数据库查询

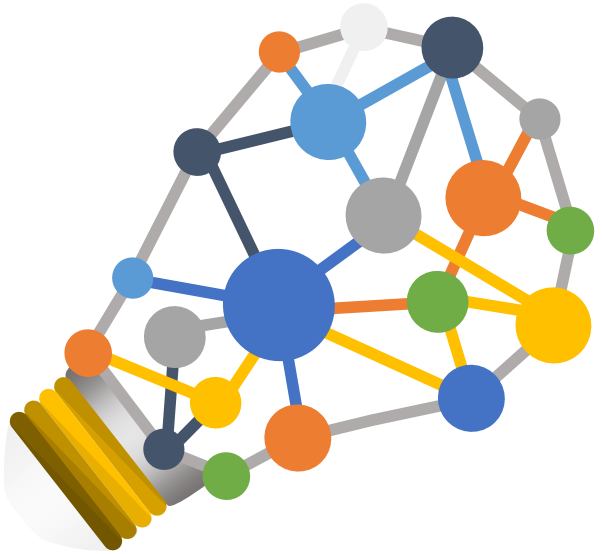
04

Select语句的操作符

SELECT语句的操作符

- 算术操作符
+（加号）、－（减号）、*（乘号）和 /（除号）。

- 比较操作符
=（等于）、>（大于）、<（小于）、<=（小于等于）、>=（大于等于）、!=或<>（不等于）、!>（不大于）和 !<（不小于），共9种操作符。



聚合类函数

函数名称	功能
AVG	按列计算平均值
SUM	按列计算值的总和
MAX	求一列中的最大值
MIN	求一列中的最小值
COUNT	按列值计个数

SQL查询

SELECT语句的语法

```
SELECT〈目标列组〉
    FROM〈数据源〉
    [WHERE〈元组选择条件〉]
    [GROUP BY〈分列组〉[HAVING 〈组选择条件〉]]
    [ORDER BY〈排序列1〉〈排序要求1〉[，…n]];
```

```
-- 对大气质量表进行有选择的查询
select city_name, avg(pm25), avg(pm10)
        from Monthly_Indicator
        where pm25 > 50
        group by city_name, month_key having city_name <> '北京'
        order by avg(pm25) desc;
```

单表查询练习

```
-- 查询大气质量表中的全部内容
select *
from monthly_indicator;

-- 查询北京的大气质量数据
select *
from monthly_indicator
where city_name = '北京’ ;

-- 查询不同月份PM2.5的最大值
select month_key, max(pm25)
from monthly_indicator
group by month_key;

-- 降序查询不同城市PM10的平均值
select city_name, avg(pm10)
from monthly_indicator
group by city_name
order by avg(pm10) desc;
```

多表查询

多表查询指的是将两个以上的数据表通过关键字段连接在一起，并从不同表中取不同字段进行查询的方法

关键字段： 用来连接两表的内容信息能够匹配的上的字段

1. 相连的两表中都需要有关键字段
2. 关键字段中的记录信息能够匹配的上
3. 最理想的连接状态是两表中的两个关键字段都是主键，而且两个主键的值能够一一匹配的上

“SELECT <select_list> FROM <表1> xx join <表2> on 表1.key = 表2.key”语句连接两表

1. xx代表链接的方向，可以是inner、left、right等关键字
2. 在连接语句前边的表是 “左表”，在连接语句后边的表是“右表”

select 学员信息表.*, 学员成绩表.* from 学员信息表 left join 学员成绩表 on 学员信息表.学号 = 学员成绩表.学号;

学员信息表

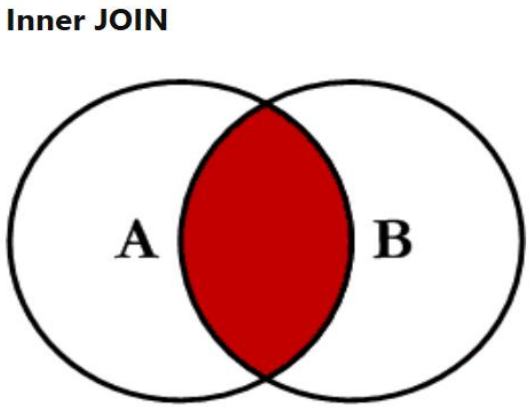
学号	学员姓名	年龄
a	赵大	16
b	钱二	16
c	张三	17
d	李四	17

