**SQL语法汇总**

**一：SQL概述**

**数据定义语言(DDL)**：主要用于对数据库及数据库中的各种对象进行创建、删除、修改等操作。数据库对象

**CREATE**:用于创建数据库或数据库对象

**ALTER**:用于对数据库或数据库对象进行修改

**DROP**:用于删除数据库或数据库对象

**数据操纵语言(DML)**：主要用于操纵数据库各种对象，特别是检索和修改数据。

**SELECT**:用于从表或试图中检索数据，使用次数频繁

**INSERT**:用于将数据插入到表或视图中

**UPDATE**:用于修改表或视图中的数据，其既可修改表或视图中一行数据，也可同时修改多行或全部数据

**DELETE**:用于从表或视图中删除数据，其中可根据条件删除指定的数据

**数据控制语言(DCL)**：主要用于安全管理，例如确定哪些用户可以查看或修改数据库中的数据。

**GRANT**:用于授予权限，可把语句许可或对象许可的权限授予其他用户和角色

**REVOKE**:用于收回权限，其功能与**GRANT**相反，但不影响该用户或角色从其他角色中作为成员继承许可权限

二：**MySQL数据库模式定义语法**

1. 创建数据库语法：

CREATE {DATABASE**|**SCHEMA} [IF NOT EXISTS] db\_name

[DEFAULT] CHARACTER SET [=] charset\_name**|**

[DEFAULT] COLLATE [=] collation\_name

**“{}”:**符号标示其内容为必选项，输入命令时必写

**“[]”:**符号标示其内容为可选项,输入命令时视情况定写与不写

**“|”:**符号用于分隔花括号中的选择项，表示可任选其中一项与花括号外的语法成分共同组成SQL语句命令

**“CREATE {DATABASE|SCHEMA}”:**创建数据库命令语句

**“IF NOT EXISTS”:**关键字用于创建数据库前进行检测判断该创建的数据库是否已经存在。

若检测判断为不存在，则执行CREATE {DATABASE|SCHEMA}语句命令，创建数据库。若检测判断已存在，则不执行命令。

此外，“IF NOT EXISTS”关键字还可判断需删除的语句是否已删除，同理，若判断为已删除，则不执行删除语句命令，反之，则执行删除语句命令。

**“db\_name”:**用于标示具体的数据库名称，且该数据库名必须符合操作系统文件夹命名规则，MySQL中不区分大小写。

**“DEFAULT”:**关键字用于指定默认值

**“CHARACTER SET”:**用于指定数据库字符集(Charset)

**“COLLATE”:**关键字用于指定字符集的校对规则

举例:mysql> **create schema zhangzhikai**;

Query OK, 1 row affected (0.00 sec)

mysql> **create schema** **zhangzhikai**;

(**报错**)ERROR 1007 (HY000): Can't create database 'zhangzhikai'; database exists

mysql> **create schema if not exists zhangzhikai**;

(**不报错**)Query OK, 1 row affected, 1 warning (0.00 sec)

1. 选择数据库语法：

USE db\_name;

**“USE”:**语句可以实现从一个数据库“跳转”到另一个数据库，在用CREATE DATABASE 语句创建了数据库之后，该数据库不会自动成为当前数据库，需要用这条USE语句来指定。指定后方可对该数据库及其储存的数据对象执行各种后续操作。

举例:mysql> use zhangzhikai

Database changed

1. 修改数据库语法：

ALTER {DATABASE|SCHEMA} [db\_name]

alter\_specification ...

其中alter\_specification:

[Default] character set charset\_name

[Default] collate collation\_name

**“ALTER {DATABASE|SCHEMA}”:**修改已创建数据库的相关参数

举例:修改已有数据库zhangzhikai的默认字符集和校对规则

mysql> **alter schema** zhangzhikai

-> **default character set** gb2312默认字符集

-> **default collate** gb2312\_chinese\_ci;校对规则

Query OK, 1 row affected (0.00 sec)

1. 删除数据库语法：

DROP {DATABASE|SCHEMA} [IF EXISTS] db\_name

**“DROP {DATABASE|SCHEMA}”:**删除已创建的数据库语句，该数据库中所有表所有数据将永久删除。

**“IF EXISTS”:**可以避免删除不存在的数据库时出现的MySQL错误信息。

举例:mysql> **drop schema** zzk;

Query OK, 0 rows affected (0.01 sec)

mysql> **drop schema** zzk;

