**MySQL**

1. MySQL管理

**连接MySQL**

命令行：mysql -u root -p  
**退出MySQL**

语法:mysql>exit

1. **MySQL创建数据库**语法：mysql>create database Name(数据库的名字)
2. **MySQL删除数据库**语法：drop database Name(数据库的名字)
3. **MySQL选择数据库**

语法：use Name(数据库的名字)

1. **MySQL数据类型**MySQL支持多种类型，大致可以分为三类：数值，日期/时间和字符串(字符)类型。  
   **5.1 数值类型  
   5.1.1 严格数值类型**INTEGER,SMALLINT,DECIMAL和NUMERIC(关键字INT是INTEGER的同义词，关键字DEC是DECIMAL的同义词) **5.1.2 近似数值数据类型**FLOAT,REAL,DOUBLE,PRECISION

**备注：BIT数据类型保存位字段值,并支持MyISAM,MEMORY,InnoDB和BDB表。**





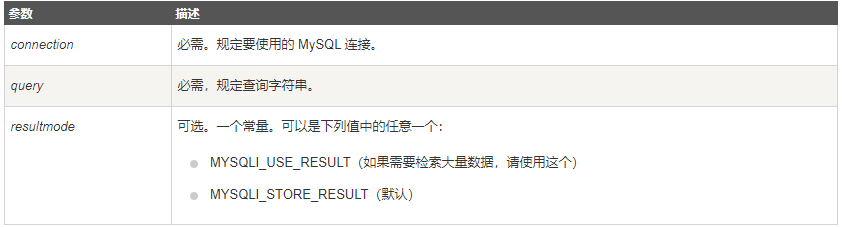


1. **MySQL创建数据表**创建数据表需要一下信息

* 表名
* 表字段名
* 定义每个表字段

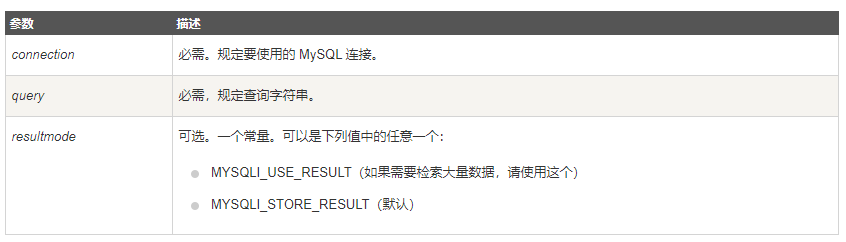
创建MySQL数据表的通用语法：CREATE TABLE table\_name(表的名字)(column\_name column\_type);  


Php语法：mysqli\_query(connection,query,resultmode);



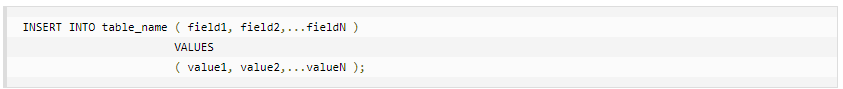
1. **删除MySQL数据表**

语法：DROP TEBLE table\_name;  
php语法：mysqli\_query(connection,query,resultmode);

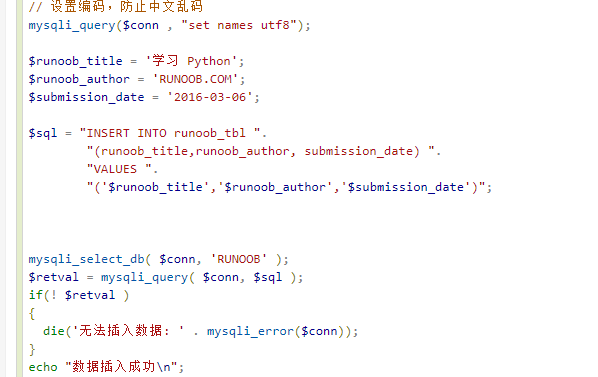


1. **MySQL插入数据**

语法：INSERT INTO SQL语法：



php语法：mysqli\_query(connection,dbname)