学员成绩表

学号	成绩
a	50
b	60
c	70
d	80

内连接查询(inner join)

内连接：
按照连接条件合并两个表，返回满足条件的行。
SELECT <select_list> FROM A
INNER JOIN B ON A.Key = B.Key;



select 学员信息表.*, 学员成绩表.* from 学员信息表 inner join 学员成绩表 on 学员信息表.学号 = 学员成绩表.学号;

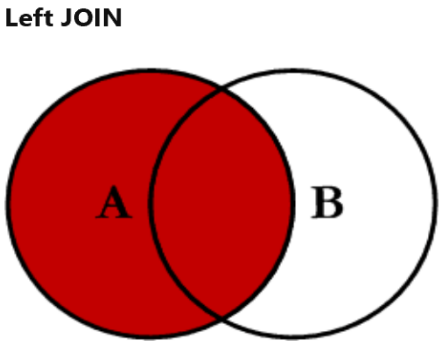
学号	学员姓名	年龄
a	赵大	16
b	钱二	16
c	张三	17
d	李四	17
e	王五	16
a	赵大	16

学号	成绩
a	50
b	60
c	70
d	80
a	50
f	90

学员信息表			学员成绩表	
学号	学员姓名	年龄	学号	成绩
a	赵大	16	a	50
a	赵大	16	a	50
a	赵大	16	a	50
a	赵大	16	a	50
b	钱二	16	b	60
c	张三	17	c	70
d	李四	17	d	80

左连接查询(left join)

左连接：
结果中除了包括满足连接条件的行外，还包括左表的所有行
SELECT <select_list> FROM A
LEFT JOIN B ON A.Key = B.Key;



select 学员信息表.*, 学员成绩表.* from 学员信息表 left join 学员成绩表 on 学员信息表.学号 = 学员成绩表.学号;

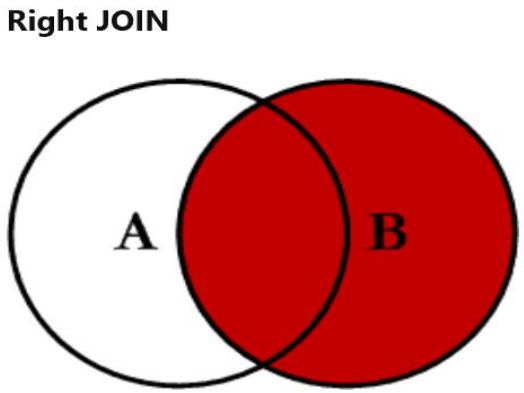
学号	学员姓名	年龄
a	赵大	16
b	钱二	16
c	张三	17
d	李四	17
e	王五	16
a	赵大	16

学号	成绩
a	50
b	60
c	70
d	80
a	50
f	90

学员信息表			学员成绩表	
学号	学员姓名	年龄	学号	成绩
a	赵大	16	a	50
a	赵大	16	a	50
a	赵大	16	a	50
a	赵大	16	a	50
b	钱二	16	b	60
c	张三	17	c	70
d	李四	17	d	80
e	王五	16	null	null

右连接查询(right join)

右连接：
结果中除了包括满足连接条件的行外，还包括右表的所有行
SELECT <select_list> FROM A
RIGHT JOIN B ON A.Key = B.Key;



select 学员信息表.*, 学员成绩表.* from 学员信息表 right join 学员成绩表 on 学员信息表.学号 = 学员成绩表.学号;

学号	学员姓名	年龄
a	赵大	16
b	钱二	16
c	张三	17
d	李四	17
e	王五	16
a	赵大	16

学号	成绩
a	50
b	60
c	70
d	80
a	50
f	90

学员信息表			学员成绩表	
学号	学员姓名	年龄	学号	成绩
a	赵大	16	a	50
a	赵大	16	a	50
a	赵大	16	a	50
a	赵大	16	a	50
b	钱二	16	b	60
c	张三	17	c	70
d	李四	17	d	80
null	null	null	f	90

