SHOW INDEX FROM table_name\G
2. ....
```

## کلام آخر

دستوراتی مانند INSERT و UPDATE در زمان افزودن و یا حذف داده ها در جداول ، index های آن ها را نیز تغییر می دهد به همین خاطر زمان بیشتری را می گیرد، اما شما می توانید با **INDEX ها در MySQL** مدت زمان این عملیات را کمتر کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت **پی وی لرن** ، و کاربرانی که **دوره آموزش MySQL** را دنبال می کنند. تاکنون با درج جداول عادی در دیتابیس و ویرایش آن ها آشنا شدیم ، علاوه بر آن می توانید از جداول موقت در MySQL نیز برای نگهداری اطلاعات موقت خود استفاده کنید. مهمترین نکته ای که باید در نظر داشته باشید این است که جداول موقت با پایان یافتن جلسه کاری کاربر جاری حذف می شوند. برای آشنایی بیشتر با **جداول موقت در MySQL** و چگونگی استفاده از آن ها در ادامه ی این مبحث با ما همراه باشید.

### جداول موقت در MySQL

در این آموزش شما را با طرز ایجاد جداول موقت در MySQL و حذف آن ها آشنا می کنیم.

### جداول موقت چیست

جداول موقت در MySQL در نسخه ی ۳/۲۳ MySQL اضافه شده است.

اگر شما از نسخه های قدیمی تر از ۳/۲۳ استفاده کنید، نمی توانید از جداول موقت استفاده کنید، اما می توانید از جداول **Heap** استفاده کنید.

همانطور که قبلا ذکر شد، جداول موقت تنها تا زمانی که جلسه در جریان است، ادامه می یابد، وقتی که اسکریپت های PHP را اجرا می کنید، وقتی که اجرای PHP تمام شود، جداول موقت نیز پایان می یابد، اگر به سرور MySQL از طریق سرویس گیرنده MySQL وصل شده اید، یا باید جدول موقت را دستی حذف کنید، یا زمانیکه جلسه کاری تمام شود حذف می شود.

### مثال

برنامه ی زیر یک مثال است که چگونگی استفاده از جداول موقت را نشان می دهد، این تابع را می توان در قالب تابع **mysql\_query()** در PHP استفاده کرد :

### مثال :

```
1. mysql> CREATE TEMPORARY TABLE SalesSummary (  
2.     -> product_name VARCHAR(50) NOT NULL  
3.     -> , total_sales DECIMAL(12,2) NOT NULL DEFAULT 0.00  
4.     -> , avg_unit_price DECIMAL(7,2) NOT NULL DEFAULT 0.00  
5.     -> , total_units_sold INT UNSIGNED NOT NULL DEFAULT 0  
6. );  
7. Query OK, 0 rows affected (0.00 sec)  
8.  
9. mysql> INSERT INTO SalesSummary  
10.    -> (product_name, total_sales, avg_unit_price, total_units_sold)  
11.    -> VALUES  
12.    -> ('cucumber', 100.25, 90, 2);  
13.  
14. mysql> SELECT * FROM SalesSummary;  
15. +-----+-----+-----+-----+  
16. | product_name | total_sales | avg_unit_price | total_units_sold |  
17. +-----+-----+-----+-----+  
18. | cucumber    | 100.25      | 90.00          | 2                |  
19. +-----+-----+-----+-----+  
20. 1 row in set (0.00 sec)
```



زمانیکه یک دستور **SHOW TABLES** را اجرا می کنید. جداول موقت در قالب لیست قرار نمی گیرند، حالا اگر از جلسه ی MySQL خارج شوید و یک دستور **SELECT** را اجرا کنید، جدول موقت را پیدا نخواهید کرد.

## حذف جداول موقت

در حالت پیش فرض ، زمانیکه connection دیتابیس تمام شود، تمام جداول موقت حذف می شوند، اما اگر بخواهید برخی از آن ها را خوتان حذف کنید، می توانید دستور **DROP TABLE** را اجرا کنید.

برنامه ی زیر یک مثال ساده از حذف یک جدول موقت است :

مثال :

```
1. mysql> CREATE TEMPORARY TABLE SalesSummary (  
2.     -> product_name VARCHAR(50) NOT NULL  
3.     -> , total_sales DECIMAL(12,2) NOT NULL DEFAULT 0.00  
4.     -> , avg_unit_price DECIMAL(7,2) NOT NULL DEFAULT 0.00  
5.     -> , total_units_sold INT UNSIGNED NOT NULL DEFAULT 0  
6. );  
7. Query OK, 0 rows affected (0.00 sec)  
8.  
9. mysql> INSERT INTO SalesSummary  
10.    -> (product_name, total_sales, avg_unit_price, total_units_sold)  
11.    -> VALUES  
12.    -> ('cucumber', 100.25, 90, 2);  
13.  
14. mysql> SELECT * FROM SalesSummary;  
15. +-----+-----+-----+-----+  
16. | product_name | total_sales | avg_unit_price | total_units_sold |  
17. +-----+-----+-----+-----+  
18. | cucumber    | 100.25     | 90.00         | 2                |  
19. +-----+-----+-----+-----+  
20. 1 row in set (0.00 sec)  
21. mysql> DROP TABLE SalesSummary;  
22. mysql> SELECT * FROM SalesSummary;  
23. ERROR 1146: Table 'TUTORIALS.SalesSummary' doesn't exist
```

## کلام آخر

اگر داده های موقتی در دیتابیس خود دارید که می خواهید موقتا آن ها را در قالب جدولی ذخیره داشته باشید، نیازی به ذخیره آن ها در جداول موجود در دیتابیس نیست، شما به راحتی می توانید همانند مباحث فوق از **جداول موقت در MySQL** استفاده کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در مواردی ممکن است بخواهید که یک نسخه ی کپی دقیق از یکی از جداول خود در دیتابیس داشته باشید، برای این منظور دستورات **CREATE TABLE ... SELECT** و... شما را به مقصودتان نمی رساند. ما در این بخش یک راه حل ساده را برای ایجاد یک نسخه ی کپی دقیقاً مشابه یک جدول موجود در دیتابیس را به شما ارائه می دهیم. به این نسخه های مشابه جداول Clone گفته می شود، در ادامه ی این بخش ما شما را با چگونگی ایجاد و مدیریت **جداول Clone در MySQL** آشنا می کنیم.

### آشنایی با جداول Clone در MySQL

در این آموزش در قالب یک مثال شما را با سینتکس ایجاد جداول Clone در MySQL آشنا می کنیم، شما می توانید با دنبال کردن مراحل زیر با چگونگی ایجاد جداول Clone در MySQL آشنا شوید:

- با استفاده از دستور **SHOW CREATE TABLE** تمام جداول موجود در دیتابیس را لیست کنید.
- دستور ایجاد نسخه اصلی را برای جدول کپی (Clone) با نام و ترتیبی متفاوت تغییر دهید.
- اگر به محتوای جدول هم نیاز دارید از یک دستور **INSERT INTO ... SELECT** نیز استفاده کنید.

### مثال ایجاد جداول Clone در MySQL

در مثال زیر یک جدول Clone از جدول **tutorials\_tbl** ایجاد می شود:

**مرحله-۱ :** ساختار کامل ایجاد جداول را پیاده سازی کنید :

مثال :

