# 实验四：数据高级查询

## 1. 实验环境

- 华为 ECS+openGauss 数据库服务器平台
- 前提：openGauss 数据库服务器正常运行
- 已建立带样例数据的 SALES 数据库

## 2. 实验目的

- 熟练掌握设计正确的 SQL 查询语句以实现数据高级查询的方法
- 熟练掌握 openGauss 连接查询、子查询和集合查询的语法结构及使用方法
  - （内）连接、（全）外连接、左外连接、右外连接
  - 子查询（嵌套查询）
  - 不相关子查询与相关子查询
  - EXISTS/NOT EXISTS
  - ANY
  - ALL
  - 集合运算：UNION、INSERSECT、MINUS/EXCEPT
- 理解不相关子查询与相关子查询的不同，掌握构造相应 SQL 语句的方法
- 熟练掌握基于派生表的查询方法

建议：对同一查询要求尽量使用不同的查询语句实现。如，所有带 IN 谓词、比较运算符、ANY 或 ALL 谓词的子查询都能用带 EXISTS 谓词的子查询等价替换。

## 3. 实验要求

4. 设计正确的 SQL 查询语句并测试其结果是否满足查询要求。

- 完成实验内容并提交实验报告到 FTP 上的相应文件夹“实验四”。
- 实验报告提交截至日期：2022 年 4 月 21 日星期四。

## 5. 实验内容与步骤

### 4.1 实验内容

（1）openGuass 的连接查询语法：

```
SELECT … FROM table1 [INNER] JOIN table2 ON table1.column= table2.column;

SELECT … FROM table1 FULL [OUTER] JOIN table2 ON table1.column= table2.column;

SELECT … FROM table1 RIGHT [OUTER] JOIN table2 ON table1.column= table2.column;

SELECT … FROM table1 LEFT [OUTER] JOIN table2 ON table1.column= table2.column;
```

创建两张表 palette_a 和 palette_b（结构相同，但表名不同，color 为颜色）

| | |
|---|---|
| CREATE TABLE palette_a<br>（id INT PRIMARY KEY,<br>color VARCHAR2 (100) NOT NULL); | CREATE TABLE palette_b<br>（ id INT PRIMARY KEY,<br>color VARCHAR2 (100) NOT NULL); |

（2）为表 palette_a 添加样例数据：{(1, 'Red'), (2, 'Green'), (3, 'Blue'), (4, 'Purple')}

（3）为表 palette_b 添加样例数据：{(1, 'Green'), (2, 'Red'), (3, 'Cyan'), (4, 'Brown')}

（4）查询两张表中相同颜色的所有信息

（5）查询 palette_a 表中颜色不出现在 palette_b 表中的两张表的 id 和颜色（用左外连接）

（6）查询 palette_b 表中颜色不出现在 palette_a 表中的两张表的 id 和颜色（用右外连接）

（7）查询（5）或（6）两种情况的信息（用（全）外连接）

- 子查询（Subquery）

（8）查询产品表 products 中的 product_id, product_name, list_price 信息，要求产品定价 list_price 大于其平均定价 list_price

（9）查询没有一个订单的顾客姓名（实现要求：NOT IN（必须）+其它查询方法（如果找到））

- 相关子查询（correlated subquery）

（10）查询产品表 products 中最便宜产品的 product_id, product_name, list_price。

（11）查询产品表 products 中产品的 product_id, product_name, list_price，要求产品定价 list_price 大于其同类产品（可由 category_id 表达）的平均定价

实现要求：相关子查询（必须）+基于派生表的查询（如果找到）

- EXISTS 的使用

（12）查询有订单 order 的所有顾客 customer 姓名（查询涉及 customers 表和 orders 表）

实现要求：使用 EXISTS（必须）+其它查询方法（如果找到）

- EXISTS 与 IN 的不同

（13）执行以下两条语句，观察有何不同，能否得出某些初步结论?

```
SELECT * FROM customers WHERE customer_id IN (NULL);

SELECT * FROM customers WHERE EXISTS (SELECT NULL FROM dual);
```

- NOT EXISTS 的使用

（14）找出所有没有订单的顾客姓名（查询涉及 customers 表和 orders 表）

实现要求：使用 NOT EXISTS（必须）+其它查询方法（如果找到）

- ANY 的使用

（15）查询产品表 products 中的产品名 product_name 和定价 list_price，要求其定价高于产品种类 1 中的任何产品定价

实现要求：ANY（必须）+其它查询方法（如果找到）

- ALL 的使用

（16）查询产品表 products 中的产品名 product_name 和定价 list_price，要求其定价高于产品种类 1 中的所有定价

（17）查询产品表 products 中的产品名 product_name 和定价 list_price，要求其定价低于产品种类的所有平均定价

实现要求：ALL（必须）+其它查询方法（如果找到）

- UNION 的使用

（18）查询 contacts 表和 employees 表中的所有 last_name，并以 last_name 升序显示

实现要求：去重+UNION（必须）+其它查询方法（如果找到）

（19）查询 contacts 表和 employees 表中的所有 last_name，并以 last_name 升序显示

实现要求：保留重复+UNION ALL（必须）+其它查询方法（如果找到）

- INTERSECT 的使用

（20）查询同时出现在 contacts 表和 employees 表中的所有 last_name

实现要求：INTERSECT（必须）+其它查询方法（如果找到）

- MINUS/EXCEPT 的使用

（21）查询在产品表 products 中而不在库存表 inventories 中的产品号 product_id

实现要求：MINUS/EXCEPT（必须）+其它查询方法（如果找到）

### 4.2 实验步骤

（1）以 root 用户登录到 ECS 服务器>以 omm 操作系统管理员身份登录数据库>使用 gsql 连接到数据库。操作步骤见《openGauss 开发者指南》中的 3.2.3 节。

（2）依次完成 4.1 中的实验内容。

## 6. 实验思考

- 什么类型的查询只能用子查询实现？试举例说明。
- 相关子查询与不相关子查询的区别？什么情形下使用相关子查询？如何将相关子查询转换成一般查询？（说明：一般查询指不一定必须使用子查询）

## 7. 参考资料

- openGauss 开发者指南.pdf 之 SELECT 语法（16.14.165 节 SELECT）
- 实验一教程
- openGauss 松鼠会： openGauss.org
- 墨天轮：https://www.modb.pro/tag/openGauss