联合查询

union: 用于合并两个或多个 SELECT 语句的结果集，并消去表中任何重复行。

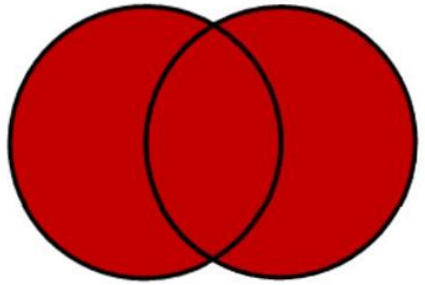
例： 用union合并t1与t2表
```
select t1.*  from t1
union
select t2.* from t2;
```

union all:用于合并两个或多个 SELECT 语句的结果集，保留重复行。

例： 用union all合并t1与t2表
```
select t1.*  from t1
union all
select t2.* from t2;
```

全连接查询

全连接：
全连接会返回两张表中全部的记录,本质上是对两个表中的记录取并集。
全连接没有主附表的区别，按照连接条件可以匹配到的记录会返回匹配后的结果，匹配不到的记录用null进行填充。



```
select * from t1 left join t2 on key1=key2
union
select * from t1 right join t2 on key1=key2;
```

查询操作符与子查询

05

水果价格表

f_id	s_id	f_name	f_price
a1	101	apple	5.20
b1	101	blackberry	10.20
c0	101	cherry	3.20
bs1	102	orange	11.20
t1	102	banana	10.30
t2	102	grape	5.30
a2	103	apricot	25.20
o2	103	coconut	9.20
b2	104	berry	7.60
l2	104	lemon	6.40
bs2	105	melon	8.20
m2	105	xbabay	2.60
m1	106	mango	15.60
b5	107	xxxx	3.60
t4	107	xbababa	3.60

```
-- 创建fruits数据表
create table fruits(
f_id char(10) not null,
s_id int not null,
f_name varchar(255) not null,
f_price decimal(8,2) not null,
primary key(f_id)
);。
```

字段说明：

f_id： 水果ID
s_id： 品类ID
f_name： 水果名称
f_price： 水果价格

操作符（1）

AND：用来联合多个条件进行查询，条件与条件间是“和”的意思
　　条件表达式1 AND 条件表达式2 【…AND 条件表达式n】
例：用and操作符查询s_id为101并且f_id为a1的水果记录
select * from fruits where s_id = 101 and f_id = 'a1’ ;

OR：用来联合多个条件进行查询，条件与条件间是“或”的意思
　　条件表达式1 OR 条件表达式2 【…OR 条件表达式n】
例：用or操作符查询苹果或者橙子的相关记录
select * from fruits where f_name = 'apple' or f_name = 'orange’ ;

IN：判断某个字段的值是否在指定的集合中，如果在集合中则满足查询条件，如果不在则不满足查询条件
　　【NOT】 IN(元素1,元素2,…,元素n)
※其中NOT是可选参数，加上NOT表示不在集合内满足条件
例：用in操作符查询苹果和橙子的相关记录
select * from fruits where f_name in('apple','orange’);
例：用not in操作符查询苹果和橙子之外的水果的相关记录
select * from fruits where f_name not in('apple','orange’);

操作符（2）

BETWEEN：判断某个字段的值是否在指定的范围内，如果在则满足查询条件，如果不在则不满足查询条件

【NOT】 BETWEEN 取值1 AND 取值2

※其中NOT是可选参数，加上NOT表示不在指定范围内满足条件

例：用between...and操作符查询f_price在10元到20元之间的水果记录

```
select * from fruits where f_price between 10 and 20;
```

LIKE：用来匹配字符串是否相等，如果字段的值与指定的字符串相匹配，则满足查询条件，如果与指定的字符串不匹配，则不满足查询条件

【NOT】 LIKE ‘字符串’

※其中NOT是可选参数，加上NOT表示指定的字符串不匹配时满足条件，字符串参数的值可以是一个完整的字符串，也可是包含%或者_的通配符。其中%代表任意长度的字符串。例如b%k表示以字母b开头，以字母k结尾的任意长度的字符串。比如bak，book，break等都可以。而_只能表示单个字符。例如b_k表示以字母b开头，以字母k结尾的3个字符。只有上例中的bak是匹配项，而book与break均不满足匹配要求。

例：用like操作符查询所有f_id由b开始且字符长度为两位的水果记录

```
select * from fruits where f_id like 'b_’ ;
```