```
1. mysql> SHOW CREATE TABLE tutorials_tbl \G;
2. ***** 1. row *****
3.      Table: tutorials_tbl
4.      Create Table: CREATE TABLE `tutorials_tbl` (
5.        `tutorial_id` int(11) NOT NULL auto_increment,
6.        `tutorial_title` varchar(100) NOT NULL default '',
7.        `tutorial_author` varchar(40) NOT NULL default '',
8.        `submission_date` date default NULL,
9.        PRIMARY KEY (`tutorial_id`),
10.        UNIQUE KEY `AUTHOR_INDEX` (`tutorial_author`)
11. ) TYPE = MyISAM
12. 1 row in set (0.00 sec)
13.
14. ERROR:
15. No query specified
```

**مرحله-۲ :** تغییر دادن نام جداول و ایجاد سایر جداول :

مثال :

```
1. mysql> CREATE TABLE clone_tbl (
2.   -> tutorial_id int(11) NOT NULL auto_increment,
3.   -> tutorial_title varchar(100) NOT NULL default '',
4.   -> tutorial_author varchar(40) NOT NULL default '',
5.   -> submission_date date default NULL,
```

```

6.      -> PRIMARY KEY (tutorial_id),
7.      -> UNIQUE KEY AUTHOR_INDEX (tutorial_author)
8.  -> ) TYPE = MyISAM;
9. Query OK, 0 rows affected (1.80 sec)

```

**مرحله-۳ :** بعد از اجرای مرحله-۲ یک جدول clone در دیتابیس ایجاد می کنیم، اگر می خواهید داده ها را از جدول قدیمی کپی کنید، می توانید آن را با استفاده از INSERT INTO ... SELECT اضافه کنید :

**مثال :**

```

1. mysql> INSERT INTO clone_tbl (tutorial_id,
2.      -> tutorial_title,
3.      -> tutorial_author,
4.      -> submission_date)
5.
6.      -> SELECT tutorial_id,tutorial_title,
7.      -> tutorial_author,submission_date
8.      -> FROM tutorials_tbl;
9. Query OK, 3 rows affected (0.07 sec)
10. Records: 3  Duplicates: 0  Warnings: 0

```

در نهایت، شما یک جدول clone دقیق خواهید داشت که شما می خواهید.

## کلام آخر

گاهی لازم می شود به دلایلی از جمله اهمیت بالای داده های یک جدول، یک نسخه ی کپی و دقیقا مشابه از یکی از جداول موجود در دیتابیس داشته باشید، که شما به راحتی می توانید از **جدول Clone در MySQL** استفاده کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت **پی وی لرن** ، و کاربرانی که **دوره آموزش MySQL** را دنبال می کنند. تمام جداولی که برای ذخیره ی داده های خود در MySQL ایجاد می کنید در یک دیتابیس واحد قرار دارند، و غالباً پیش خواهد آمد که نیاز دارید اطلاعاتی در مورد تمام جداول موجود در دیتابیس ، نتایج Query های دیتابیس و یا سرور MySQL خود بدست آورید، به عنوان مثال ممکن است بخواهید بین چند جدول در دیتابیس خود ارتباط ایجاد کنید و نیاز دارید که تمام جداول را برای ایجاد بهترین ارتباط مشاهده کنید، برای اینگونه مقاصد کلی در یک دیتابیس باید از فرامین و دستورات **اطلاعات دیتابیس در MySQL** استفاده کنید که در ادامه ی این مبحث به آن پرداخته ایم.

### اطلاعات دیتابیس در MySQL

در ادامه ی این آموزش شما را با دلایل و نمونه هایی از دستورات اطلاعات دیتابیس در MySQL آشنا خواهیم کرد.

شما می توانید سه نوع اطلاعات از دیتابیس خود داشته باشید :

- اطلاعاتی در مورد نتایج Query ها : شامل رکوردهایی می شود که با دستورات SELECT, UPDATE یا DELETE بدست آمده است.
- اطلاعاتی در مورد جداول و دیتابیس ها : شامل اطلاعات ساختار جداول و دیتابیس ها می شود.
- اطلاعاتی در مورد سرور : MySQL این شامل وضعیت های سرور دیتابیس، شماره نسخه و... می شود.

در MySQL دریافت اطلاعات دیتابیس آسان است اما در هنگام استفاده از PERL یا PHP باید API های مختلف را برای دریافت اطلاعات فراخوانی کنید.

### بدست آوردن تعداد سطرهای یک query

در ادامه مثال هایی از چگونگی بدست آوردن اطلاعات دیتابیس در MySQL را بررسی کرده ایم.

در اسکریپت های DBI تعداد سطرهای ظاهر شده با دستور **do ( )** یا **execute ( )** بازگردانده می شود، که به چگونگی اجرای query بستگی دارد:

مثال :