1. **MySQL查询数据**

MySQL数据库通过使用SQL SELECT语句来查询数据。  
使用PHP脚本来查询数据：

语法：通过使用PHP函数的mysqli\_query和SELECT命令来获取数据,该函数用于执行SQL命令，然后通过mysqli\_featch\_array()来使用或者输出所有查询的数据。  
**备注：mysqli\_featch\_array()函数从结果集中取得一行作为关联数组，或数字数组或二者兼有的返回根据从结果集取得的行生成的数组,如果没有更多则返回false。Mysqli\_featch\_array()可以通过设置第二个参数为MYSQL\_ASSOC使返回关联数组可以使用字段名称来作为索引。也可以通过设置第二参数为MYSQL\_NUM返回数字数组。**

9.1内存释放：

执行完SELECT语句之后释放游标内存是一个很好的习惯。

PHP语法：mysqli\_free\_result()

1. **MySQL WHERE语句：**

配合SELECT语句有条件的筛选数据。

* 查询语句中你可以使用的一个或者多个表,表之间使用**逗号**隔开，并使用WHERE语句来设定查询条件。
* 可以再WHERE子句中指定任何条件。
* 可以使用AND或OR指定一个或者多个条件。
* WHERE语句也可以运用于SQL的DELETE 或者 UPDATE命令。
* WHERE子句类似于程序语言的if条件,根据MySQL表中的字段来读取指定数据。

  
语法：SELECT \* from tableName WHERE tableKey的筛选条件。

**备注：可以使用BINARY关键字来设定WHERE子句的字符串比较去区分大小写的。**

例子：SELECT \* from runoob\_tbl WHERE BINARY runoob\_id = ‘1’;

1. **MySQL UPDATE更新**

语法：UPDATE table\_name SET field1 = new\_val ,field2 = new\_val[WHERE Clause]  
PHP语法：使用mysqli\_query()来执行SQL语句，在执行SQL UPDATE语句中如果不使用WHERE子句将对**数据表全部数据进行更新，需要谨慎。**

1. **MySQL DELETE语句**

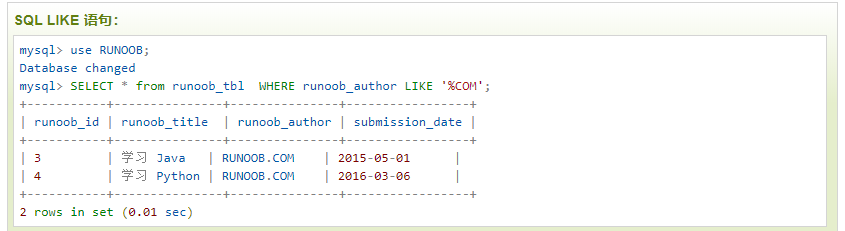
语法：DELETE FROM table\_name [WHERE Clause]

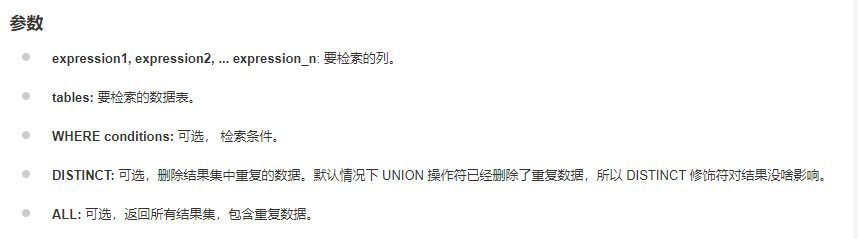
**备注:**如果不适用WHERE子句中的任何条件将**删除所有数据。**

1. **MySQL LIKE语句**

SQL LIKE子句中使用%字符来表示任意字符，如果没有%，LIKE子句与等号=的效果是一样的。

语法：SELECT field1,field2,...fieldn FROM table\_name WHERE field1 LIKE condition1[AND[OR]] field2 = ‘somevalue’