(**报错**)ERROR 1008 (HY000): Can't drop database 'zzk'; database doesn't exist

mysql> **drop schema if exists** zzk;

(**不报错**)Query OK, 0 rows affected, 1 warning (0.00 sec)

1. 查看数据库语法：

SHOW {DATABASES|SCHEMAS}

[LIKE’pattern’|WHERE expr]

**“SHOW {DATABASES|SCHEMAS}”:**查看可用数据库列表，可查看当前用户权限范围内所能查看到的所有数据库名称

**“[LIKE’pattern’|WHERE expr]”:**“like”用于匹配指定的数据库名称，“where”从句用于指定数据库名称查询范围的条件

举例:

mysql> **show schemas;**

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

| test |

| zhangzhikai |

+--------------------+

5 rows in set (0.00 sec)

mysql> **show schemas**

-> **like**'zhangzhikai';

+------------------------+

| Database (zhangzhikai) |

+------------------------+

| zhangzhikai |

+------------------------+

1 row in set (0.00 sec)

**三：MySQL数据库表定义语法**

1.创建表语法

CREATE [TEMPORARY] TABLE tbl\_name

(字段名1 数据类型 [列级完整性约束条件][默认值]

[,字段名2 数据类型 [列级完整性约束条件][默认值]]

[,......]

[,表级完整性约束条件]

)[ENGINE=引擎类型]；

**“CREATE TABLE”**:创建表语句

**“[TEMPORARY]”**:创建表时使用该语句，则创建的表为临时表，若不使用该语句，则创建的表为持久表，持久表一旦创建，则一直存在，视情况定是否使用该语句。

**“AUTO INCREMENT”:**使用该关键词，可以为表中数据类型为整型的列设置为自增属性，从而能实现当插入NULL值或数字0到一个AUTO INCREMENT列中时，该列的值会被自动设置为“此前表中该列的最大值加1”。其中，AUTO INCREMENT顺序是从1开始，同时每个表只能有一个AUTO INCREMENT列，并且它必须被索引。

**“DEFAULT”:**指定默认值。默认值是指向表插入数据时，如果没有明确给出某个表列所对应的值。在MySQL中，默认值是用CREATE TABLE语句的列定义中关键字“DEFAULT”来指定。例如在下列例子中，给cust\_sex列的描述添 “DEFAULT 0”，用于指示MySQL在未给出用户性别的情况下，使用默认值0(代表“男”)。

**“NULL”:**指没有值或缺值，在MySQL中用来指定允许NULL的列，可以在插入行时不给出该列的值。而指定**NOT NULL**值的列，则不接受该列没有值的行，即在插入或更新行时，该列必须要有值。NULL为默认设置，如果不指定NOT NULL，则认为指定的是NULL。

**“PRIMARY KEY”:**用与指定主键，例如下列例子中，指定用户的id号cut\_id作为该表的主键，主键值必须唯一。主键一定要为NOT NULL。

举例：在一个已有的数据库zhangzhikai中新建一个包含客户姓名、性别、地址、联系方式、等内容的客户基本信息表，要求将客户的id号指定为该表的主键。

mysql> USE zhangzhikai

Database changed

mysql> **CREATE TABLE** customers

-> (

-> cust\_id **INT** **NOT NULL** **AUTO\_INCREMENT**,

-> cust\_name **CHAR**(50) NOT NULL,

-> cust\_sex CHAR(1) NOT NULL **DEFAULT** 0,

-> cust\_address CHAR(50) **NULL**,

-> cust\_contact CHAR(50) NULL,

-> **PRIMARY KEY** (cust\_id)

-> );

Query OK, 0 rows affected (0.01 sec)

2.更新表数据语法

1. 添加字段子句

ADD [COLUMN]

**“ADD”:**可在ALTER TABLE语句中添加ADD [COLUMN]子句实现向表中增加新列，且其可同时增加多个列。

**“AFTER”:**可将新列插入指定的某列之后，如下列例子所示，将cust\_city列插入cust\_sex列之后。也可通过关键字“FIRST”将新列作为表的第一列。若不指定这两个关键字，系统默认将新列添加到原表的最后。

**“zhangzhikai.customers”:**此例中对表名的指定采用的是完全限定的表名方式，即“db\_name.tbl\_name”的表名格式。

举例：向数据库zhangzhikai的表customer中添加一列，并命名为cust\_city，显示用户所在城市，要求其不能为NULL，默认值为字符串‘Wuhan’,且该列位与原表cust\_sex列之后。

mysql> **ALTER TABLE** **zhangzhikai.customers**

-> **ADD COLUMN** cust\_city char(10) **NOT NULL**

**DEFAULT'Wuhan' AFTER cust\_sex**;

Query OK, 0 rows affected (0.02 sec)