```
1. # Method 1
2. # execute $query using do( )
3. my $count = $dbh->do($query);
4. # report 0 rows if an error occurred
5. printf "%d rows were affected\n", (defined ($count) ? $count : 0);
6.
7. # Method 2
8. # execute query using prepare( ) plus execute( )
9. my $sth = $dbh->prepare($query);
10. my $count = $sth->execute( );
11. printf "%d rows were affected\n", (defined ($count) ? $count : 0);
```

## مثال PHP

در PHP از تابع `mysql_affected_rows()` برای بدست آوردن تعداد رکوردهایی که در query تغییر کرده اند استفاده می شود :

مثال :

```
1. $result_id = mysql_query ($query, $conn_id);
2. # report 0 rows if the query failed
3. $count = ($result_id ? mysql_affected_rows ($conn_id) : 0);
4. print (" $count rows were affected\n");
```

## لیست کردن جداول و دیتابیس ها

لیست کردن تمام پایگاه های داده و جداول موجود در یک سرور پایگاه داده بسیار آسان است، البته اگر اجازه ی اینکار را نداشته باشید نتیجه ی `null` را دریافت می کنید.

غیر از روشی که در کد زیر بررسی می شود، می توان از `SHOW TABLES` یا `SHOW DATABASES` برای دریافت لیستی از جداول یا پایگاه های داده در PHP یا PERL استفاده کرد.

## مثال PERL-2

مثال :

```
1. # Get all the tables available in current database.
2. my @tables = $dbh->tables ( );
3.
4. foreach $table (@tables ){
5.     print "Table Name $table\n";
6. }
```

## مثال PHP-2

مثال :

```
1. <?php
2.     $con = mysql_connect("localhost", "userid", "password");
3.
4.     if (!$con) {
5.         die('Could not connect: ' . mysql_error());
6.     }
7.     $db_list = mysql_list_dbs($con);
8.
9.     while ($db = mysql_fetch_object($db_list)) {
10.         echo $db->Database . "<br />";
11.     }
12.     mysql_close($con);
13. ?>
```

## دریافت Metadata ی سرور

چند دستور مهم در MySQL وجود دارد که می توان از آنها در قالب MySQL و یا در اسکریپت PHP برای دریافت اطلاعات مهم مختلف در مورد سرور پایگاه داده اجرا کرد.

| ردیف | دستور و توضیحات                                 |
|------|-------------------------------------------------|
| ۱    | <b>SELECT VERSION( )</b><br>رشته نسخه سرور      |
| ۲    | <b>SELECT DATABASE( )</b><br>نام جاری دیتابیس   |
| ۳    | <b>SELECT USER( )</b><br>username جاری          |
| ۴    | <b>SHOW STATUS</b><br>شاخص های وضعیت های سرور   |
| ۵    | <b>SHOW VARIABLES</b><br>متغیرهای پیکربندی سرور |

## کلام آخر

غالبا شما با چندین جدول مرتبط یا غیر مرتبط و همچنین چندین query برای این جداول در یک دیتابیس سر و کار دارید. بنابراین گاهی نیاز به دریافت اطلاعاتی کلی در مورد جداول و query ها خواهید داشت که با دستورات دریافت **اطلاعات دیتابیس در MySQL** که در مباحث فوق بررسی شد به سادگی به این مقصود خود می رسید.

## جلسه ۲۸ : توالی ها (sequence) در MySQL

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. توالی ها یا **sequence در MySQL**، مجموعه ای از اعداد صحیح ۱، ۲، ۳، ... هستند که به منظور تقاضاهای خاصی تولید می شوند. توالی ها اغلب در پایگاه های داده استفاده می شوند، زیرا بسیاری از برنامه ها نیاز به هر ردیف در یک جدول برای داشتن یک مقدار منحصر به فرد دارند و **توالی ها در MySQL** یک راه آسان برای تولید آنها می باشند. برای آشنایی با چگونگی کار با توالی ها در MySQL ، در ادامه ی این مباحث با ما همراه باشید.

### کار با توالی ها در MySQL

در ادامه ی این آموزش شما را با چگونگی کار با sequence در MySQL آشنا خواهیم کرد.

### استفاده از ستون AUTO\_INCREMENT

ساده ترین راه استفاده از توالی ها در MySQL ، تعریف یک ستون از نوع **AUTO\_INCREMENT** می باشد.

در مثال زیر جدولی با تعدادی ستون ایجاد شده و سپس تعداد رکورد به آن اضافه می شود، در این جدول نیازی به وارد کردن مقدار فیلد ID نیست، چرا که مقدار آن به صورت خودکار در هر رکورد افزایش می یابد:

#### مثال :

```
1. mysql> CREATE TABLE insect
2.   -> (
3.     -> id INT UNSIGNED NOT NULL AUTO_INCREMENT,
4.     -> PRIMARY KEY (id),
5.     -> name VARCHAR(30) NOT NULL, # type of insect
6.     -> date DATE NOT NULL, # date collected
7.     -> origin VARCHAR(30) NOT NULL # where collected
8.   );
9. Query OK, 0 rows affected (0.02 sec)
10. mysql> INSERT INTO insect (id,name,date,origin) VALUES
11.   -> (NULL,'housefly','2001-09-10','kitchen'),
12.   -> (NULL,'millipede','2001-09-10','driveway'),
13.   -> (NULL,'grasshopper','2001-09-10','front yard');
14. Query OK, 3 rows affected (0.02 sec)
15. Records: 3 Duplicates: 0 Warnings: 0
16. mysql> SELECT * FROM insect ORDER BY id;
17. +-----+-----+-----+-----+
18. | id | name      | date      | origin |
19. +-----+-----+-----+-----+
20. | 1 | housefly  | 2001-09-10 | kitchen |
21. | 2 | millipede | 2001-09-10 | driveway |
22. | 3 | grasshopper | 2001-09-10 | front yard |
23. +-----+-----+-----+-----+
24. 3 rows in set (0.00 sec)
```

### گرفتن مقادیر AUTO\_INCREMENT

( **LAST\_INSERT\_ID** ) یک تابع SQL است، بنابراین شما می توانید از آن در هر client که شامل دستورات SQL است، استفاده کنید.

تابع فوق به منظور دریافت آخرین مقدار فیلد **AUTO\_INCREMENT** کاربرد دارد.

## مثال PERL-1

از صفت **mysql\_insertid** برای گرفتن مقدار **AUTO\_INCREMENT** که با یک query تولید شده استفاده می کنیم.

این صفت در دسته دیتابیس و یا دسته ای از دستورات قابل دسترسی است، مثال زیر آن را از طریق دسته دیتابیس ارجاع می دهد:

### مثال :