1. **MySQL UNION操作符**用于连续两个以上的SELECT语句的结果组合到一个结果集合中。多个SELECT语句会删除重复的数据。  
   

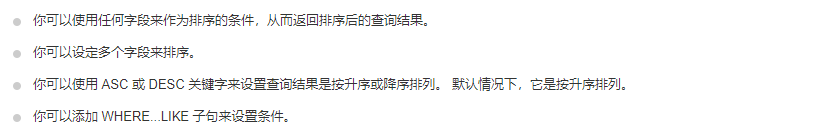
例子：



**备注：ORDER BY 根据指定列队结果集进行排序。**

1. **MySQL排序**

可以使用**ORDER BY**子句来设定你想按哪个字段哪种方式来进行排序,再返回搜索结果。  
语法：SELECT field1,field2,...fieldN FROM table\_name1,table\_name2,table\_nameN ORDER BY field1[ASC[DESC[默认ASC]]],[field2...][ASC[DESC[默认ASC]]]

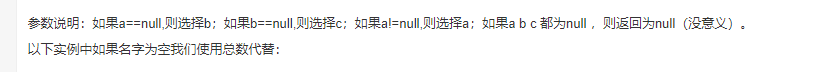


例子：SELECT \* FROM runoob\_tbl ORDER BY submission\_date ASC;  
16. **MySQL GROUP BY语句**GROUP BY语句根据一个或多个列对结果进行分组，在分组上能使用COUNT,SUM,AVG等函数。

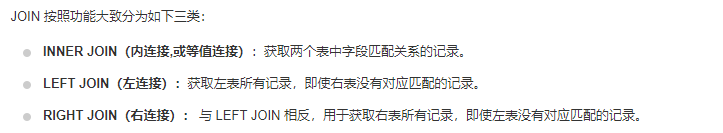
语法：SELECT column\_name,function(column\_name) FROM table\_name WHERE column\_name operator value GROUP BY column\_name

例子:SELECT COUNT(\*) FROM runoob\_tlb GROUP BY runoob\_id

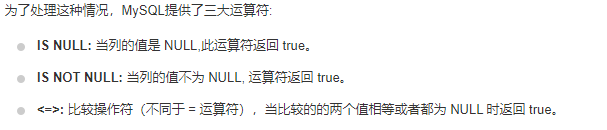
**备注：以上数据按照runoob\_id进行分组并统计出每个分组的数据的数量。**

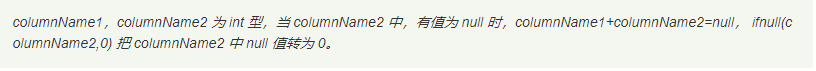
* 1. **使用WITH ROLLUP**WITH ROLLUP可以实现在分组统计数据基础上再进行相同的统计(SUN,AVG,COUNT....)  
     例子：  
     SELECT SUM(singin) as singin\_count FROM employee\_tbl GROUP BY name WITH ROLLUP  
     **备注：可以用select coalesce来去取代NULL的名称 select coalesce(a,b,c)**例子：  
     SELECT conalesce(name,’总数’) SUM(singin) as singin\_count FROM employee\_tbl GROUP BY name WITH ROLLUP

**17.MySQL连接的使用**



1. **MySQL NULL值的处理**



语法:select \* ,columName1+ifnull(columName2,0) from tableName  


1. **MySQL正则表达式**用REGEXP操作符来操作。



例子：SELECT name from person\_tbl WHERE name REGEXP ‘^st’。

匹配name中开头是st的数据

1. **MySQL事务**

MySQL事务主要用于处理操作量大，复杂度高的数据。

* 在MySQL中只有使用了Innodb数据库引擎的数据库或表才支持事务。
* 事务可以用来维护数据库的完整性，用来保证成批的SQL语句要么全部执行，要么都不执行。
* 使用用来管理inset,update,delete语句

**一般来说事务要满足4个条件(ACID):原子性/一致性/隔离性/持久性**



**事务控制语句：**

* BEGIN或START TRANSACTION显示的开启一个事务；
* COMMIT 也可以使用COMMIT WORK并使已对数据库进行的所有修改成永久性的；
* ROLLBACK也可以使用ROLLBACK WORK，回滚，回滚会结束用户的事务，并撤销正在进行的所有未提交的修改；
* SAVEPOINT identifier,SAVEPOINT在事务中创建一个保存点,一个事务中可以有多个SAVEPOINT
* RELEASE SAVEPOINT identifier删除一个事务保存点，当没有指定保存点时会抛出一个错误
* ROLLBACK TO identifier把事务回滚到标记点
* SET TRANSCTION用来设置事务的隔离级别。InnoDB储存引擎

