前言:以下是MySQL最基本的增删改查语句,很多IT工作者都必须要会的命令,也是IT行业面试最常考的知识点,由于是入门级基础命令,所有所有操作都建立在单表上,未涉及多表操作。

前提:在进行"增删改查"的操作之前,先建立一个包含数据表student的数据库(具体操作可以见MySOL之最基本命令):

Field	Туре	Null	Key	Default	Extra
id	int(4)	YES		NULL	
name	varchar(20)	NO		NULL	
grade	float	YES		NULL	

1、"增"——添加数据

1.1 为表中所有字段添加数据

1.1.1 INSERT 语句中指定所有字段名

语法: INSERT INTO 表名 (字段名1,字段名2,...)

VALUES(值1,值2,...);

举例: INSERT INTO student(id,name,grade)

VALUES(1,'zhangshan',98);

使用SELECT * FROM student;命令查看结果为:



表示数据已经成功插入。

1.1.2 INSERT语句中不指定字段名

若不指定字段名,则添加的值的顺序应和字段在表中的顺序完全一致。

语法:INSERT INTO 表名 VALUES(值11,值2,...);

举例: INSERT INTO student VALUES (2,'lisi',62);

使用SELECT * FROM student;命令查看结果为:

id	name	grade
1	zhangshan	98
2	lisi	62

1.2 为表的指定字段添加数据

为指定字段添加数据,即只向部分字段添加值,而其他字段的值为表定义时的默认值。

语法: INSERT INTO 表名(字段1,字段2,...)

VALUES(值1,值2,...)

举例: INSERT INTO student(id,name)

VALUES(3,'wangwu');

使用SELECT * FROM student;命令查看结果为:

id	name	grade
1	zhangshan	98
2	lisi	62
3	wangwu	NULL

从结果中可以看出,新记录的grade字段值为NULL,是因为添加时为指明grade的值,系统会自动添加默认值。

1.3 INSERT语句的其他写法

语法: INSERT INTO 表名

SET 字段名1=值1[,字段名2=值2,...]

举例: INSERT INTO student

SET id=4 , name='zhaoliu',grade=72;

使用SELECT * FROM student;命令查看结果为:

id	name	grade
1	zhangshan	98
2	lisi	62
3	wangwu	NULL
4	zhaoliu	72

1.4 同时添加多条数据

语法:INSERT INTO 表名[(字段名1,字段名2,...)] VALUES (值1,值2,...), (值1,值2,...),

(值1,值2,...)

举例: INSERT INTO student VALUES

(5, 'lilei',99),

(6,'hanmeimei',87),

(8,'poly',76);

使用SELECT * FROM student;命令查看结果为:

id	name	grade
1	zhangshan	98
2	lisi	62
3	wangwu	NULL
4	zhaoliu	72
5	lilei	99
6	hanmeimei	87
7	poly	76

2、"删"——删除数据

语法: DELETE FROM 表名 [WHERE 条件表达式

2.1 删除部分数据

即删除指定的部分数据,需要使用WHERE子句来指定删除记录的条件。

举例:删除student表中的id值为7的记录

命令: DELETE FROM student

WHERE id=7;

使用SELECT * FROM student;命令查看结果为:

id	name	grade
1	zhangshan	98
2	lisi	62
3	wangwu	NULL
4	zhaoliu	72
5	lilei	99
6	hanmeimei	87

可见id为7的记录已被成功删除。

2.2 删除全部数据

若 DELETE 语句中没有使用WHERE语句,则会将表中所有记录都删除。

语法: DELETE FROM 表名

举例:删除student表中的所有记录命令:DELETE FROM student;

使用SELECT * FROM student;命令查看结果为:

id name grade

可见student表中记录为空,说明表中所有数据已被成功删除。

2.2 删除全部数据的另一种方法——TRUNCATE

语法:TRUNCTE[TABLE] 表名

举例:TRUNCATE TABLE student;即可删除student表

注意:

- (1) DELETE 后面可以跟WHERE子句指定删除部分记录, TRUNCATE只能删除整个表的所有记录
- (2)使用TRUNCATE语句删除记录后,新添加的记录时,自动增长字段(如本文中 student表中的 id 字段)会默认从1开始,而使用DELETE删除记录后,新添加记录时,自动增长字段会从删除时该字段的的最大值加1开始计算(即原来的id最大为5,则会从6开始计算)。所以如果是想彻底删除一个表的记录而且不会影响到重新添加记录,最好使用TRUNCATE来删除整个表的记录。

3、"改"——更新数据

更新数据指对表中现存的数据进行修改。

语法: UPDATE 表名

SET 字段名1=值1, [,字段名2=值2,...]