```
1. $dbh->do ("INSERT INTO insect (name,date,origin)
2. VALUES('moth','2001-09-14','windowsill')");
3. my $seq = $dbh->{mysql_insertid};
```

## مثال PHP-1

بعد از صدور یک پرس و جو که یک مقدار **AUTO\_INCREMENT** تولید می کند، فراخوانی دستور **mysql\_insert\_id()** مقدار آن را بازپایی می کند:

### مثال :

```
1. mysql_query ("INSERT INTO insect (name,date,origin)
2. VALUES('moth','2001-09-14','windowsill')", $conn_id);
3. $seq = mysql_insert_id ($conn_id);
```

## تغییر شماره یک توالی موجود

فرض کنید موردی پیش آمده که شما تمام رکوردهای یک جدول را حذف کرده اید و می خواهید مجدداً توالی تمام رکوردها را دنبال کنید، این کار با یک ترفند ساده انجام می شود اما در صورت ترکیب جدول شما با یک جدول دیگر کمی مشکل تر است.

اگر تشخیص داده اید که تعویض مجدد یک ستون **AUTO\_INCREMENT** اجتناب ناپذیر است، یک راه حل این است که این ستون را از جدول حذف کرده و مجدداً آن را اضافه کنید.

مثال زیر نشان می دهد که چگونه مقادیر **id** در جدول را با استفاده از این تکنیک مجدداً شماره گذاری کنید:

### مثال :

```
1. mysql> ALTER TABLE insect DROP id;
2. mysql> ALTER TABLE insect
3. -> ADD id INT UNSIGNED NOT NULL AUTO_INCREMENT FIRST,
4. -> ADD PRIMARY KEY (id);
```

## شروع یک توالی در قالب یک مقدار خاص

در حالت پیش فرض MySQL یک توالی را از ۱ شروع می کند، شما می توانید هر شماره دیگری نیز در زمان ایجاد جدول، مشخص کنید.



برنامه ی زیر یک مثال است که نشان می دهد چگونه MySQL توالی را از ۱۰۰ شروع کند :

مثال :

```
1. mysql> CREATE TABLE insect
2.     -> (
3.     -> id INT UNSIGNED NOT NULL AUTO_INCREMENT = 100,
4.     -> PRIMARY KEY (id),
5.     -> name VARCHAR(30) NOT NULL, # type of insect
6.     -> date DATE NOT NULL, # date collected
7.     -> origin VARCHAR(30) NOT NULL # where collected
8. );
```

همچنین می توانید پس از ایجاد جدول ، با استفاده از دستور **ALTER TABLE** مقدار sequence را تنظیم کنید:

مثال :

```
1. mysql> ALTER TABLE t AUTO_INCREMENT = 100;
```

## کلام آخر

با استفاده از **توالی ها در MySQL** شما به راحتی می توانید یک ستون شمارنده ی خودکار به جدول اضافه کنید که با هر رکورد جدیدی که به جدول اضافه می شود یک واحد به فیلد شمارنده به طور خودکار اضافه گردد، در واقع کاربرد عمده ی **sequence در MySQL** ایجاد فیلد شمارنده ی خودکار برای جداول است.

## جلسه ۲۹ : مدیریت موارد تکراری در MySQL

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. به طور کلی، جداول یا مجموعه نتایج بعضی اوقات حاوی داده های تکراری هستند. این در اغلب موارد مجاز است، اما گاهی لازم است که رکوردهای تکراری را متوقف کند. لازم است که رکوردهای تکراری را شناسایی کرده و آنها را از جدول حذف کنید. در این بخش **مدیریت موارد تکراری در MySQL** ، نحوه جلوگیری از وقوع رکوردهای تکراری در یک جدول و نحوه حذف رکوردهای تکراری موجود را توضیح می دهیم.

### جلوگیری از تکراری بودن در یک جدول

می توان یک کلید اصلی یا یک Index یکتا را در یک جدول با فیلدهای مناسب برای متوقف کردن رکوردهای دوگانه استفاده کرد.

جدول زیر شامل index یا کلید اصلی نمی باشد، بنابراین اجازه ی درج داده های تکراری در **first\_name** و **last\_name** داده شده است:

#### مثال :

```
1. CREATE TABLE person_tbl (  
2.     first_name CHAR(20),  
3.     last_name CHAR(20),  
4.     sex CHAR(10)  
5. );
```

برای جلوگیری از رکوردهای تکراری از یک کلید اصلی برای تعریف آن اضافه کنید، وقتی این کار را انجام می دهید، لازم است که ستون های NOT NULL را اعلام کنیم، زیرا PRIMARY KEY به مقادیر NULL اجازه نمی دهد:

#### مثال :