**MySQL事务处理的主要两种方法：**

1. 用BEGIN,ROLLBACK,COMMIT来实现

* **BEGIN**开始一个事务。
* **ROLLBACK**事务回滚。
* **COMMIT**事务确认。

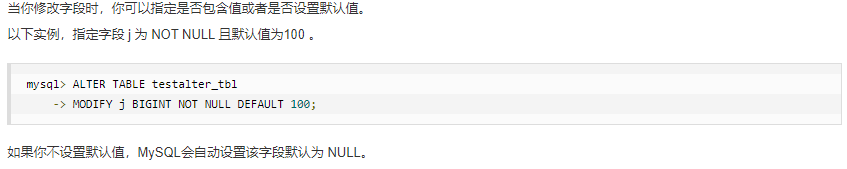
1. 直接用SET来改变MySQL的自动提交模式

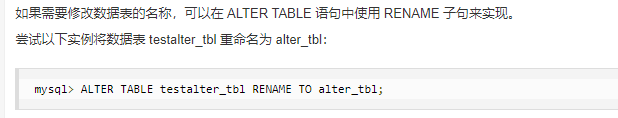
* **SET AUTOCOMMIT = 0禁止自动提交**
* **SET AUTOCOMMIT = 1开启自动提交**

**PHP事务语法**

开始事务语法： mysql\_begin\_transaction

提交/执行事务：mysql\_commit

1. **MySQL ALTER命令**用于修改数据表名跟数据字段。
2. **删除，添加或修改表字段**（通过ALTER命令及DROP子句来删除表的字段）  
   例子:ALTER TABLE table\_name DROP key\_name  
   (使用ADD子句来向数据表中添加列,并可以定义数据类型)  
   例子:ALTER TABLE table\_name ADD key\_name key\_type  
   备注：add的字段会添加在表的末尾,可以通过关键字FIRST(设为第一位)，AFTER key\_name(设置在key\_name之后)也可用于ADD与MODIFY子句。
3. **修改字段类型及名称**（可以在ALTER的命令中使用MODIFY或CHANGE子句来修改字段类型及名称）  
   ALTER TABLE table\_name MODIFY new key\_name new\_type  
   **备注：MODIFY不能用来重命名但是可以用来修改字段类型和约束，使用CHANGE子句语法有很大不同可以用来重命名但是也能用来修改字段类型和约束。**
4. **ALTER TABLE对Null值和默认值的影响**
5. **修改字段默认值**
6. **修改表名**



1. **储存过程**储存过程是一种在数据库种存储复杂程序，以便外部程序调用的一种数据库对象。储存过程完成了特定的SQL语集，经编译创建并保存在数据库中，用户可通过指定的存储过程的名字并给定参数(需要时)来调用执行。其实就是数据库SQL语言层面的代码封装与重用。  
   **优点**

* 存储过程可封装，并隐藏复杂的商业逻辑。
* 存储过程可以回传值,并可以接受参数。
* 存储过程无法使用SELECT指令来运行,因为它是子程序,与查看表,数据表或者用户定义函数不同。
* 存储过程可以用在数据检验,强制执行商业逻辑等。

**缺点**

* 存储过程,往往定制化于特定的数据库上,因为支持的编程语言不同。当切换到其它厂商的数据库系统时，需要写原有的存储过程。
* 存储过程的性能调校于撰写，受限于各种数据库系统。

**存储过程的创建和调用  
MySQL存储过程中的关键语法**

1. **声明语句结束符，可以自定义  
   语法：**DELIMITER $$ 或 DELIMITER //
2. **声明存储过程  
   语法：**CREATE PROCEDURE demo\_in\_parameter(IN p\_in\_int)
3. **存储过程的开始和结束符号  
   语法：**BEGIN.....END
4. **变量赋值  
   语法：**SET @p\_in = 1
5. **变量定义  
   语法：**DECLARE l\_int int unsigned default 4000000
6. **创建MySQL存储过程，存储函数  
   语法：**create procedure 存储过程名(参数)
7. **存储过程体  
   语法:**create function 存储函数名(参数)
8. **MySQL索引  
   优点：**MySQL索引的建立对于MySQL的高效运行很重要，所以可以大大提高MySQL的检索速度。实际上索引也是一张表,该表保存了主键的索引字段，并指向实体表的记录。