操作符（3）

IS NULL：用来判断字段的值是否为空值(NULL)。如果字段的值为空值，则满足查询条件，如果字段的值是非空值，则不满足查询条件

IS 【NOT】 NULL

※其中NOT是可选参数，加上NOT表示字段不是空值时满足条件。

例：用is null操作符查询所有f_name为空的水果记录

```
select * from fruits where f_name is null;
```

DISTINCT: 用来消除重复记录。

SELECT DISTINCT 字段名

例： 查询fruits表中所有不重复的s_id

```
select distinct s_id from fruits;
```

操作符与子查询的组合应用

子查询：写在()中，把内层查询结果当做外层查询参照的数据表来用

ANY：表示满足其中任意一个条件，使用Any关键字时，只要满足内层查询语句返回的结果中的任何一个，就可以通过该条件来执行外层查询语句
例： 用any操作符与子查询语句来查询所有f_id对应的f_price在10元到20元之间的水果记录

```
select * from fruits where f_id =  any
(select f_id from fruits where f_price between 10 and 20);
```

ALL：表示满足所有条件，使用All时，只有满足内层查询语句返回的所有结果，才可以执行外层查询语句
例： 用all操作符与子查询语句来查询所有f_price大于20元的水果记录

```
select * from fruits where f_price > all
(select f_price from fruits where f_price <= 20);
```

EXISTS：表示存在，使用Exists关键字时，内层查询语句不返回查询的记录，而是返回一个真假值，如果内层查询语句查询到满足条件的记录，就返回真值，否则返回假值，当返回真值是，外层查询语句将进行查询，当返回假值时，外层查询语句不进行查询或者查询不出任何记录
例： 用exists操作符与子查询语句来查询是否存在f_price大于30元的水果记录

```
select * from fruits where exists
(select * from fruits where f_price > 30);
```

as重命名与limit限制查询结果行数

as：可以将表或字段名重新命名为别的名称使用，只在查询中有效

例： 用as将fruits表名重命名为f后使用
```
select f.* from fruits as f;
```

limit：查询后只显示limit指定数字的行数结果

例： 显示f_price金额最大的前三名水果记录
```
select * from fruits
order by f_price desc
limit 3;
```

函数

06

常用的数学函数

常用的数学函数：主要用于处理数字值

函数	说明
ABS(x)	返回 x 的绝对值
BIN(x)	返回 x 的二进制（OCT 返回八进制，HEX 返回十六进制）
EXP(x)	返回值 e（自然对数的底）的 x 次方
GREATEST(x1,x2,...,xn)	返回集合中最大的值
LEAST(x1,x2,...,xn)	返回集合中最小的值
LN(x)	返回 x 的自然对数
LOG(x,y)	返回 x 的以 y 为底的对数
MOD(x,y)	返回 x/y 的模（余数）
PI()	返回 pi 的值（圆周率）
RAND()	返回 0 到 1 内的随机值,可以通过提供一个参数(种子)使 RAND()随机数生成器生成一个指定的值。
FLOOR(x)	返回小于 x 的最大整数值，（去掉小数取整）
CEILING(x)	返回大于 x 的最小整数值，（进一取整）
ROUND(x,y)	返回参数 x 的四舍五入的有 y 位小数的值，（四舍五入）
TRUNCATE(x,y)	返回数字 x 截短为 y 位小数的结果
SIGN(x)	返回数字 x 的符号的值（正数返回 1，负数返回-1，0 返回 0）
SQRT(x)	返回一个数的平方根