[WHERE 条件表达式]

在执行后面的语句之前,先使用INSERT语句往student表中插入以下数据:

id	name	grade
1	caocao	50
1	zhangshan	98
2	lisi	87
3	wangwu	92
4	zhaoliu	76
5	sunbin	87
6	boya	86
7	lilei	87

3.1 UPDATE 更新部分数据

指更新指定表中的指定记录,使用WHERE 子句来指定。

举例:将student表中id值为1=记录,将其name字段的值改为'caocao', grade字段的值改为50。

先查询之前的记录: SELECT * FROM student WHERE id=1;

显示为:



命令: UPDATE student

SET name=\caocao',grade=50
WHERE id=1;

使用SELECT * FROM student;命令查看结果为:



可见表中数据已被成功更新。

注意:还可以使用其他WHERE条件表达式,如:id > 4;

3.2 UPDATE 更新全部数据

在UPDATE 语句中若不使用WHERE 子句,则会将表中所有记录的指定字段都进行更新。

举例: 更新student表中全部记录,将grade字段都更新为80

命令: UPDATE student SET grade=80;

使用SELECT * FROM student;命令查看结果为:

name	grade
caocao	80
zhangshan	80
lisi	80
wangwu	80
zhaoliu	80
sunbin	80
boya	80
lilei	80
	caocao zhangshan lisi wangwu zhaoliu sunbin boya

可以看出所有数据已被成功更新。

4、"查"——之单表查询

MySQL从数据表中查询数据最基本的语句是SELECT语句,在前面的"增删查"已经使用过:SELECT*FROM表名,也就是查询指定数据表中的所有数据。下面将对SELECT语句进行详细介绍。

在进行后面的操作之前我们先建立一个新的数据表student2,如下:

```
CREATE TABLE student2
(
   id INT(3) PRIMARY KEY AUTO_INCREMENT,
   name VARCHAR(20) NOT NULL,
   grade FLOAT,gender CHAR(2)
);
```

此处的PRIMARY KEY表示将该字段设为主键,AUTO_INCREMENT表示将该字段设为开始值是 1,每条新记录自动递增 1,,所以在插入数据时不需为该字段设值;NOT NULL表示将该字段限制为非空值(此处不作详细讲解)

再向student2表中插入如下数据:

INSERT INTO student2(name,grade,gender)

```
VALUES ('songjiang',40,'男'),('wuyong',100,'男'),
('qinming',90,'男'),('husanniang',88,'女'),('sunerniang',66,'女'),
('wusong',86,'男'),('linchong',92,'男'),('yanqing',90,NULL);
```

id	name	grade	gender
1	songjiang	40	男
2	wuyong	100	男
3	qinming	90	男
4	husanniang	88	女
5	sunerniang	66	女
6	wusong	86	男
7	linchong	92	男
8	yanqing	90	NULL

注意:若因为此处插入的数据包含了中文而导致无法插入,可将"男/女"改为"man/woman",或者将字段编码改为utf-8(方法自行百度,此处不做讲解)。

4.1 简单查询

4.1.1 查询所有字段

语法: SELECT 字段名1, 字段名2, ...

FROM 表名

举例:查询student2表中的所有记录

命令: SELECT id,name,grade,gender

FROM student2;

结果:

id	name	grade	
1	songjiang	40	男
2	wuyong	100	男
3	qinming	90	男
4	husanniang	88	女
5	sunerniang	66	女
6	wusong	86	男
7	linchong	92	男
8	yanqing	90	NULL

注意:字段顺序可以更改,如:

SELECT id,grade,gender ,name

FROM student2;

则显示的结果也会作出对应的调整:

id	grade	gend	ler	name
1	40	男		songjiang
2	100	男		wuyong
3	90	男		qinming
4	88	女		husanniang
5	66	女		sunerniang
6	86	男		wusong
7	92	男		linchong
8	90	/	VULL	yanqing

4.1.2 在SELECT语句中使用(`*')通配符代替所有字段

语法:SELECT * FROM 表名;

在此前已多次使用此命令,所以此处不作赘述。

4.1.3 查询指定的部分字段

语法: SELECT 字段名1,字段名2, ... FROM 表名; 举例: 查询student2表中的name字段和gender字段 命令: SELECT name,gender FROM student2;

结果:

name	gender
songjiang	男
wuyong	男
qinming	男
husanniang	女
sunerniang	女
wusong	男
linchong	男
yanqing	NULL