```
1. CREATE TABLE person_tbl (  
2.     first_name CHAR(20) NOT NULL,  
3.     last_name CHAR(20) NOT NULL,  
4.     sex CHAR(10),  
5.     PRIMARY KEY (last_name, first_name)  
6. );
```

وقتی جدولی شامل یک index یکتا می باشد افزودن مقادیر تکراری به آن index موجب بروز خطا می شود. برای جلوگیری از اعلام خطای موارد تکراری در index یکتا از دستور **INSERT IGNORE** به جای دستور **INSERT** استفاده کنید.

اگر رکوردی رکورد دیگری را کپی نمی کند پس MySQL آن را به طور معمولی قرار داده است. اگر رکوردی تکراری باشد کلمه ی کلیدی **IGNORE** به MySQL اعلام می کند که آن مقدار را بدون اعلام خطا نادیده بگیرد.

مثال زیر هیچ خطایی برای درج موارد تکراری در فیلدهای یکتا اعلام نکرده و از درج موارد تکراری جلوگیری می شود:

#### مثال :

```
1. mysql> INSERT IGNORE INTO person_tbl (last_name, first_name)  
2.     -> VALUES( 'Jay', 'Thomas');  
3. Query OK, 1 row affected (0.00 sec)
```

```

4.
5. mysql> INSERT IGNORE INTO person_tbl (last_name, first_name)
6.     -> VALUES( 'Jay', 'Thomas');
7. Query OK, 0 rows affected (0.00 sec)

```

همچنین از کلمه کلیدی **REPLACE** به جای INSERT استفاده کنید، در این حالت اگر مقدار جدید تکراری باشد، جایگزین مقدار قبلی خواهد شد:

### مثال :

```

1. mysql> REPLACE INTO person_tbl (last_name, first_name)
2.     -> VALUES( 'Ajay', 'Kumar');
3. Query OK, 1 row affected (0.00 sec)
4.
5. mysql> REPLACE INTO person_tbl (last_name, first_name)
6.     -> VALUES( 'Ajay', 'Kumar');
7. Query OK, 2 rows affected (0.00 sec)

```

دستور INSERT IGNORE اولین مجموعه ای از رکوردهای تکراری را حفظ کرده و باقی مانده را از بین می برد، دستور REPLACE آخرین مجموعه ی از تکراری ها را حفظ کرده و هر گونه اطلاعات قبلی را پاک می کند.

راه های دیگر ایجاد فیلد یکتا استفاده از **UNIQUE** به جای PRIMARY KEY در یک جدول است:

### مثال :

```

1. CREATE TABLE person_tbl (
2.     first_name CHAR(20) NOT NULL,
3.     last_name CHAR(20) NOT NULL,
4.     sex CHAR(10)
5.     UNIQUE (last_name, first_name)
6. );

```

## شمارش و تشخیص موارد تکراری

مثال زیر یک query است که رکوردهای تکراری ایجاد شده در فیلدهای first\_name و last\_name را شمارش می کند:

### مثال :

```

1. mysql> SELECT COUNT(*) as repetitions, last_name, first_name
2.     -> FROM person_tbl
3.     -> GROUP BY last_name, first_name
4.     -> HAVING repetitions > 1;

```

این query یک لیست از تمام رکوردهای تکراری در در جدول person\_tbl بر می گرداند، عموماً برای شناسایی مجموعه مقادیری که کپی شده اند، مراحل زیر را دنبال کنید:

- مشخص کنید کدام ستون ها حاوی مقادیری هستند که ممکن است تکرار شوند.
- لیست این ستون ها را در لیست انتخاب ستون همراه با (\*) . COUNT
- ستون ها را در بند GROUP BY نیز نمایش دهید.
- یک بند HAVING اضافه کنید که ارزش های منحصر به فرد را از بین می برد.

## حذف تکراری از یک نتیجه پرس و جو

می توانید از دستور **DISTINCT** در یک ساختار SELECT برای پیدا کردن رکوردهای یکتای قابل دسترس در یک جدول استفاده کنید:

مثال :

```
1. mysql> SELECT DISTINCT last_name, first_name
2.     -> FROM person_tbl
3.     -> ORDER BY last_name;
```

یک جایگزین برای دستور DISTINCT این است که یک بند GROUP BY را اضافه کنید که ستون هایی را که انتخاب می کنید نام می گیرد.

این برای از بین بردن تکراری ها و انتخاب ترکیبات یکتا از مقادیر مشخص ستون ها استفاده می شود :

مثال :

```
1. mysql> SELECT last_name, first_name
2.     -> FROM person_tbl
3.     -> GROUP BY (last_name, first_name);
```

## حذف موارد تکراری با جایگزینی

اگر رکوردهای تکراری در یک جدول دارید و می خواهید تمام رکوردهای تکراری از یک جدول حذف شود، روش زیر را دنبال کنید:

مثال :

```
1. mysql> CREATE TABLE tmp SELECT last_name, first_name, sex
2.     -> FROM person_tbl;
3.     -> GROUP BY (last_name, first_name);
4.
5. mysql> DROP TABLE person_tbl;
6. mysql> ALTER TABLE tmp RENAME TO person_tbl;
```

یک روش ساده برای از بین بردن رکوردهای تکراری از یک جدول، یک INDEX یا یک کلید اصلی برای آن جدول است، حتی اگر همچنین جدولی موجود باشد می توانید با افزودن INDEX یکتا یا کلید اصلی موارد تکراری را حذف کنید:

مثال :