Records: 0 Duplicates: 0 Warnings: 0

1. 修改字段子句

CHANGE[COLUMN]

**“CHANGE”:**可同时表中指定列的名字和数据类型，且在ALTER TABLE语句中可以同时放入多个CHANGE[COLUMN]子句，只需用逗号分隔。

举例：将数据库zhanghzhikai中表customer的cust\_sex列重命名为sex，且将其数据类型更改为长度为(1)的字符数据类型char(1),允许其为NULL,默认值为字符常量‘M’。

mysql> ALTER TABLE zhangzhikai.customers

-> **CHANGE COLUMN** cust\_sex sex char(1) NULL DEFAULT'M';

Query OK, 0 rows affected (0.02 sec)

Records: 0 Duplicates: 0 Warnings: 0

1. 修改或删除表中指定列的默认值

ALTER[COLUMN]

**“ALTER”:**可在ALTER TABLE语句中添加ALTER[COLUMN]子句，用来修改或删除表中指定列的默认值。

举例：将数据库zhangzhikai中表customer的city\_city的默认值修改为字符常量‘Beijing’.

mysql> ALTER TABLE zhangzhikai.customers

-> **ALTER COLUMN** cust\_city SET **DEFAULT**'Beijing';

Query OK, 0 rows affected (0.01 sec)

Records: 0 Duplicates: 0 Warnings: 0

1. 修改指定列的数据类型

MODIFY[COLUMN]

**“MODIFY”:**在ALTER TABLE语句中添加MODIFY[COLUMN]子句只会修改指定列的数据类型，而不会干涉他的列名。另外MODIFY[COLUMN]子句还可以通过关键字**“FIRST”**或**“AFTER”**修改指定列表中的位置。

举例：将数据库zhangzhikai中表customer的city\_name列的数据类型由之前的字符长度为50的定长字符数据类型char(50)更改为字符长度为20的定长字符数据类型char(20)，并将此列设置成表的第一列。

mysql> ALTER TABLE zhangzhikai.customers

-> **MODIFY COLUMN** cust\_name char(20) **FIRST**;

Query OK, 0 rows affected (0.02 sec)

Records: 0 Duplicates: 0 Warnings: 0

1. 删除字段子句

DROP[COLUMN]

**“DROP[COLUMN]”:**在ALTER TABLE语句中添加DROP[COLUMN]子句操作卸除多余的列。

举例：删除数据库zhangzhikai中表customers的cust\_conta列

mysql> ALTER TABLE zhangzhikai.customers

-> **DROP COLUMN** cust\_contact;

Query OK, 0 rows affected (0.02 sec)

Records: 0 Duplicates: 0 Warnings: 0

1. 重命名表名

RENAME[TO]

**“RENAME”:**在ALTER TABLE语句中添加DROP[COLUMN]子句为表重新赋予一个表名。

举例：使用RENAME[TO]子句，重命名数据库zhangzhikai中表customers的表名为zzk。

mysql> ALTER TABLE zhangzhikai.customers

-> **RENAME TO** **zhangzhikai.zzk**;

Query OK, 0 rows affected (0.01 sec)

3.重命名表语法

RENAME TABLE tbl\_name TO new\_tbl\_name

[,tbl\_name2 TO new\_tbl\_name2]...

**“RENAME TABLE”：**可同时重命名多个表

举例：将表zzzk重新命名为ak,zzzzk重新命名为aak

mysql> **RENAME TABLE** zhangzhikai.zzzk TO zhangzhikai.ak

-> ,zhangzhikai.zzzzk TO zhangzhikai.aak;

Query OK, 0 rows affected (0.01 sec)

4.删除表语法

DROP[TEMPORARY] TABLE [IF EXISTS]

tbl\_name[,tbl\_name]...