**注意：**创建索引时，需要保证该索引是应用在SQL查询语句的条件(一般为WHERE的子句的条件)，

**缺点：**会降低更新表的速度，如对表进行INSERT,UPDATE,DELETE，更新表时，MySQL不仅要保存数据，还要保存一下索引文件。建立索引会占用磁盘空间。

**语法：CREATE INDEX indexName ON mytable(username(length));**

**删除索引的语法：**DROP INDEX [indexName] ON mytable;

1. **MySQL临时表（3.23以上版本）**

MySQL临时表保存一些我们一些临时数据。临时表只在当前连接可见，当关闭连接时,MySQL会自动删除表并释放所以空间。当用PHP来创建MySQL的临时表每当PHP脚本执行完成后改临时表也会被销毁。  
**语法：CREATE TEMPORARY TABLE tebleName**

1. **MySQL复制表**

完整的复制MySQL数据表的步骤：

* 使用**SHOW CREATE TABLE**命令创建数据表，该语句包含了原数据的结构，索引等。
* 复制以下命令显示的SQL语句，修改数据表名，并执行SQL语句，通过以上命令，将完全的复制数据表结构。
* 复制表的内容可以用**INSERT INTO ...SELECT**语句来实现。

1. **MySQL元数据**

**MySQL的三种信息**

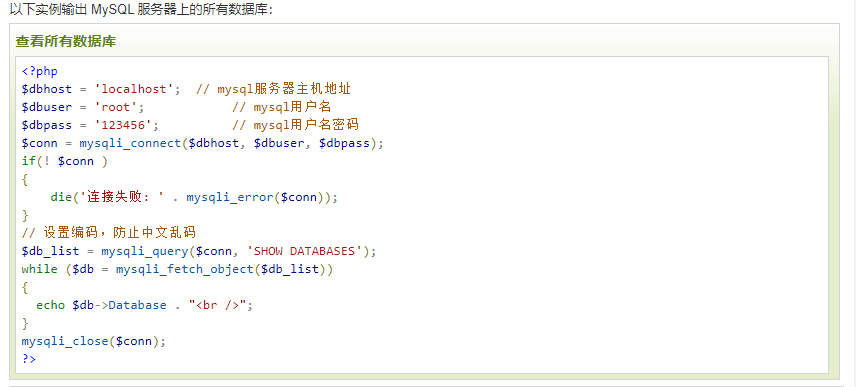
* **查询结果信息：**SELECT,UPDATE,DELETE语句影响的记录数。
* **数据库和数据表的信息：**包含了数据库及数据表的结构信息。
* **MySQL服务器信息：**包含了数据库服务器的当前状态,版本号等。

**PHP获取查询语句影响的记录数**在PHP中可以通过mysqli\_affected\_rows()函数来获取。

**PHP获取数据库和数据列表**

可以通过SHOW TABLES 或 SHOW DATABASES语句来获取数据库和数据列表。

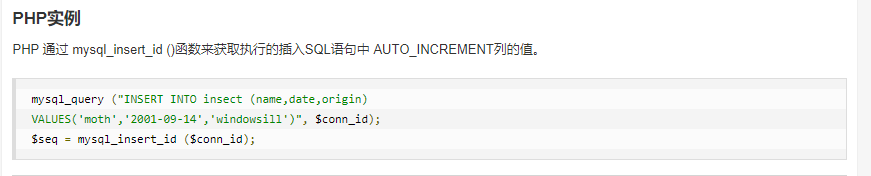
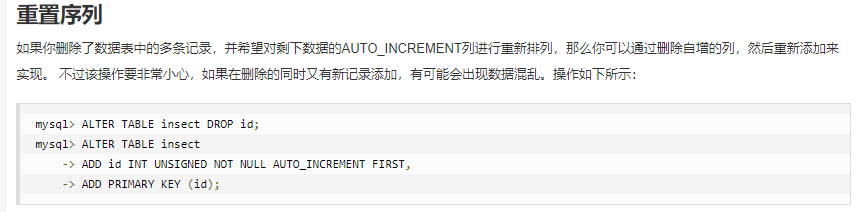
**例子：**