```
1. mysql> ALTER IGNORE TABLE person_tbl
2.     -> ADD PRIMARY KEY (last_name, first_name);
```

## کلام آخر

در حالت عادی جداول ایجاد شده در دیتابیس موارد تکراری را می پذیرد، اما شما به راحتی با روش های گفته شده می توانید برای مدیریت موارد تکراری در MySQL و جلوگیری از درج آن در آینده اقدام کنید.

## جلسه ۳۰ : تزریق به دیتابیس (SQL Injection) در MySQL

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بسیاری از برنامه ها و وب سایت ها یک صفحه ی ورودی یا Login برای اعتبار سنجی و ورود اعضاء ایجاد می شود، در اینگونه صفحات اطلاعاتی از کاربر دریافت شده و در سمت سرور با داده هایی در دیتابیس مقایسه می شود تا اعتبار سنجی انجام شود، در اینگونه موارد احتمال حملات SQL Injection یا تزریق به دیتابیس توسط هکرها وجود دارد. که در این صورت تمام داده های ذخیره شده در دیتابیس فاش خواهد شد، احتمال **SQL Injection در MySQL** نیز که یک سیستم مدیریت داده ی SQL می باشد نیز وجود دارد. در ادامه ی این مبحث به چگونگی جلوگیری از حملات **تزریق به دیتابیس در MySQL** پرداخته ایم.

### MySQL در SQL Injection

در این آموزش با کدنویسی استاندارد برای جلوگیری از تزریق به دیتابیس در MySQL آشنا خواهید شد.

### آشنایی با SQL Injection در MySQL

تزریق به دیتابیس در MySQL معمولا زمان دریافت ورودی از کاربران رخ می دهد، برای جلوگیری از SQL Injection در MySQL باید از یک الگوی تطبیق اعتبار سنجی کنید.

در مثال زیر ورودی دریافتی از کاربران به حروف الفبا، کاراکتر زیر خط دار و طول ورودی بین ۸ تا ۲۰ کاراکتر محدود می شود:

#### مثال :

```
1. if (preg_match("/^\w{8,20}$/", $_GET['username'], $matches)) {
2.     $result = mysql_query("SELECT * FROM users WHERE username = '$matches[0]'");
3. } else {
4.     echo "username not accepted";
5. }
```

برای نشان دادن مشکل ، کد زیر را در نظر بگیرید :

#### مثال :

```
1. // supposed input
2. $name = "Qadir"; DELETE FROM users;";
3. mysql_query("SELECT * FROM users WHERE name = '{$name}'");
```

در کد فوق ورودی \$name دریافتی از کاربر می تواند شامل تمام حروف الفبا و فضاهای خالی نیز باشد، در اینجا ما ورودی \$name که ورودی کاربر است را یک دستور DELETE نوشته ایم، که تمام داده های کاربران را حذف می کند!

البته در هنگام استفاده از MySQL تابع `mysql_query()` اجازه نمی دهد که چنین Query هایی اجرا شوند.

با این وجود در سایر دیتابیس هایی که در PHP از جمله SQLite و PostgreSQL چنین حملاتی می تواند انجام شود.

## جلوگیری از SQL Injection در MySQL

شما می توانید تمام کاراکترهای فرار را به صورت هوشمندانه در زبان های برنامه نویسی مانند PERL و PHP اداره کنید.

MySQL در زبان PHP تابع `mysql_real_escape_string()` را برای جلوگیری از حملات **Injection** ارائه می دهد:

مثال :

```
1. if(get_magic_quotes_gpc()) {  
2.     $name = stripslashes($name);  
3. }  
4.  
5. $name = mysql_real_escape_string($name);  
6. mysql_query("SELECT * FROM users WHERE name = '{$name}'");
```

## استفاده از دستور LIKE

با استفاده از دستور LIKE می توانید با یک مکانیزم الگوی خاص ورودی کاربر را به رشته تبدیل کنید تا استفاده از تابع `addslashes()` نیز می توانید دامنه ای محدود از کاراکترها را تعیین کنید:

مثال :

```
1. $sub = addslashes(mysql_real_escape_string("%something_"), "%_");  
2. // $sub == \%something\  
3. mysql_query("SELECT * FROM messages WHERE subject LIKE '{$sub}%')");
```

## کلام آخر

**تزریق به دیتابیس در MySQL** یکی از حملات بسیار جدی است که دیتابیس های زبان SQL را تهدید می کند، و داده ها را در اختیار هکرها قرار می دهد. برای جلوگیری از اینگونه حملات حتما باید از کدنویسی های خاصی در سمت سرور همانند آنچه در مباحث فوق اشاره شد استفاده کنید.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. گاهی اوقات ممکن است بخواهید پروژه یا برنامه خود را روی سیستم های دیگری اجرا کرده و یا توسعه دهید، در این گونه مواقع علاوه بر کدهای برنامه باید داده های دیتابیس خود را نیز در یک خروجی دریافت و در سیستم مورد نظر پیاده کنید، هر کدام از دیتابیس های معروف از جمله MySQL یک گزینه برای Export (خروجی) گرفتن از داده های ذخیره شده در خود دارند، پس شما باید با چگونگی **Export کردن دیتابیس در MySQL** آشنایی داشته باشید.

### Export کردن دیتابیس در MySQL با SELECT...INTO OUTFILE

ساده ترین روش برای Export داده های جدول به یک فایل متنی، استفاده از دستور SELECT ... INTO OUTFILE است.

این دستور نتیجه Query را به طور مستقیم به یک فایل در میزبان سرور صادر می شود.

ساختار اینکار ترکیبی از یک دستور SELECT معمولی با INNO OUTFILE نام فایل در پایان است، فرمت خروجی پیش فرض همانند آن برای دستور LOAD DATA است.

بنابراین دستور زیر جدول **tutorials\_tbl** را در فایل **tmp/tutorials.txt** به عنوان tab-delimited کپی می کند:

#### مثال :

```
1. mysql> SELECT * FROM tutorials_tbl
2.      -> INTO OUTFILE '/tmp/tutorials.txt';
```

شما می توانید فرمت خروجی را با استفاده از گزینه های مختلف تغییر دهید تا نشان دهید که چگونه ستون ها و رکوردها را نقل و تعریف کنید.

برای export کردن جدول tutorial\_tbl در یک فرمت CSV از دستور زیر استفاده کنید:

#### مثال :