常用的字符串函数

常用的字符串函数：主要用于处理字符串值

函数	说明
ASCII(char)	返回字符的 ASCII 码值
BIT_LENGTH(str)	返回字符串的比特长度
CONCAT(s1,s2...,sn)	将 s1,s2...,sn 连接成字符串
CONCAT_WS(sep,s1,s2...,sn)	将 s1,s2...,sn 连接成字符串，并用 sep 字符间隔
INSERT(str,x,y,instr)	将字符串 str 从第 x 位置开始，y 个字符长的子串替换为字符串 instr，返回结果
FIND_IN_SET(str,list)	分析逗号分隔的 list 列表，如果发现 str，返回 str 在 list 中的位置
LCASE(str)或 LOWER(str)	返回将字符串 str 中所有字符改变为小写后的结果
UCASE(str)或 UPPER(str)	返回将字符串 str 中所有字符转变为大写后的结果
LEFT(str,x)	返回字符串 str 中最左边的 x 个字符
RIGHT(str,x)	返回字符串 str 中最右边的 x 个字符
LENGTH(str)	返回字符串 str 中的字符数
POSITION(substr,str)	返回子串 substr 在字符串 str 中第一次出现的位置
QUOTE(str)	用反斜杠转义 str 中的单引号
REPEAT(str,srchstr,rplcstr)	返回字符串 str 重复 x 次的结果
REVERSE(str)	返回颠倒字符串 str 的结果
LTRIM(str)	去掉字符串 str 开头的空格
RTRIM(str)	去掉字符串 str 尾部的空格
TRIM(str)	去除字符串首部和尾部的所有空格

常用的日期及时间函数

日期及时间函数：用来处理日期时间型数据

函数	说明
DATE_FORMAT(date,fmt)	依照指定的 fmt 格式格式化日期 date 值
FROM_UNIXTIME(ts,fmt)	根据指定的 fmt 格式，格式化 UNIX 时间戳 ts
MONTHNAME(date)	返回 date 的月份名(英语月份，如 October)
DAYNAME(date)	返回 date 的星期名(英语星期几，如 Saturday)
NOW()	返回当前的日期和时间
CURDATE()或 CURRENT_DATE()	返回当前的日期
CURTIME()或 CURRENT_TIME()	返回当前的时间
QUARTER(date)	返回 date 在一年中的季度(1~4)
WEEK(date)	返回日期 date 为一年中第几周(0~53)
DAYOFYEAR(date)	返回 date 是一年的第几天(1~366)
DAYOFMONTH(date)	返回 date 是一个月的第几天(1~31)
DAYOFWEEK(date)	返回 date 所代表的一星期中的第几天(1~7)
YEAR(date)	返回日期 date 的年份(1000~9999)
MONTH(date)	返回 date 的月份值(1~12)
DAY(date)	返回 date 的天数部分
HOUR(time)	返回 time 的小时值(0~23)
MINUTE(time)	返回 time 的分钟值(0~59)
SECOND(time)	返回 time 的秒值（0-59）
DATE(datetime)	返回 datetime 的日期值
TIME(datetime)	返回 datetime 的时间值

其他函数

其他函数：除上述函数之外的一些常用函数

函数	说明
GROUP_CONCAT(col)	返回由属于一组的列值连接组合而成的结果
CAST()	将一个值转换为指定的数据类型

※GROUP_CONCAT()函数：常与关键字 GROUP BY 一起使用，能够将分组后指定的字段值都显示出来。

例： 使用group_concat函数查询不同s_id下对应的所有f_name信息

```
SELECT s_id, GROUP_CONCAT(f_name) FROM fruits
GROUP BY s_id;
```

其他函数

其他函数：除上述函数之外的一些常用函数

函数	说明
GROUP_CONCAT(col)	返回由属于一组的列值连接组合而成的结果
CAST()	将一个值转换为指定的数据类型

※GROUP_CONCAT()函数：常与关键字 GROUP BY 一起使用，能够将分组后指定的字段值都显示出来。

例： 将文本字符串'12'转换为整型12

```
SELECT CAST('12' AS int)
```

其他函数 – 逻辑函数

逻辑函数用来对表达式进行判断，根据满足的条件不同，执行相应的流程

1. 空值函数
ifnull(expression, alt_value)
-- 查询每位员工的实发工资（基本工资+奖金）
select ename,sal+ifnull(comm,0) 实发工资 from emp;

2. 判断函数
if(expr1,expr2,expr3)

3.逻辑表达式
case when expr1 then expr2 [when expr3 then expr4...else expr] end;
-- 查询员工的工资级别，3000及以上为高，1500及以下为低，其余为中

-- 使用if函数
```
select ename,sal,
if(sal>=3000,'高',if(sal<=1500,'低','中')) 工资级别
from emp;
```

-- 使用case
```
select ename,sal,
      case when sal>=3000 then '高'
when sal<=1500 then '低' else '中' end 工资级别
from emp;
```