从结果中可以看到只显示了name和gender两个字段。

4.2 按条件查询

4.2.1 带关系运算符的查询

语法: SELECT 字段名1,字段名2,... FROM 表名

WHERE 条件表达式

在WHERE子句中可以使用如下关系运算符:

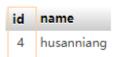
关系运算符	说 明
=	等于
<>	不等于
!=	不等于

<	小于
<=	小于等于
>	大于
>=	大于等于

举例:查询student2表中id为4的人的id和name字段

命令: SELECT id, name FROM student2 WHERE id=4;

结果:



举例:查询student2表中grade大于80的人的name和grade字段

命令: SELECT name FROM student2 WHERE grade>80;

结果:

name	grade	
wuyong	100	
qinming	90	
husanniang	88	
wusong	86	
linchong	92	
yanqing	90	

4.2.2 带 IN 关键字的查询

IN关键字用于判断某个字段的值是否在指定集合中,若在,则该字段所在的记录将会被查询出来.

语法:SELECT * | 字段名1,字段名2,...

FROM 表名

WHERE 字段名[NOT] IN (元素1,元素2,...)

举例:查询student2表中id值为1,2,3的记录

命令: SELECT * FROM student2 WHERE id IN (1,2,3);

结果:

id	name	grade	gender
1	songjiang	40	男
2	wuyong	100	男
3	qinming	90	男

注意: NOT IN 与 IN 相反,查询的是不在指定范围内的记录。

4.2.3 带 BETWEEN AND 关键字的查询

BETWEEN AND 用于判断某个字段的值是否在指定范围之内,若在,则该字段所在的记录会被查询出来,反之不会。

语法:SELECT * | { 字段名1,字段名2, ... }

FROM 表名

WHERE 字段名[NOT] BETWEEN 值1 AND 值2;

举例:查询student2表中id值在2~5之间的人的id和name

命令: SELECT id,name FROM students WHERE id BETWEEN 2 AND 5;

结果:

id	name
2	wuyong
3	qinming
4	husanniang
5	sunerniang

注意:NOT BETWEEN AND 表示查询指定范围外的记录。

4.2.4 空值查询

在数据表中有些值可能为空值(NULL),空值不同于0,也不同于空字符串,需要使用 IS NULL 来判断字段的值是否为空值。

语法:SELECT * | 字段名1,字段名2,...

FROM 表名

WHERE 字段名 IS [NOT] NULL

举例:查询student2表中gender值为空值的记录。

命令: SELECT * FROM student2 WHERE gender IS NULL;

结果:

id	name	grade	gender
8	yanqing	90	NULL

注意: IS NOT NULL 关键字用来查询字段不为空值的记录。

4.2.5 带 DISTINCT 关键字的查询

很多表中某些字段的数据存在重复的值,可以使用DISTINCT关键字来过滤重复的值,只保留一个值。

语法: SELECT DISTINCT 字段名 FROM 表名;

举例:查询student2表中gender字段的值,结果中不允许出行重复的值。

命令: SELECT DISTINCT gender FROM student2;

结果:

gender

男

女

NULL

注意:DISTINCT 关键字还可作用于多个字段,则只有多个字段的值都完全相同时才会被认作是重复记录。

4.2.6 带 LIKE 关键字的查询

语法: SELECT * | 字段名1,字段名2,...

FROM 表名

WHERE 字段名[NOT]LIKE '匹配字符串';

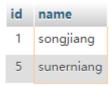
(1)百分号(%)通配符

匹配任意长度的字符串,包括空字符串。例如,字符串"c%"匹配以字符 c 开始,任意长度的字符串,如"ct","cut","current"等;字符串"c%g"表示以字符 c 开始,以 g 结尾的字符串;字符串"%y%"表示包含字符"y"的字符串,无论"y"在字符串的什么位置。

举例1:查询student2表中name字段以字符"s"开头的人的id,name

命令: SELECT id,name FROM student2 WHERE name LIKE "S%";

结果:



举例2:查询student2表中name字段以字符"w"开始,以字符"g"结尾的人的id,name。

命令: SELECT id,name FROM student2 WHERE name LIKE 'w%g';

结果:



举例3:查询student2表中name字段不包含"y"的人的id, name。

命令: SELECT id,name FROM student2 WHERE name NOT LIKE '%y%';