```
1. mysql> SELECT * FROM passwd INTO OUTFILE '/tmp/tutorials.txt'
2.      -> FIELDS TERMINATED BY ',' ENCLOSED BY '"'
3.      -> LINES TERMINATED BY '\r\n';
```

دستور **SELECT ... INTO OUTFILE** خصوصیات زیر را داراست :

فایل خروجی به طور مستقیم توسط سرور MySQL ایجاد می شود، بنابراین نام فایل باید محل آن را در سرور میزبان نشان دهد.

شما بایستی امتیاز فایل MySQL را برای اجرای عبارت SELECT ... INTO داشته باشید.

یک فایل خروجی دیگری نباید از قبل ایجاد شده و وجود داشته باشد.

شما باید یک حساب ورود به سیستم در سرور میزبان و یا راهی برای بازیابی فایل از آن میزبان داشته باشید.

تحت یونیکس، این پرونده قابل خواندن است و متعلق به سرور MySQL است و این بدان معنی است که اگر چه شما می توانید فایل را بخوانید، ممکن است نتوانید آن را پاک کنید.

## Export کردن جداول در قالب داده ی Raw

برنامه ی **mysqldump** برای کپی گرفتن و یا پشتیبانی جداول و دیتابیس هاست، این می تواند از داده های جداول را در قالب یک فایل **Raw** کپی بگیرید، برای اینکار باید گزینه ی **tab** که یک دایرکتوری را نشان می دهد مشخص کنید.

برای مثال، برای کپی کردن جدول **tutorials\_tbl** از دیتابیس **TUTORIALS** در یک فایل در دایرکتوری **/tmp** از دستور زیر استفاده کنید:

مثال :

```
1. $ mysqldump -u root -p --no-create-info \  
2.    --tab=/tmp tutorials tutorials_tbl \  
3. password *****
```

## export کردن محتوای جداول یا تعریف در قالب SQL

برای export کردن یک جدول در یک فرمت SQL در یک فایل، از دستور زیر استفاده کنید :

مثال :

```
1. $ mysqldump -u root -p TUTORIALS tutorials_tbl > dump.txt \  
2. password *****
```

کد فوق شامل محتوای زیر را ایجاد می کند :

مثال :

```
1. -- MySQL dump 8.23  
2. --  
3. -- Host: localhost    Database: TUTORIALS  
4. -----  
5. -- Server version      3.23.58  
6.  
7. --  
8. -- Table structure for table `tutorials_tbl`  
9. --  
10.  
11. CREATE TABLE tutorials_tbl (  
12.   tutorial_id int(11) NOT NULL auto_increment,  
13.   tutorial_title varchar(100) NOT NULL default '',  
14.   tutorial_author varchar(40) NOT NULL default '',  
15.   submission_date date default NULL,  
16.   PRIMARY KEY (tutorial_id),  
17.   UNIQUE KEY AUTHOR_INDEX (tutorial_author)  
18. ) TYPE = MyISAM;  
19.  
20. --  
21. -- Dumping data for table `tutorials_tbl`  
22. --  
23.  
24. INSERT INTO tutorials_tbl  
25.   VALUES (1,'Learn PHP','John Poul','2007-05-24');
```



```

26. INSERT INTO tutorials_tbl
27.     VALUES (2,'Learn MySQL','Abdul S','2007-05-24');
28. INSERT INTO tutorials_tbl
29.     VALUES (3,'JAVA Tutorial','Sanjay','2007-05-06');

```

برای کپی گرفتن از چندین جدول همه ی آن ها را با نام آرگومان دیتابیس نام گذاری کنید.

مثال :

```

1. $ mysqldump -u root -p TUTORIALS > database_dump.txt
2. password *****

```

برای گرفتن back up از تمام دیتابیس های موجود در host خود، از کد زیر استفاده کنید :

مثال :

```

1. $ mysqldump -u root -p --all-databases > database_dump.txt
2. password *****

```

گزینه ی `--all-databases` در نسخه ی MySQL 3.23.12 در دسترس است، این متد می تواند برای اجرای استراتژی پشتیبان پایگاه داده استفاده شود.

## کپی جداول یا دیتابیس ها به host های دیگر

اگر می خواهید جداول یا دیتابیس ها را از یک MySQL server به جاهای دیگر کپی کنید، از `mysqldump` با نام جدول و دیتابیس استفاده کنید.

دستور زیر را در یک host منبع اجرا کنید. اینکار تمام دیتابیس را در یک فایل `dump.txt` کپی می کند:

مثال :

```

1. $ mysqldump -u root -p database_name table_name > dump.txt
2. password *****

```

شما می توانید دیتابیس را بدون استفاده از یک نام جدول کپی کنید، در حال حاضر فایل `ftp dump.txt` در میزبان دیگری قرار دارد و از دستور زیر استفاده کنید.

قبل از اجرای این دستور، اطمینان حاصل کنید که نام سرور `database_name` را روی سرور مقصد ایجاد کرده اید.

مثال :

```

1. $ mysql -u root -p database_name < dump.txt
2. password *****

```

راه دیگری برای انجام این کار بدون استفاده از فایل میانجی این است که خروجی mysqldump را مستقیماً بر روی شبکه به سرور MySQL خروجی ارسال کنید.

اگر می‌توانید به هر دو سرور از میزبان که در آن پایگاه داده منبع است، متصل شوید، از دستور زیر استفاده کنید:

مثال :