其他函数 – 开窗函数

对数据的每一行，都使用与该行相关的行进行计算并返回计算结果，有几条记录执行完返回结果还是几条

1. over()函数

开窗函数名([<字段名>]) over([partition by <分组字段>] [order by <排序字段> [desc]] [< 滑动窗口>])

```
-- 聚合函数，返回公司所有员工的平均工资
select avg(sal) 平均工资 from emp;
```

平均工资
2073.2143

```
-- 开窗函数，查看每位员工与公司所有员工的平均工资之间的情况
select *,avg(sal) over() 平均工资 from emp;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno	平均工资
7369	smith	clerk	7902	1980-12-17	800	NULL	20	2073.2143
7499	allen	salesman	7698	1981-02-20	1600	300	30	2073.2143
7521	ward	salesman	7698	1981-02-22	1250	500	30	2073.2143
7566	jones	manager	7839	1981-04-02	2975	NULL	20	2073.2143
7654	martin	salesman	7698	1981-09-28	1250	1400	30	2073.2143
7698	blake	manager	7839	1981-05-01	2850	NULL	30	2073.2143
7782	clark	manager	7839	1981-06-09	2450	NULL	10	2073.2143
7788	scott	analyst	7566	1987-04-19	3000	NULL	20	2073.2143
7839	king	persident	NULL	1981-11-17	5000	NULL	10	2073.2143
7844	turner	salesman	7698	1981-09-08	1500	0	30	2073.2143
7876	adams	clerk	7788	1987-05-23	1100	NULL	20	2073.2143
7900	james	clerk	7698	1981-12-03	950	NULL	30	2073.2143
7902	ford	analyst	7566	1981-12-03	3000	NULL	20	2073.2143
7934	miller	clerk	7782	1982-01-23	1300	NULL	10	2073.2143

其他函数 – 开窗函数

2. partition by子句
partition by类似于group by子句，在over()函数使用它来指定用来分组的一个或者多个字段，开窗函数在不同的分组内分别执行聚合运算，并将每个组的计算聚合结果显示在组内每条记录中

-- 聚合函数，查询各部门的平均工资
select deptno,avg(sal) from emp group by deptno;

deptno	avg(sal)
10	2916.6667
20	2175.0000
30	1566.6667

-- 开窗函数，查询每位员工与所属部门平均工资之间的情况
select *,avg(sal) over(partition by deptno) 平均工资 from emp;

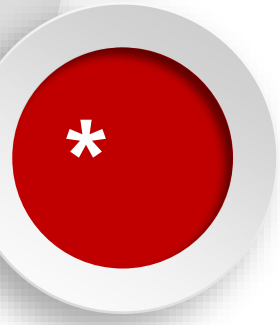