结果:

id	name
1	songjiang
3	qinming
4	husanniang
5	sunerniang
6	wusong
7	linchong

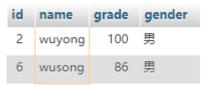
(2)下划线(__)通配符

下划线通配符只匹配单个字符,若要匹配多个字符,需要使用多个下划线通配符。例如,字符串"cu_"匹配以字符串"cu"开始,长度为3的字符,如"cut","cup";字符串"c__l"匹配在"c"和"l"之间包含两个字符的字符串,如"cool"。需要注意的是,连续的"_"之间不能有空格,例如"M__QL"只能匹配"My SQL",不能匹配"MySQL"。

举例:查询在student2表中name字段值以"wu"开始,以"ong"结束,并且中间只有一个字符的记录。

命令: SELECT * FROM student2 WHERE name LIKE 'wu_ong';

结果:



注意:若要查询的字段值本来就含有"%"或者"_",则要用"\"进行转义,如要查询本身含有"%"的字符串,命令应改为"%\%%"。

4.2.7 带 AND 关键字的多条件查询

在使用SELECT语句查询数据时,优势为了使查询结果更加精确,可以使用多个查询条件,如使用 AND 关键字可以连接两个或多个查询条件。

语法: SELECT * | 字段名1, 字段名2, ...

FROM 表名

WHERE 条件表达式1 AND 条件表达式2 [... AND 条件表达式 n];

举例:查询student2表中 id 字段小于5,并且 gender 字段值为"女"的人的id和name

命令: SELECT id,name FROM student2 WHERE id<5 AND gender='女';

结果:

id name
4 husanniang

4.2.8 带 OR 关键字的多条件查询

与 AND 关键字不同, OR 关键字只要满足任意一个条件就会被查询出来

语法:SELECT * | 字段名1,字段名2,...

FROM 表名

WHERE 条件表达式1 OR 条件表达式2 [... OR 条件表达式 n];

举例:查询student2表中 id 字段小于3,或者 gender 字段值为"女"的人的id, name和

gender

命令: SELECT id,name ,gender FROM student2 WHERE id<3 OR gender='女';

结果:

id	name	gender
1	songjiang	男
2	wuyong	男
4	husanniang	女
5	sunerniang	女

4.2.9 OR 和 AND 一起使用的情况

OR 和 AND 一起使用的时候,AND 的优先级高于 OR,因此二者一起使用时,会先运算 AND 两边的表达式,再运算 OR 两边的表达式。

举例:查询student2表中gender值为"女"或者gender值为"男"并且grade字段值为100的人的记录

命令: SELECT * FROM student2 WHERE gender='女' OR gender='男' AND grade=100; 结果:

id	name	grade	gender
2	wuyong	100	男
4	husanniang	88	女
5	sunerniang	66	女

4.3 高级查询

4.3.1 聚合函数

函数名称	作用
COUNT()	返回某列的行数
SUM()	返回某列值的和
AVG()	返回某列的平均值

MAX()	返回某列的最大值
MIN()	返回某列的最小值

(1) COUNT()函数:统计记录的条数

语法: SELECT COUNT(*) FROM 表名

举例:查询student2表中一共有多少条记录命令:SELECT COUNT(*) FROM student2;

结果: count(*)

8

(2)SUM()函数:求出表中某个字段所有值的总和

语法:SELECT SUM(字段名)FROM 表名; 举例:求出student2表中grade字段的总和

命令: SELECT SUM(grade) FROM student2;

结果:

SUM(grade)

652

(3) AVG()函数:求出表中某个字段所有值的平均值

语法: SELECT AVG(字段名) FROM 表名; 举例: 求出student2表中grade字段的平均值

命令: SELECT AVG(grade) FROM student2;

结果:

AVG(grade)

81.5

(4) MAX()函数:求出表中某个字段所有值的最大值

语法: SELECT MAX(字段名) FROM 表名;

举例:求出student2表中所有人grade字段的最大值命令:SELECT MAX(grade) FROM student2;

结果:

max(grade)

100

(5) MIN()函数:求出表中某个字段所有值的最小值

语法: SELECT MIN(字段名) FROM 表名;

举例:求出student2表中所有人grade字段的最小值命令:SELECT MIN(grade) FROM student2;

结果:

min(grade)

40

4.3.2 对查询结果进行排序

语法: SELECT 字段名1,字段名2,... FROM 表名 ORDER BY 字段名1 [ASC | DESC],字段名2 [ASC | DESC]...