```
1. $ mysqldump -u root -p database_name \  
2. | mysql -h other-host.com database_name
```

در mysqldump، نیمی از فرمان به سرور محلی متصل می‌شود و خروجی را به pipe می‌فرستد، نیمه باقی مانده از فرمان نیز به سرور MySQL remote در سرور دیگر متصل می‌شود.

این pipe برای ورودی را خوانده و هر دستور را به سرور دیگر – host.com می‌فرستد.

## کلام آخر

اگر گاهی لازم می‌شود که پروژه‌های خود را که شامل دیتابیس MySQL متصل به پروژه می‌شود، را در سایر سیستم‌ها تست کرده و توسعه دهید، باید با چگونگی **Export کردن دیتابیس در MySQL** آشنایی داشته باشید، که در مباحث فوق به صورت خلاصه به آن اشاره شد.

### مقدمه

با عرض سلام و وقت بخیر خدمت کاربران سایت [پی وی لرن](#) ، و کاربرانی که [دوره آموزش MySQL](#) را دنبال می کنند. در بخش قبلی به انواع روشهای Export کردن دیتابیس در MySQL پرداختیم، و چند نمونه کد نیز در این مورد بررسی کردیم. پس از انجام فرآیند Export دیتابیس و جداول شما برای وارد کردن در دیتابیس مقصد که در سیستم دیگری قرار دارد آماده می شود، در آن طرف نیز با طی کردن مراحل و کدهای ساده می توانید دیتابیس و داده های آن را که در قالب یک فایل کپی کرده اید را Import (وارد) کنید. در ادامه ی این مباحث با چگونگی **Import کردن دیتابیس در MySQL** با ما همراه باشید.

### Import کردن دیتابیس در MySQL

در ادامه ی این مباحث به روشها و کدهای Import کردن دیتابیس در MySQL پرداخته ایم.

### Import کردن دیتابیس در MySQL با استفاده از LOAD DATA

MySQL دستور LOAD DATA را که به عنوان یک loader داده ی بزرگ عمل می کند را ارائه می دهد، یک مثال ساده که یک فایل **dump.txt** را از دایرکتوری جاری شما خوانده و در جدول **mytbl** در دیتابیس جاری شما load می کند:

مثال :

```
1. mysql> LOAD DATA LOCAL INFILE 'dump.txt' INTO TABLE mytbl;
```

اگر کلمه ی کلیدی **LOCAL** ارائه نشده باشد، MySQL داده ها را از یک آدرس محلی سیستم دریافت می کند، در حالت پیش فرض **LOAD DATA** به نظر می رسد که فایل داده شامل خطوطی است که با **linefeed** پایان یافته است.

برای مشخص کردن فرمت یک فایل ، از **FIELDS** برای توصیف کاراکترهای یک فیلد در یک خط استفاده کنید، و همچنین از **LINES** برای مشخص کردن توالی **line-ending** استفاده کنید.

دستور **LOAD DATA** مشخص می کند که فایل **data** شامل مقادیر جدا شده توسط کولون ها و خطوط متوقف شده با کاراکتر خط جدید است:

مثال :

```
1. mysql> LOAD DATA LOCAL INFILE 'dump.txt' INTO TABLE mytbl
2.      -> FIELDS TERMINATED BY ':'
3.      -> LINES TERMINATED BY '\r\n';
```

دستور **LOAD DATA** فرض می کند که ستون ها در فایل **data** همانند ستون های جدول مشابه هستند، شما می توانید یک لیست را مشخص کنید تا مشخص شود کدامیک از ستون های **datafile** باید بارگذاری شود.

فرض کنید که جدول شما دارای ستون **a**، **b** و **c** اما ستونهای پیوندی در فایل **data** مربوط به ستون های **b**، **c** و **a** است.

شما می توانید فایل فوق را با استفاده از کد زیر load کنید :

```
1. mysql> LOAD DATA LOCAL INFILE 'dump.txt'
2. -> INTO TABLE mytbl (b, c, a);
```

## Import کردن دیتابیس در MySQL با استفاده از mysqlimport

خروجی زیر نیز شامل یک برنامه به نام mysqlimport است، این برنامه به عنوان یک پوشه در DATA LOAD عمل می کند.

بنابراین شما می توانید فایل های ورودی را مستقیماً از خط فرمان بارگذاری کنید، برای load کردن داده ها از **dump.txt** در **mytbl** از دستور زیر در یونیکس استفاده کنید:

```
1. $ mysqlimport -u root -p --local database_name dump.txt
2. password *****
```

اگر از mysqlimport استفاده می کنید، گزینه های خط فرمان ارائه دهنده های قالب را ارائه می دهند، دستورات **mysqlimport** که با دو عبارت قبلی DATA LOAD مطابقت دارند به نظر می رسد در بلوک کد زیر نشان داده شده است:

```
1. $ mysqlimport -u root -p --local --fields-terminated-by = ":" \
2. --lines-terminated-by = "\r\n" database_name dump.txt
3. password *****
```

دستور که در آن گزینه ها را مشخص می کنید، برای mysqlimport مهم نیست، به جز اینکه همه آنها باید قبل از نام دیتابیس باشند.

دستور **mysqlimport** گزینه **columns-** را برای ترتیب ستون ها مشخص می کند :

```
1. $ mysqlimport -u root -p --local --columns=b,c,a \
2. database_name dump.txt
3. password *****
```

## دستکاری کوتیشن و کاراکترهای خاص

بند **FIELDS** می تواند سایر گزینه های قالب را علاوه بر **TERMINATED BY** مشخص کند.

در حالت پیش فرض **LOAD DATA** فرض می کند که مقادیر بدون کوتیشن و backslash را به عنوان یک کاراکتر escape تفسیر می کند.

برای نشان دادن مقدار کوتیشن به صورت صریح، از دستور **ENCLOSED BY** استفاده کنید، **MySQL** این کاراکتر را از انتهای مقادیر داده در پردازش ورودی حذف می کند.

برای تغییر کاراکتر پیشفرض، از **ESCAPED BY** استفاده کنید.

برای مثال اگر کوتیشن و کاراکتر escape " و \ است مقدار ورودی **"a""b\"c"** به **a"b"c** تفسیر می شود.

برای **mysqlimport**، گزینه های مربوط به خط فرمان و مشخص کردن مقادیر کوتیشن `fields-escaped--` و `fields-enclosed-by` هستند.

## کلام آخر

با ارائه ی مباحث Export کردن دیتابیس در MySQL در بخش قبلی و **Import کردن دیتابیس در MySQL** در این بخش، چگونگی جا به جایی دیتابیس های ایجاد شده در MySQL را به همراه جداول و داده های آن ها را به عنوان آخرین مبحث این دوره ارائه کردیم.