**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。**





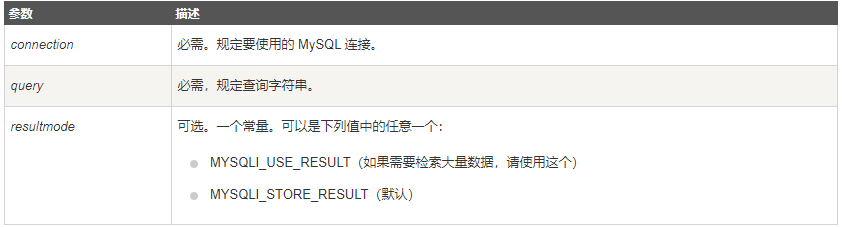


1. **MySQL创建数据表**创建数据表需要一下信息

* 表名
* 表字段名
* 定义每个表字段

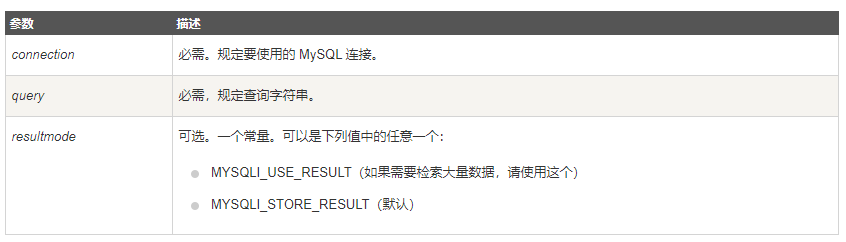
创建MySQL数据表的通用语法：CREATE TABLE table\_name(表的名字)(column\_name column\_type);  


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



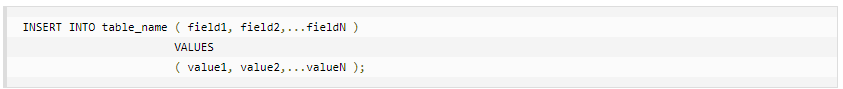
1. **删除MySQL数据表**

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

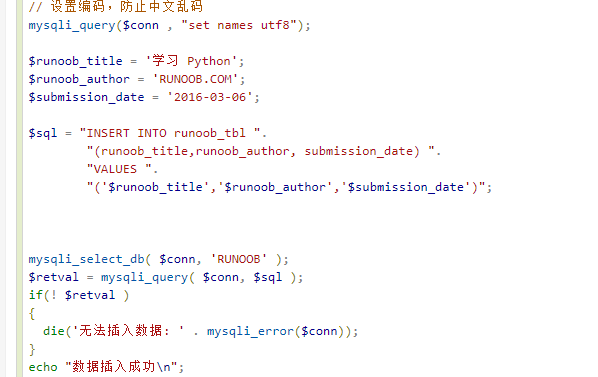


1. **MySQL插入数据**

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**

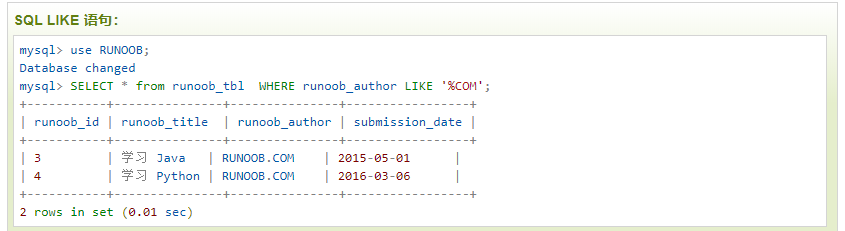
语法：DELETE FROM table\_name [WHERE Clause]

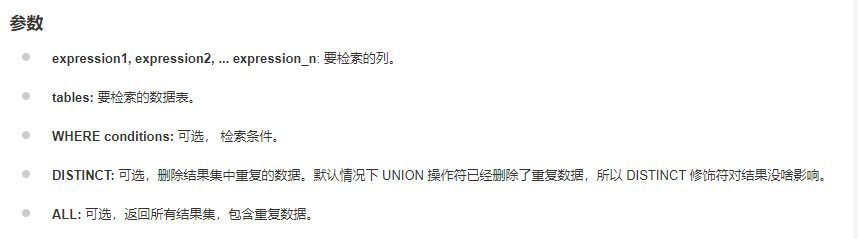
**备注:**如果不适用WHERE子句中的任何条件将**删除所有数据。**

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。  
   

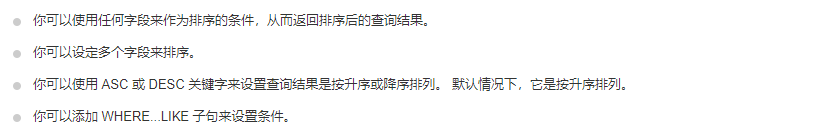
例子：



**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]]]

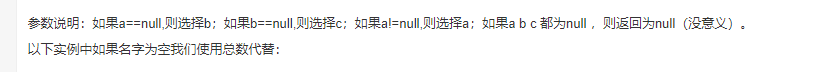


例子：SELECT \* FROM runoob\_tbl ORDER BY submission\_date ASC;  
16. **MySQL GROUP BY语句**GROUP BY语句根据一个或多个列对结果进行分组，在分组上能使用COUNT,SUM,AVG等函数。

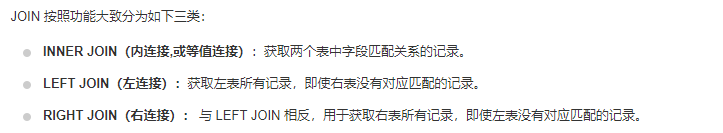
语法：SELECT column\_name,function(column\_name) FROM table\_name WHERE column\_name operator value GROUP BY column\_name

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

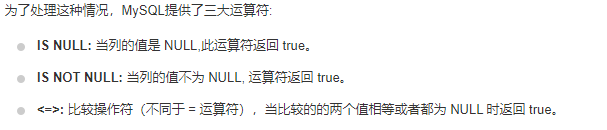
**备注：以上数据按照runoob\_id进行分组并统计出每个分组的数据的数量。**

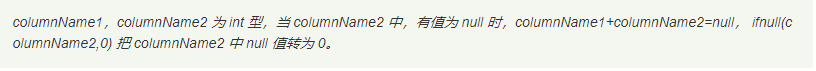
* 1. **使用WITH ROLLUP**WITH ROLLUP可以实现在分组统计数据基础上再进行相同的统计(SUN,AVG,COUNT....)  
     例子：  
     SELECT SUM(singin) as singin\_count FROM employee\_tbl GROUP BY name WITH ROLLUP  
     **备注：可以用select coalesce来去取代NULL的名称 select coalesce(a,b,c)**例子：  
     SELECT conalesce(name,’总数’) SUM(singin) as singin\_count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