在该语法中指定的字段名是对查询结果进行排序的依据,ASC表示升序排列,DESC表示降序排列,默认情况是升序排列。

举例1:查出student2表中的所有记录,并按照grade字段进行升序排序

命令: SELECT * FROM student2

ORDER BY grade;

结果:

id	name	grade	<u>.</u> 1	gender
1	songjiang		40	男
5	sunerniang		66	女
6	wusong		86	男
4	husanniang		88	女
3	qinming		90	男
8	yanqing		90	NULL
7	linchong		92	男
2	wuyong		100	男

举例2:查出student2表中的所有记录,并按照grade字段进行降序排序

命令: SELECT * FROM student2 ORDER BY grade DESC;

结果:

-H/I	•			
id	name	grade	y 1	gender
2	wuyong		100	男
7	linchong		92	男
3	qinming		90	男
8	yanqing		90	NULL
4	husanniang		88	女
6	wusong		86	男
5	sunerniang		66	女
1	songjiang		40	男

4.3.3 分组查询

在对表中数据进行统计的时候,可以使用GROUP BY 按某个字段或者多个字段进行分组,字段中值相同的为一组,如男生分为一组,女生分为一组。

语法: SELECT 字段名1,字段名2,...

FROM 表名

GROUP BY 字段名1,字段名2, ... [HAVING 条件表达式];

(1)单独使用 GROUP BY 进行分组

单独使用GROUP BY 关键字,查询的是每个分组中的一条记录

举例:查询student2表中的数据,按照gender字段进行分组。

命令: SELECT * FROM student2 GROUP BY gender;

结果:

id	name	grade	gender
8	yanqing	90	NULL
4	husanniang	88	女
1	songjiang	40	男

注意:若在此2例中键入命令之后出现"#1055"错误,无法完成分组查询,是由于MySQL版本升级到了5.7,可自行百度"MySQL 1055错误"参考解决办法。

(2) GROUP BY 和聚合函数一起使用

GROUP BY 和聚合函数一起使用,可以统计出某个或者某些字段在一个分组中的最大值、最小值、平均值等。

举例:将student2表按照gender字段进行分组查询,计算出每组共有多少个人。 命令:SELECT COUNT(*),gender FROM student2 GROUP BY gender;

结果:

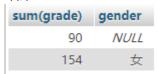


(2) GROUP BY 和 HAVING 关键字一起使用

HAVING关键字和WHERE关键字的作用相同,区别在于HAVING 关键字可以跟聚合函数,而WHERE 关键字不能。通常HAVING 关键字都和GROUP BY一起使用,用于对分组后的结果进行过滤。

举例:将student2表按照gender字段进行分组查询,查询出grade字段值之和小于300的分组命令:SELECT sum(grade),gender FROM student2 GROUP BY gender HAVING SUM(grade) < 300;

结果:



4.3.4 使用 LIMIT 限制查询结果的数量

语法: SELECT 字段名2, 字段名2, ...

FROM 表名

LIMIT [OFFSET ,] 记录数

在此语法中,LIMIT 后面可以跟两个参数,第一个参数"OFFSET "表示偏移量,如果偏移量为0,则从查询结果的第一条记录开始,偏移量为1则从查询结果中的第二条记录开始,以此类推。OFFSET为可选值,默认值为0,第二个参数"记录数"表示指定返回查询记录的条数。

举例1:查询student2表中的前四条记录。

命令: SELECT * FROM student LIMIT 4;

结果:

id	name	grade	gender
1	songjiang	40	男
2	wuyong	100	男
3	qinming	90	男
4	husanniang	88	女

举例2:查询student2表中grade字段从第五位到第八位的人(从高到低)命令:SELECT * FROM student2 ORDER BY grade DESC LIMIT 4,4;

结果:

id	name	grade 🔻 1	gender
4	husanniang	88	女
6	wusong	86	男
5	sunerniang	66	女
1	songjiang	40	男

4.3.5 函数(列表)

见MySQL之常用函数

4.4 为表和字段取别名

4.4.1 为表取别名

在进行查询操作时,如果表名很长使用起来不方便,可以为表取一个别名来代替表的名称。

语法: SELECT * FROM 表名[AS]别名;

举例:为student2表起一个别名s,并查询student2表中gender字段值

为"女"的记录

命令: SELECT * FROM student2 AS s WHERE s.gender='女';

结果:

id	name	grade	gender
4	husanniang	88	女
5	sunerniang	66	女

4.4.1 为字段取别名

语法: SELECT 字段名[AS]别名[,字段名[AS]别名,...] FROM 表名;

举例:查询student2表中的所有记录的name和gender字段值,并未这两个字段起别名

stu_name和stu_gender

命令: SELECT name AS stu_name,gender AS stu_gender FROM student2;

结果:

stu_name	stu_gender
songjiang	男
wuyong	男
qinming	男
husanniang	女
sunerniang	女
wusong	男
linchong	男
yanqing	NULL