数据查询：函数、分组和排序
一、聚合函数
示例1：使用函数统计零售价的平均值，最大值和最小值
示例2：使用 count() 统计 product 表中的记录个数
示例3: 使用 count(distinct()) 统计 product 不重复值的个数
二、 ORDER BY 子句: 对查询结果排序
示例4： 找出商品名称中含有'理光'和'墨粉'的商品记录， 按零售价降序排列。（理光公司的墨粉产品）
示例5： 查询 product 表中的 product_id, product_name, product_place, price ，返回结果先按 product_place 降序排列，然后按 price 升序排序
三、 GROUP BY 子句: 对查询结果统计分组
示例6：按 Product_Place 分组，显示理光牌墨粉的名称，对应的记录数和平均价格
示例7：按 Product_Place 分组，显示各产地对应的产品记录个数
示例8：根据 product 表计算不同产地的商品单价最大值，按 Product_place 降序排列
示例9：根据 product 表计算不同产地的商品单价最大值，将单价最大值大于100元的产品的产地及单价最大值按 Product_ID 降序排列
示例10：查询 product 表中（类别，子类别）对应的最大商品价格，返回 sort_id, subsort_id 和对应的最大价格
示例11：根据 product 表计算不同 SubSort_ID 的商品单价平均值，列出前10条记录。
示例：查询各子类对应的 product_name ，用逗号连接起来
四、系统函数
示例12: 数学函数
示例13：字符串函数
示例14：日期和时间函数
示例15：条件判断
示例16: 查询 product 表中的 product_id, product_name, sort_id 和 subSort_id ， 把 sort_id 和 subSort_id 用“-”连接起来
示例17: 查询 product 表中的 product_id, product_name 和 product_date 字段值，如果 product_date 字段的月份大于6则返回下半年，否则返回上半年。
五、为表和字段取别名
示例18：给product表起一个别名tb_prod
示例19: 多表连接时，简化表名
示例20：给 Product_Place 字段起一个别名为 Place
练习
在 product 表中完成以下查询

数据查询：函数、分组和排序

```
use purchase;
```

一、聚合函数

- `MAX()` 取列中的最大值
- `MIN()` 取列中的最小值
- `AVG()` 计算列的平均值
- `COUNT()` 统计集合中元素的个数
- `DISTINCT 字段` 剔除重复值

示例1：使用函数统计零售价的平均值，最大值和最小值

```
SELECT AVG(price) FROM product;
SELECT MAX(price) FROM product;
SELECT MIN(price) FROM product;
```

示例2：使用 `count()` 统计 `product` 表中的记录个数

```
SELECT COUNT(*) FROM product;
SELECT COUNT(product_name) FROM product;
```

示例3: 使用 `count(distinct())` 统计 `product` 不重复值的个数

```
SELECT DISTINCT(sort_id) FROM product;
SELECT COUNT(DISTINCT(sort_id)) FROM product;
```

二、 `ORDER BY` 子句: 对查询结果排序

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]……
```

> 查询结果首先按字段1排序，如果字段1中有部分重复值，则按字段2进行排序，以此类推。

示例4： 找出商品名称中含有'理光'和'墨粉'的商品记录， 按零售价降序排列。（理光公司的墨粉产品）

```
SELECT *
FROM product
WHERE product_name LIKE '%理光%墨粉%' OR product_name LIKE '%墨粉%理光'
ORDER BY price DESC;
```

> 注意: 需要注意的是，在按照指定字段进行升序排列时，如果某条记录的字段值为 `null` ，则这条记录会在第一条显示， `null 小于任何值`。

示例5：查询 `product` 表中的 `product_id, product_name, product_place, price`，返回结果先按 `product_place` 降序排列，然后按 `price` 升序排序

```
SELECT product_id, product_name, product_place, price
FROM product
ORDER BY product_place DESC, price ASC;
```

- 字符串排序

```
SELECT product_id, product_name, product_place, price
FROM product
ORDER BY product_id;