1. **MySQL的序列使用**MySQL中最简单的使用序列的方法就是使用AUTO\_INCREMENT来定义列。  
     
   **获取AUTO\_INCREMENT值**在MySQL的客户端中你可以用SQL的LAST\_INSERT\_ID()函数来获取最后的插入表中的自增列的值。

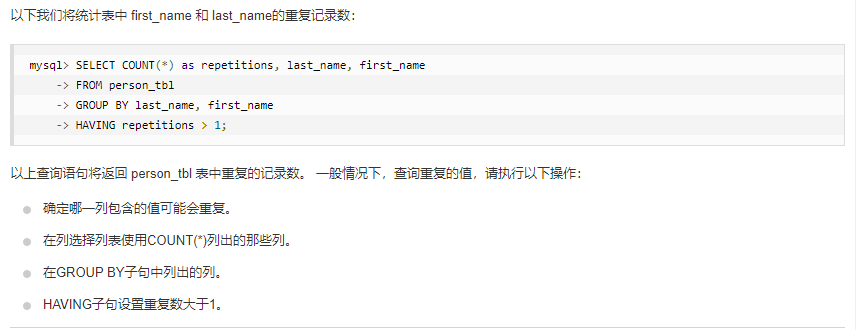
**在PHP中获取AUTO\_INCREMENT的值**

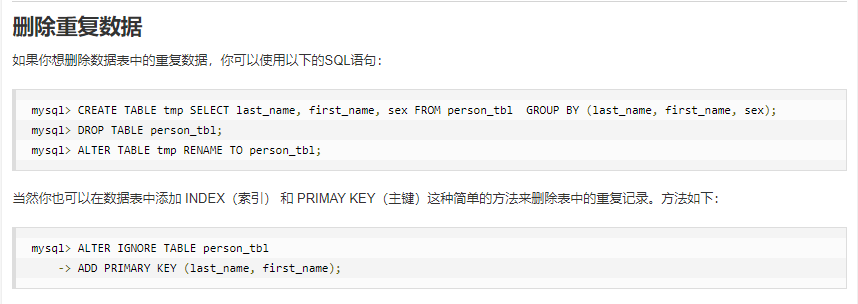
  


1. **MySQL处理重复数据  
   防止表中出现了重复的数据**可以再MySQL数据表中设置指定得字段为PRAMARY KEY(主键)或者 UNIQUE(唯一)索引来保证数据的唯一性  
   **备注：INSERT IGNORE INTO 与 INSERT INTO的区别**INSERT IGNORE INTO会忽略数据库中存在的数据,如果数据库没有数据则插入数据，如果有数据的话就跳过这条数据。这样可以保留数据库中已经存在的数据，达到间隙中插入数据的目的。

IBSERT IGNORE INTO 当插入数据时，在设置记录的唯一性后，如果插入重复的数据,将不返回错误,只以警告形式返回。而REPLACE INTO 如果存在primary 或 unique相同的记录，则先删除掉再插新记录。

**另一种设置数据唯一的方法:**添加一个UNIQUE索引：  
  
**统计重复数据**

1. **MySQL及SQL注入  
   防止SQL注入，我们需要注意以下几个要点：**

* 永远不要信任用户的输入。对用户的输入进行校验，可以通过正则表达式，或者限制长度;对单引号和双“-”进行转换。
* 永远不要使用动态拼装sql,可以使用参数化的sql或者直接使用储存过程进行数据查询存取。
* 永远不要使用管理员权限的数据库连接，为每个应用使用单独的权限有限的数据库连接。
* 不要把机密信息直接存放，加密或者hash掉密码和敏感信息。
* 应用的异常信息应该给出尽可能少的提示，最好用自己定义的错误信息对原始错误信息进行包装。
* SQL注入的检测方法一般采用辅助软件或者网站平台来检测，软件一般采用sql注入检测工具jsky，网站平台有忆思网站安全平台检测工具。MDCSOFT SCAN等。采用**MDCSOFT-IPS**可以有效的预防SQL注入,XSS攻击。

**PHP防止SQL注入**

PHP的MySQL扩展提供了mysqli\_real\_escape\_string()函数来转义特殊的输入字符。