[RESTRICT|CASCADE]

“DROP[TEMPORARY]”:可以同时删除多个表(包括临时表)

举例：同时删除ak,aak。

mysql> **drop table** zhangzhikai.ak

-> ,zhangzhikai.aak;

Query OK, 0 rows affected (0.01 sec)

5.查看表语法

1. 显示表名称

SHOW[FULL]TABLES[{FROM|IN}db\_name]

**“SHOW[FULL]TABLES”:**可以查看指定数据库的所有表名

举例：

mysql> show tables;

+-----------------------+

| Tables\_in\_zhangzhikai |

+-----------------------+

| aaa |

| bbb |

| ccc |

+-----------------------+

3 rows in set (0.00 sec)

1. 显示表的结构

SHOW[FULL] COLUMNS{FROM|IN} tbl\_name [{FROM|IN} db\_name]

[LIKE’PATTERN’|WHERE expr]

或者

{DESCRIBE|DESC} tbl\_name [col\_name|wild]

**“SHOW[FULL] COLUMNS{FROM|IN}”:**用来显示指定数据表的结构

**“{DESCRIBE|DESC}”:**DESCRIBE语句作为SHOW COLUMNS FROM语句的一种快捷方式。

举例：查看zhangzhikai数据库中的aaa表

mysql> desc zhangzhikai.aaa;

+-------+---------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------+---------+------+-----+---------+-------+

| 11 | int(11) | NO | | NULL | |

| 22 | int(11) | NO | | NULL | |

| 33 | int(11) | NO | | NULL | |

| 44 | int(11) | NO | | NULL | |

+-------+---------+------+-----+---------+-------+

4 rows in set (0.01 sec)

mysql> desc zhangzhikai.kehuxinxi;

+---------+----------+------+-----+---------+----------------+

| Field | Type | Null | Key | Default | Extra |

+---------+----------+------+-----+---------+----------------+

| id | int(11) | NO | PRI | NULL | auto\_increment |

| name | char(50) | NO | | NULL | |

| sex | char(1) | NO | | 0 | |

| address | char(50) | YES | | NULL | |

| contact | char(50) | YES | | NULL | |

+---------+----------+------+-----+---------+----------------+

5 rows in set (0.01 sec)

四：索引定义

1. 索引的创建
2. CREATE INDEX语句**(最常用)**

CREATE [UNIQUE] INDEX index\_name ON

tbl\_name(col\_name[(length)][ASC|DSC])

**“[UNIQUE]”:**用于指定创建唯一性索引

**“index\_name”:**用于指定索引名，一个表可以创建多个索引，但每个索引在该表中的名称必须唯一

**“tbl\_name”:**用于指定要建立索引的表名

**“col\_name[(length)][ASC|DSC]”:**用于描述索引列

“**col\_name**”:用于指定要创建索引的列名

“**[ASC|DSC]**”:可选项，用于指定索引按升序(ASC)还是降序(DESC)，默认时为ASC

举例：mysql> create index index\_dd on aaa(dd asc);

Query OK, 0 rows affected (0.02 sec)

Records: 0 Duplicates: 0 Warnings: 0

1. CREATE TABLE语句

CREATE TABLE tbl\_name[col\_name data\_type]

[CONSTRAINT index\_name]

[UNIQUE][INDEX|KEY][index\_name](index\_col\_name[lenght])

[[ASC|DESC]

**“[CONSTRAINT[symbol]]FOREIGN KEY**

**[index\_name](index\_col\_name,...)”:**用于表示在创建表时同时创建外键。

举例1:创建表course的同时创建主键和索引，指定id为主键并设置为自增属性(A\_I)，为cname创建一个索引。

mysql> create table course

-> (

-> id int not null auto\_increment,

-> cname char(50) not null,

-> primary key(id),

-> index(cname)

-> );

Query OK, 0 rows affected (0.01 sec)

mysql> desc course;

+-------+----------+------+-----+---------+----------------+

| Field | Type | Null | Key | Default | Extra |

+-------+----------+------+-----+---------+----------------+

| id | int(11) | NO | PRI | NULL | auto\_increment |

| cname | char(50) | NO | MUL | NULL | |

+-------+----------+------+-----+---------+----------------+

2 rows in set (0.01 sec)

举例2：创建一个产品卖家信息表seller，包含了产品卖家id、姓名name、地址address、联系方式contact、当月销量sales、售卖产品类型product\_type。要求在建表同时，为该表添加由id和product\_type组成的联合主键，并在sales上创建索引。

**“primary key(col\_name,...)”:**当需要创建的主键是由多个列组成的多列索引时，用该语法创建主键。

mysql> create table seller

-> (id int not null auto\_increment,

-> name char(50) not null,address char(50) null,

-> contact char(50) null,product\_type int(5) null,

-> sales int null,primary key(id,product\_type),

-> index index\_seller(sales));