-- 将字符串转换为数值
SELECT product_id, product_name, product_place, price
FROM product
ORDER BY CAST(product_id AS UNSIGNED) DESC;
```

> `CAST(value AS type)` 或 `CONVERT(value, type)` 可对数据类型进行转换，`type` 可以为：
>
> - `CHAR[(N)]` 字符型
> - `DATE` 日期型
> - `DATETIME` 日期和时间型
> - `TIME` 时间型
> - `DECIMAL` float型
> - `SIGNED` 有符号 int类型
> - `UNSIGNED` 无符号int类型

三、`GROUP BY` 子句: 对查询结果统计分组

语法：

```
SELECT 字段名1,字段名2,……
FROM 表名
WHERE 条件表达式
GROUP BY 字段名1，字段名2，……
HAVING 条件表达式
ORDER BY 排序字段;
```

> 根据(字段1,字段2,...)的值进行分组
> 先由 `WHERE` 子句进行初步筛选，然后进行 `GROUP BY` ，再通过 `HAVING` 子句对分组结果进行筛选
> 执行次序: `FROM -> WHERE -> GROUP BY -> HAVING -> ORDER BY -> SELECT`

示例6：按 `Product_Place` 分组，显示理光牌墨粉的名称，对应的记录数和平均价格

```sql
SELECT *
FROM product
WHERE product_name LIKE '%理光%墨粉%';
```

```sql
-- 错误写法
SELECT *
FROM product
WHERE product_name LIKE '%理光%墨粉%'
GROUP BY product_place;
```

```sql
SELECT product_place, COUNT(*), AVG(price)
FROM product
WHERE product_name LIKE '%理光%墨粉%'
GROUP BY product_place;
```

> 注意：为使查询具有现实意义，`select` 后的字段应为聚合函数或则 `group by` 中出现的字段。

示例7：按 `Product_Place` 分组，显示各产地对应的产品记录个数

```sql
SELECT product_place, COUNT(*)
FROM product
GROUP BY product_place;
```

示例8：根据 `product` 表计算不同产地的商品单价最大值，按 `Product_place` 降序排列

```sql
SELECT product_place, MAX(price)
FROM product
GROUP BY product_place
ORDER BY product_place DESC;
```

- `HAVING` 可以对结果集进行第二次过滤，即 `WHERE` 子句确定用于分组的数据集，`HAVING` 确定分组之后的哪些数据行可以保留下来。

示例9：根据 `product` 表计算不同产地的商品单价最大值，将单价最大值大于100元的产品的产地及单价最大值按 `Product_ID` 降序排列

```sql
SELECT product_place, MAX(price)
FROM product
GROUP BY product_place
HAVING MAX(price) > 100;
```

示例10：查询 `product` 表中（类别，子类别）对应的最大商品价格，返回 `sort_id, subsort_id` 和对应的最大价格

```sql
SELECT sort_id, subsort_id, MAX(price)
FROM product
GROUP BY sort_id, subsort_id;
```

示例11：根据 `product` 表计算不同 `SubSort_ID` 的商品单价平均值，列出前10条记录。

```
SELECT subsort_id, AVG(price)
FROM product
GROUP BY subsort_id;
```

```
SELECT subsort_id, AVG(price)
FROM product
GROUP BY subsort_id LIMIT 10;
```

`group_concat()` 函数：将某一分组中的某一字段对应的所有字符串连接起来，即返回分组中对应字段的所有值

```
GROUP_CONCAT(col1 [ORDER BY col2 [DESC|ASC]] [SEPARATOR ''])
```

示例：查询各子类对应的 `product_name` ，用逗号连接起来

```
SELECT subsort_id, group_concat(product_name)
FROM product
GROUP BY subsort_id;
```

四、系统函数

示例12: 数学函数

```
SELECT ABS(-1);
SELECT SQRT(4);
SELECT MOD(10, 3);
SELECT CEILING(9.3), FLOOR(9.3);
SELECT ROUND(9.32, 1);
SELECT TRUNCATE(9.321, 2);
SELECT SIGN(-8.2), SIGN(0), SIGN(6);