empno	ename	job	mgr	hiredate	sal	comm	deptno	平均工资
7782	clark	manager	7839	1981-06-09	2450	NULL	10	2916.6667
7839	king	persident	NULL	1981-11-17	5000	NULL	10	2916.6667
7934	miller	clerk	7782	1982-01-23	1300	NULL	10	2916.6667
7369	smith	clerk	7902	1980-12-17	800	NULL	20	2175.0000
7566	jones	manager	7839	1981-04-02	2975	NULL	20	2175.0000
7788	scott	analyst	7566	1987-04-19	3000	NULL	20	2175.0000
7876	adams	clerk	7788	1987-05-23	1100	NULL	20	2175.0000
7902	ford	analyst	7566	1981-12-03	3000	NULL	20	2175.0000
7499	allen	salesman	7698	1981-02-20	1600	300	30	1566.6667
7521	ward	salesman	7698	1981-02-22	1250	500	30	1566.6667
7654	martin	salesman	7698	1981-09-28	1250	1400	30	1566.6667
7698	blake	manager	7839	1981-05-01	2850	NULL	30	1566.6667
7844	turner	salesman	7698	1981-09-08	1500	0	30	1566.6667
7900	james	clerk	7698	1981-12-03	950	NULL	30	1566.6667

其他函数 – 开窗函数

MYSQL8.0版本支持的11种开窗函数：

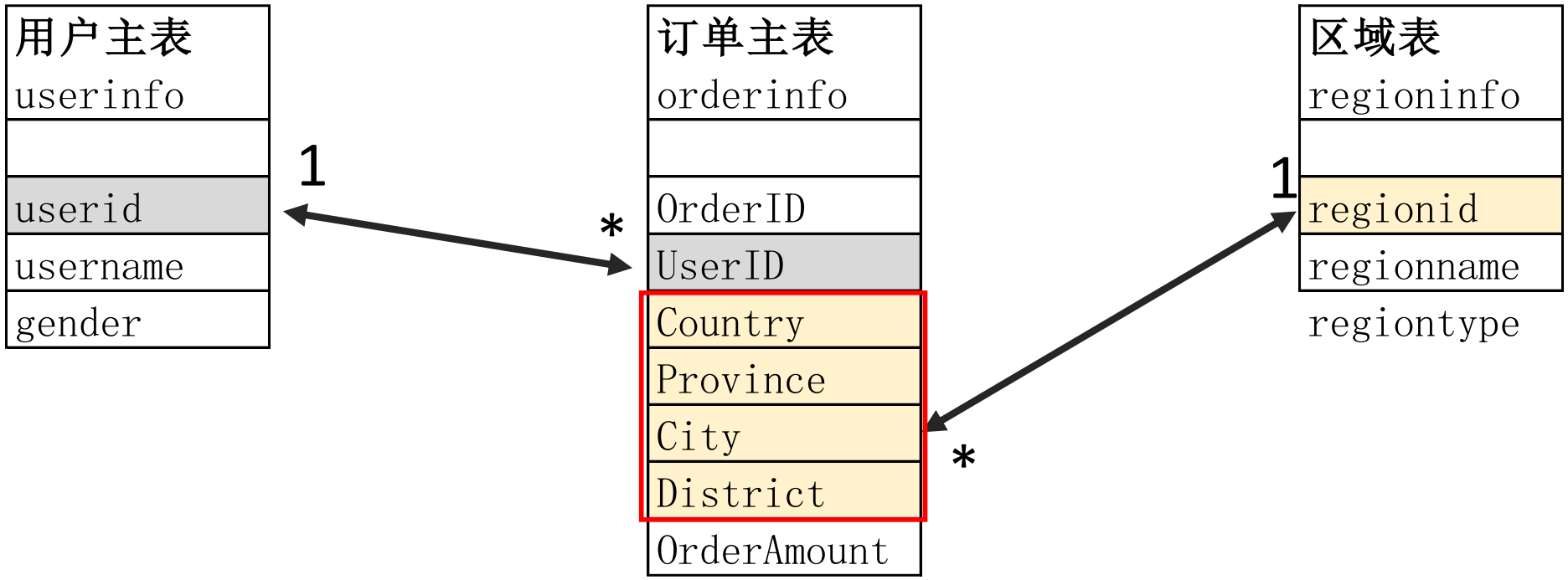
函数名	描述
CUME_DIST()	计算一组值中一个值的累积分布
DENSE_RANK()	根据该ORDER BY子句为分区中的每一行分配一个等级。它将相同的等级分配给具有相等值的行。如果两行或更多行具有相同的排名，排名值序列中将没有间隙
FIRST_VALUE()	返回相对于窗口框架第一行的指定表达式的值
LAG()	返回分区中当前行之前的第N行的值。如果不存在前一行，则返回NULL
LAST_VALUE()	返回相对于窗口框架中最后一行的指定表达式的值
LEAD()	返回分区中当前行之后的第N行的值。如果不存在后续行，贝腿回NULL
NTH_VALUE()	从窗口框架的第N行返回参数的值
NTILE()	将每个窗口分区的行分配到指定数量的排名组中
PERCENT_RANK()	计算分区或结果集中行的百分数等级
RANK()	与DENSE_RANK()函数相似，不同之处在于当两行或更多行具有相同的等级时，等级值序列中存在间隙
ROW_NUMBER()	为分区中的每一行分配一个顺序整数

查询练习



电商数据表ER图

E-R图也称实体-联系图(Entity Relationship Diagram)，用来描述现实世界的概念模型。



多表查询练习：电商数据查询练习

1.求出购买产品金额最多的前十名顾客

2.求出购买产品金额最多的前十名顾客的所在城市

3.求出购买力最强的前十个城市

4.求出购买力最强的前十个城市以及他们所在的省份