Query OK, 0 rows affected (0.00 sec)

mysql> desc seller;

+--------------+----------+------+-----+---------+----------------+

| Field | Type | Null | Key | Default | Extra |

+--------------+----------+------+-----+---------+----------------+

| id | int(11) | NO | PRI | NULL | auto\_increment |

| name | char(50) | NO | | NULL | |

| address | char(50) | YES | | NULL | |

| contact | char(50) | YES | | NULL | |

| product\_type | int(5) | NO | PRI | 0 | |

| sales | int(11) | YES | MUL | NULL | |

+--------------+----------+------+-----+---------+----------------+

6 rows in set (0.00 sec)

1. ALTER TABLE语句

ALTER TABLE tbl\_name

ADD[UNIQUE|DULLTEXT][INDEX|KEY][index\_name]

(col\_name[length][AS|desc],......)

举例，在aaa表中的ff列添加一个非唯一索引，取名为ffname。

mysql> desc aaa;

+-------+---------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------+---------+------+-----+---------+-------+

| aa | int(11) | NO | | NULL | |

| ss | int(11) | NO | | NULL | |

| dd | int(11) | NO | MUL | NULL | |

| ff | int(11) | NO | | NULL | |

| gg | int(11) | NO | | NULL | |

+-------+---------+------+-----+---------+-------+

5 rows in set (0.01 sec)

mysql> alter table aaa add index ffname(ff);

Query OK, 0 rows affected (0.02 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> desc aaa;

+-------+---------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------+---------+------+-----+---------+-------+

| aa | int(11) | NO | | NULL | |

| ss | int(11) | NO | | NULL | |

| dd | int(11) | NO | MUL | NULL | |

| ff | int(11) | NO | MUL | NULL | |

| gg | int(11) | NO | | NULL | |

+-------+---------+------+-----+---------+-------+

5 rows in set (0.01 sec)

1. 索引的查看

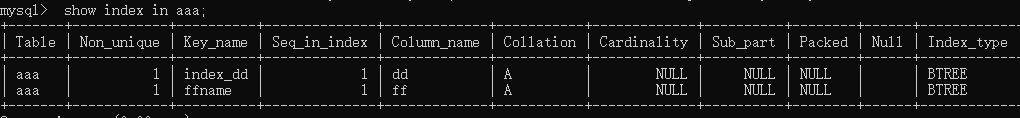
SHOW {INEDX|INDEXES|KEYS}

{FROM|IN} tbl\_name

[{FROM|IN}db\_name]

[WHERE expr]

举例:mysql> show index in aaa;



1. 索引的删除
2. Drop index

DROP INDEX index\_name on table\_name

**“index\_name”:**用于指定要删除的索引名

**“table\_name”:**用于指定该索引所在的表

举例：删除customers表中的索引index\_cust

mysql>DROP INDEX index\_cust on customers。

1. ALTER TABLE

**“[DROP PRINARY KEY]”：**用于删除表中的主键，由于一个表中只有一个主键，其也是一个索引

**“[DROP INDEX index\_name]”：**用于删除各类型的索引

**“[DROP FOREIGN KEY]”：**用于删除外键

举例；删除数据库中mysql\_test中表customers的主键和索引index\_customers.

mysql>ALTER TABLE mysql\_tes.customers

->DROP PRINARY KEY

->DROP INDEX index\_customers;

五：数据更新

1. 插入数据

INSERT [INTO] tbl\_name[(col\_name,...)]

{VALUES|VALUE}({expr|DEFAULT},...),(...),...

**“ tbl\_name”:**指定被插入数据的表名

**“col\_name”:**指定被插入数据的列名列表。注意，如向表中所有列插入数据，则全部列名均可省略

**“{VALUES|VALUE}”:**通过该关键字引导的子句，其包含各列需要插入的数据清单。注意，数据清单中数据的顺序必须与列的顺序相对应，同时该子句中的值可以是 ①:‘expr’,表示一个常量、变量、一个表达式、空值NULL，其值的数据类型要与列的数据类型一致。②:关键字‘DEFAULT’，即用于指定此列的值为该列已经明确的默认值，若该列未明确指定默认值，使用该关键字会报错，另外当列值为字符型时，需要用单引号括起。

举例：使用INSERT...VALUES语句向数据库zhangzhikai的表kehuxinxi中插入一行数据，要明确给出name列和address列的信息，即分别为“李四”和“武汉市”，而id列的值有系统自动生成，sex列选用表中默认值，contact暂不确定，可不用指定。

mysql> INSERT INTO zhangzhikai.kehuxinxi

->(id,name,sex,address,contact)

-> VALUES(0,'李四',DEFAULT,'武汉市',NULL);

Query OK, 1 row affected, 4 warnings (0.01 sec)

注意，第一列id指定数字为0，也可以设置为NULL，因为id列设置为自增属性，每次插入一行，该列都会在前一列该列值的基础上自动增量。

1. 删除数据

DELETE FROM tbl\_name

[WHERE where\_name]

[ORDER BY...]

