第05章_排序与分页

1. 排序数据

1.1 排序规则

如果没有是使用排序规则，默认按照添加顺序进行排序

- 使用 ORDER BY 子句排序
  - **ASC（ascend）: 升序**
  - **DESC（descend）:降序**
- **ORDER BY 子句在SELECT语句的结尾**

1.2 单列排序

- **按照salary从高到低的顺序显示员工信息**

```
SELECT employee_id,last_name,salary
FROM employees
ORDER BY salary DESC;
+-------------+-------------+----------+
| employee_id | last_name   | salary   |
+-------------+-------------+----------+
|         100 | King        | 24000.00 |
|         101 | Kochhar     | 17000.00 |
|         132 | Olson       |  2100.00 |
+-------------+-------------+----------+
107 rows in set (0.00 sec)
```

- **按照salary从低到高的顺序显示员工信息**

默认就是降序

```
SELECT employee_id,last_name,salary
FROM employees
ORDER BY salary ASC;

SELECT employee_id,last_name,salary
FROM employees
ORDER BY salary; # 如果在ORDER BY 后没有显式指名排序的方式的话，则默认按照升序排列

+-------------+-------------+----------+
| employee_id | last_name   | salary   |
+-------------+-------------+----------+
```

```
|         132 | Olson       |  2100.00 |
|         128 | Markle      |  2200.00 |
|         183 | Geoni       |  2800.00 |
|         193 | Everett     |  3900.00 |
|         102 | De Haan     | 17000.00 |
|         100 | King        | 24000.00 |
+-------------+-------------+----------+
107 rows in set (0.00 sec)
```

使用列的别名，进行排序

```
SELECT employee_id,salary,salary * 12 annual_sal
FROM employees
ORDER BY annual_sal;
```

- **注意**

```
#列的别名只能在 ORDER BY 中使用，不能在WHERE中使用。
#如下操作报错!
SELECT employee_id,salary,salary * 12 annual_sal     【3】
FROM employees                                       【1】
WHERE annual_sal > 81600;                            【2】
```

- **强调格式：WHERE 需要声明在FROM后，ORDER BY之前**

```
SELECT employee_id,salary               【3】
FROM employees                          【1】
WHERE department_id IN (50,60,70)       【2】
ORDER BY department_id DESC;            【4】
```

1.3 多列排序

此处以二级排序为例：

- **显示员工信息，按照department_id的降序排列，salary的升序排列**

```
mysql> SELECT employee_id,salary,department_id
    -> FROM employees
    -> ORDER BY department_id DESC,salary ASC;
+-------------+----------+---------------+
| employee_id | salary   | department_id |
+-------------+----------+---------------+
|         206 |  8300.00 |           110 |
|         205 | 12000.00 |           110 |
|         181 |  3100.00 |            50 |
|         196 |  3100.00 |            50 |
|         201 | 13000.00 |            20 |
|         200 |  4400.00 |            10 |
|         178 |  7000.00 |          NULL |
+-------------+----------+---------------+
```

```
107 rows in set (0.00 sec)
```

- 可以使用不在SELECT列表中的列排序（先确定从那个表获取数据，再过滤一波数据，然后才是获取指定的字段，接着进行排序，分页.....）
- 在对多列进行排序的时候，首先排序的第一列必须有相同的列值，才会对第二列进行排序。如果第一列数据中所有值都是唯一的，将不再对第二列进行排序

2. 分页

2.1 背景

背景1：查询返回的记录太多了，查看起来很不方便，怎么样能够实现分页查询呢？

背景2：表里有 4 条数据，我们只想要显示第 2、3 条数据怎么办呢？

2.2 实现规则

- 分页原理

  所谓分页显示，就是将数据库中的结果集，一段一段显示出来需要的条件。

- **MySQL中使用 LIMIT 实现分页**
- 格式：

> LIMIT [位置偏移量,] 行数

第一个“位置偏移量”参数指示MySQL从哪一行开始显示，是一个可选参数，如果不指定“位置偏移量”，将会从表中的第一条记录开始

（第一条记录的位置偏移量是0，第二条记录的位置偏移量是1，以此类推）；

第二个参数“行数”指示返回的记录条数。

- 举例

```
--前10条记录：（第一页）
SELECT * FROM 表名 LIMIT 0,10;
或者
SELECT * FROM 表名 LIMIT 10;

--第11至20条记录：  （第二页）
SELECT * FROM 表名 LIMIT 10,10;

--第21至30条记录：  （第三页）
SELECT * FROM 表名 LIMIT 20,10;
```

- 分页显式公式：

> （当前页数-1）*每页条数，每页条数

```
SELECT * FROM table
LIMIT(PageNo - 1)*PageSize,PageSize;
```

- **注意：LIMIT 子句必须放在整个SELECT语句的最后！**
- 使用 LIMIT 的好处

约束返回结果的数量可以`减少数据表的网络传输量`，也可以`提升查询效率`。如果我们知道返回结果只有 1 条，就可以使用`LIMIT 1`，告诉 SELECT 语句只需要返回一条记录即可。这样的好处就是 SELECT 不需要扫描完整的表，只需要检索到一条符合条件的记录即可返回

WHERE ORDER BY LIMIT 声明顺序

注意：这不是执行顺序

- **如下**

```
SELECT employee_id,last_name,salary
FROM employees
WHERE salary > 6000
ORDER BY salary DESC
#limit 0,10;
LIMIT 10;
+-------------+-----------+----------+
| employee_id | last_name | salary   |
+-------------+-----------+----------+
|         100 | King      | 24000.00 |
|         101 | Kochhar   | 17000.00 |
|         102 | De Haan   | 17000.00 |
|         145 | Russell   | 14000.00 |
|         146 | Partners  | 13500.00 |
|         201 | Hartstein | 13000.00 |
|         108 | Greenberg | 12000.00 |
|         147 | Errazuriz | 12000.00 |
|         205 | Higgins   | 12000.00 |
|         168 | Ozer      | 11500.00 |
+-------------+-----------+----------+
10 rows in set (0.00 sec)
```