SELECT PI(),SIN(PI()),COS(PI()),TAN(0);
```

示例13：字符串函数

```
SELECT LENGTH('abcdef123');
SELECT CONCAT('背景', '--', '音乐');
SELECT LENGTH(TRIM(' aabdfe '));
SELECT LENGTH(LTRIM(' aabdfe '));
SELECT LENGTH(RTRIM(' aabdfe '));
SELECT REPLACE('背景音乐', '背景', '北京');
SELECT SUBSTRING('abcdef123', 1, 3);
SELECT REVERSE('abcdef123');
SELECT LOCATE('c','abcdef123');
```

示例14：日期和时间函数

```sql
SELECT NOW();
SELECT CURDATE();
SELECT CURRENT_DATE();
SELECT CURTIME();
SELECT CURRENT_TIME();
SELECT SYSDATE();
SELECT CURRENT_TIMESTAMP();
SELECT TIME_TO_SEC(CURTIME());
SELECT ADDDATE('2012-12-21', '7');
SELECT SUBDATE('2012/12/21', '7');
SELECT DATE_FORMAT(NOW(),'%m-%d-%y');
SELECT DATE_FORMAT(NOW(),'%b %d %Y %h:%i %p'); -- b为缩写月名
SELECT DATE_FORMAT(NOW(),'%m-%d-%Y'); -- Y 4位年份
SELECT DATE_FORMAT(NOW(),'%d %b %y'); -- y 2位年份
SELECT DATE_FORMAT(NOW(),'%d %b %Y %T:%f'); -- T时间, 24-小时(hh:mm:ss)
```

示例15：条件判断

```sql
SELECT IF(5>6, '对', '错'); -- 如果第1个参数为真，则取第2个参数，否则去取第3个参数值
SELECT IFNULL(null, '空值'), IFNULL(1, '空值'); -- 若第1个参数为空，取第2个参数的值

-- 让空值排末尾
SELECT * FROM product ORDER BY ifnull(price, -100);
```

示例16: 查询 `product` 表中的 `product_id, product_name, sort_id` 和 `subSort_id`， 把 `sort_id` 和 `subSort_id` 用“-”连接起来

```sql
SELECT CONCAT(product_id,product_name,sort_id,'-',subsort_id)
FROM product;
```

示例17: 查询 `product` 表中的 `product_id, product_name` 和 `product_date` 字段值，如果 `product_date` 字段的月份大于6则返回下半年，否则返回上半年。

```sql
SELECT product_id, product_date, product_name,
 IF(MONTH(product_date)>6, '下半年', '上半年') AS 半年
FROM product;
```

五、为表和字段取别名

为表起别名的语法格式: 表名 [AS] 别名;

示例18：给product表起一个别名tb_prod

```sql
SELECT tb_prod.product_name, product_place
FROM product AS tb_prod;
```

示例19: 多表连接时，简化表名

```
SELECT a.product_id, a.product_name, b.sort_name
FROM product a, sort b
WHERE a.sort_id = b.sort_id;
```

示例20：给 `Product_Place` 字段起一个别名为 `Place`

```
SELECT product_place AS place
FROM product;
```

练习

在 `product` 表中完成以下查询

(1)查找各品牌复印机的品牌名（命名为 `品牌` ）和最高零售价（命名为 `最高价` ）

> 假定商品名的前两个字符为品牌，按品牌分组可以用 `group by substring(Product_Name, 1, 2)`

(2) 查找 `Product_Place` 和该产地的产品数（命名为 `产品数量` ），显示产品数在100种以上的产地和产品数量。

(3) 根据 `product` 表计算不同 `SubSort_ID` 的商品单价平均值（命名为 `Avg_Price` ），列出前10条记录。

(4) 查询 `product` 表中的 `Product_ID, Product_Name` 和 `Product_Date` 字段值，如果 `Product_Date` 字段的月份大于6则返回 `下半年` ，否则返回 `上半年` ，并把 `if` 条件表达式命名为 `半年` 。

(5) 查找 `Product_ID, Product_Name,Product_Date` ，并标记 `Product_Date` 对应的季度，把计算季度的表达式命名为 `季度` 。

(6) 查找按产地和生产月份分组的零售价平均值，显示 `Product_Place` ,生产月份和零售价均值，并将生产月份命名为 `月份` ，零售价均值命名为"均价"。