[LIMIT row\_count]

**“DELETE”:**用于删除表里面的数据

**“tbl\_name”:**要删除数据的表名

**“[WHERE where\_name]”:**为删除操作限定删除条件，从而删除特定的行。若省略WHERE子句，则表示删除该表中所有行

**“[ORDER BY...]”:**用于将各行按照子句中指定的顺序进行删除

**“[LIMIT row\_count]”:**用于告知服务器在控制命令被返回到客户端前被删除的行的最大值

举例：删除zhangzhikai数据库的表kehuxinxi中客户名为lile的客户信息。

mysql> delete from zhangzhikai.kehuxinxi

-> where name='lilei';

Query OK, 1 row affected (0.00 sec)

1. 修改数据

UPDATE tbl\_name

SET col\_name1={exp1|DEFAULT}[,col\_name2={exp2|DEFAULT}...

[WHERE where\_condition][ORDER BY...][LIMIT roe\_count]

**“SET col\_name1={exp1|DEFAULT}”：**用于指定表中要修改的列名及其列值，其中每个指定的列值可以时表达式，也可以是该列所对应的默认值，如果是默认值，用DEFAULT关键字表示

**“WHERE”：**用于限定表中要修改的行，若不指定此语句，则UPDATE语句会修改表中所有的行

**“[ORDER BY...]”：**用于限定表中的行被修改的次序

**“[LIMIT roe\_count]”：**用于限定被修改的行数

举例：将zhangzhikai数据库的表kehuxinxi中name为‘hanmeimie’的name改为‘hanmeimei’

mysql> select\*from zhangzhikai.kehuxinxi;

+----+-----------+-----+---------+---------+

| id | name | sex | address | contact |

+----+-----------+-----+---------+---------+

| 3 | hanmeimie | 1 | hefei | NULL |

+----+-----------+-----+---------+---------+

mysql> update zhangzhikai.kehuxinxi

-> set name='hanmeimei'

-> where name='hanmeimie';

Query OK, 1 row affected (0.00 sec)

Rows matched: 1 Changed: 1 Warnings: 0

mysql> select\*from zhangzhikai.kehuxinxi;

+----+-----------+-----+---------+---------+

| id | name | sex | address | contact |

+----+-----------+-----+---------+---------+

| 3 | hanmeimei | 1 | hefei | NULL |

+----+-----------+-----+---------+---------+

六：数据跟新

1. SELECT语句

SELECT[ALL|DISTINCT|DISTINCTROW]

select\_expr[,select\_expr...]

FROM table\_references

[WHERE where\_condition][ASC|DESC],...[WITH ROLLUP]]

[HAVING where\_condition]

[ORDER BY {col\_name|expr|position}[ASC|DESC],...]

[LIMT{offset,]row\_count|roe\_count OFFSET offset]

“SELECT”：用于输出字段

“FROM table\_references”：用于指定数据的来源

“[WHERE where\_condition]”：用于指定属于的选择天剑

“ORDER BY”：用于对查询的结果进行排序

“ALL、DISTINCT、DISTINC TROW”：这三个关键字为可选项，用于指定是否应返还结果集中的重复行，若没有指定这些选项，则默认为ALL，即操作的所有行，包括重复行将返回。若指定选项DISTINCT、DISTINC TROW，则会取消结果集中的重复行那个、

|  |  |  |
| --- | --- | --- |
| 子句 | 说明 | 是否必须使用 |
| SELECT | 要返回的列或者表达式 | 是 |
| FROW | 从中检索数据的表 | 仅在从表选择数据是使用 |
| WHERE | 行级过滤 | 否 |
| GROUP BY | 分组说明 | 仅在按组计算聚合时使用 |
| HAVING | 组级过滤 | 否 |
| ORDER BY | 输出排序顺序 | 否 |
| LIMIT | 要检索的行数 | 否 |