- **注意**

> 结构"LIMIT 0,条目数" 等价于 "LIMIT 条目数"

- **表里有107条数据，我们只想要显示第 32、33 条数据**

```
SELECT employee_id,last_name
FROM employees
LIMIT 31,2;#偏移量从0开始
```

- **查询员工表中工资最高的员工信息**

```
SELECT employee_id,last_name,salary
FROM employees
ORDER BY salary DESC
#limit 0,1
LIMIT 1;
```

MySQL8.0新特性：LIMIT ... OFFSET ...

- **表里有107条数据，我们只想要显示第 32、33 条数据**

```
SELECT employee_id,last_name
FROM employees
LIMIT 2 OFFSET 31;
```

注意：

后面的是偏移量，前面的是取多少条记录

> MySQL 8.0中可以使用“LIMIT 2 OFFSET 31”，意思是获取从第5条记录开始后面的3条记录，和“LIMIT 4,3;”返回的结果相同

2.3 拓展（了解）

在不同的 DBMS 中使用的关键字可能不同。在 MySQL、PostgreSQL、MariaDB 和 SQLite 中使用 LIMIT 关键字，而且需要放到 SELECT 语句的最后面。

- 如果是 SQL Server 和 Access，需要使用 `TOP` 关键字，比如：

```
SELECT TOP 5 name, hp_max FROM heros ORDER BY hp_max DESC
```

- 如果是 DB2，使用 `FETCH FIRST 5 ROWS ONLY` 这样的关键字：

```
SELECT name, hp_max FROM heros ORDER BY hp_max DESC FETCH FIRST 5 ROWS ONLY
```

- 如果是 Oracle，你需要基于 `ROWNUM` 来统计行数：

```
SELECT rownum,last_name,salary FROM employees WHERE rownum < 5 ORDER BY salary
DESC;
```

需要说明的是，这条语句是先取出来前 5 条数据行，然后再按照 hp_max 从高到低的顺序进行排序。但这样产生的结果和上述方法的并不一样。我会在后面讲到子查询，你可以使用

```
SELECT rownum, last_name,salary
FROM (
    SELECT last_name,salary
    FROM employees
    ORDER BY salary DESC)
WHERE rownum < 10;
```

得到与上述方法一致的结果

方言 LIMIT

1· LIMIT 可以使用在MySQL、PGSQL、MariaDB、SQLite 等数据库中使用，表示分页。
2· 不能使用在SQL Server、DB2、Oracle!

课后练习

- **查询员工的姓名和部门号和年薪，按年薪降序,按姓名升序显示**

```
SELECT last_name,department_id,salary * 12 annual_salary【2】
FROM employees  【1】
ORDER BY annual_salary DESC,last_name ASC;【3】

+-------------+---------------+---------------+
| last_name   | department_id | annual_salary |
+-------------+---------------+---------------+
| King        |            90 |     288000.00 |
| De Haan     |            90 |     204000.00 |
| Partners    |            80 |     162000.00 |
| Hartstein   |            20 |     156000.00 |
| Gee         |            50 |      28800.00 |
| Landry      |            50 |      28800.00 |
| Markle      |            50 |      26400.00 |
| Philtanker  |            50 |      26400.00 |
| Olson       |            50 |      25200.00 |
+-------------+---------------+---------------+
107 rows in set (0.00 sec)
```

- **选择工资不在 8000 到 17000 的员工的姓名和工资，按工资降序，显示第21到40位置的数据**

```
SELECT last_name,salary#【3】
FROM employees#【1】
WHERE salary NOT BETWEEN 8000 AND 17000#【2】
ORDER BY salary DESC#【4】
LIMIT 20,20;【#5】
```

```
+-----------+---------+
| last_name | salary  |
+-----------+---------+
| Ernst     | 6000.00 |
| Fay       | 6000.00 |
| Mourgos   | 5800.00 |
| Austin    | 4800.00 |
| Pataballa | 4800.00 |
| Whalen    | 4400.00 |
| Mallin    | 3300.00 |
| Stiles    | 3200.00 |
| Nayer     | 3200.00 |
+-----------+---------+
20 rows in set (0.00 sec)
```

- **查询邮箱中包含 e 的员工信息，并先按邮箱的字节数降序，再按部门号升序**

```
SELECT employee_id,last_name,email,department_id
FROM employees
#where email like '%e%'
WHERE email REGEXP 'e'
ORDER BY LENGTH(email) DESC,department_id;
+-------------+------------+----------+---------------+
| employee_id | last_name  | email    | department_id |
+-------------+------------+----------+---------------+
|         201 | Hartstein  | MHARTSTE |            20 |
|         114 | Raphaely   | DRAPHEAL |            30 |
|         119 | Colmenares | KCOLMENA |            30 |
|         151 | Bernstein  | DBERNSTE |            80 |
|         150 | Tucker     | PTUCKER  |            80 |
|         161 | Sewall     | SSEWALL  |            80 |
|         158 | McEwen     | AMCEWEN  |            80 |
|         145 | Russell    | JRUSSEL  |            80 |
|         110 | Chen       | JCHEN    |           100 |
|         135 | Gee        | KGEE     |            50 |
|         139 | Seo        | JSEO     |            50 |
|         165 | Lee        | DLEE     |            80 |
+-------------+------------+----------+---------------+
47 rows in set (0.00 sec)
```