《数据库概论》

实验二：高级SQL 实验报告

191220008 陈南曈

QQ：924690736

一、实验环境

操作系统：Windows 10 专业版，64 位操作系统，基于 x64 处理器

软件版本：MYSQL 8.0.26.0，Python 3.9，PyMySQL 1.0.2

二、实验过程

Q1、声明并调用存储过程，完成以下任务：

Q1.1

题目：存储过程功能：根据输入的商品名称，找到订购了该商品的客户编号、客户名称、订单编号、订货数量和订货金额，并按订货金额降序输出；调用该存储过程查询订购了“32M DRAM”商品的客户编号、客户名称、订单编号、订货数量和订货金额，并按订货金额降序输出。

SQL语句（声明存储过程）：

```
DROP PROCEDURE IF EXISTS productOrder;
delimiter //
CREATE PROCEDURE productOrder(IN product_name VARCHAR(40))
    BEGIN
        SELECT Customer.customerNo,
               Customer.customerName,
               OrderMaster.orderNo,
               OrderDetail.quantity,
               OrderDetail.quantity*OrderDetail.price AS orderSum
        FROM Customer, Product, OrderMaster, OrderDetail
        WHERE Product.productName=product_name AND
              Product.productNo=OrderDetail.productNo AND
              Customer.customerNo=OrderMaster.customerNo AND
              OrderMaster.orderNo=OrderDetail.orderNo
        ORDER BY orderSum DESC;
    END //
delimiter ;
```

SQL语句（调用存储过程）：

```
CALL productOrder("32M DRAM");
```

查询结果：

	customerNo	customerName	orderNo	quantity	orderSum
▶	C20050001	统一股份有限公司	200801090001	5	2500.00
	C20070002	世界技术开发公司	200803010001	8	1200.00
	C20080001	红度股份有限公司	200801090003	5	650.00
	C20050004	五一商厦	200803020001	2	200.00

Q1.2

题目：存储过程功能：根据输入的员工编号，查询比该员工雇佣日期早的同一部门的员工编号、姓名、性别、雇佣日期、所属部门；调用该存储过程查询比员工编号“E2008005”雇佣日期早的同一部门的员工编号、姓名、性别、雇佣日期、所属部门。

SQL语句（声明存储过程）：

```
DROP PROCEDURE IF EXISTS earlyEmployee;
delimiter //
CREATE PROCEDURE earlyEmployee(IN employee_no CHAR(8))
    BEGIN
        SELECT em1.employeeNo,
               em1.employeeName,
               em1.gender,
               em1.hireDate,
               em1.department
        FROM Employee em1, Employee em2
        WHERE em2.employeeNo=employee_no AND
              em1.hireDate<em2.hireDate AND
              em1.department=em2.department;
    END //
delimiter ;
```

SQL语句（调用存储过程）：

```
CALL earlyEmployee("E2008005");
```

查询结果：

	employeeNo	employeeName	gender	hireDate	department
▶	E2005001	喻自强	M	1990-02-06 00:00:00	财务科

Q2、声明并调用存储函数，完成以下任务：

Q2.1

题目：存储函数功能：根据输入的商品名称，返回该商品订购平均价；调用该存储函数查询商品基本信息表中所有商品名称及其订购平均价。

SQL语句（声明存储函数）：

```
SET GLOBAL log_bin_trust_function_creators = 1;
DROP FUNCTION IF EXISTS averagePrice;
delimiter //
CREATE FUNCTION averagePrice(product_name VARCHAR(40))
    RETURNS NUMERIC(7, 2)
    BEGIN
    DECLARE average_price NUMERIC(7, 2);
        SELECT AVG(OrderDetail.price) INTO average_price
        FROM Product, OrderDetail
        WHERE Product.productName=product_name AND
              Product.productNo=OrderDetail.productNo;
    RETURN average_price;
    END //
delimiter ;
```

SQL语句（调用存储函数）：

```
SELECT Product.productName, averagePrice(Product.productName) AS averagePrice
FROM Product;
```

查询结果：

	productName	averagePrice
▶	32M DRAM	220.00
	17寸显示器	350.00
	120GB硬盘	220.00
	3.5寸软驱	425.00
	键盘	375.00
	VGA显示卡	325.00
	网卡	265.00
	Pentium 100CPU	285.00
	1G DDR	293.20
	52倍速光驱	410.00
	计算机字典	283.33
	9600bits/s调...	340.00
	Pentium主板	220.00
	硕泰克SL—K8...	265.00
	龙基777FT纯...	244.00

Q2.2

题目：存储函数功能：根据输入的商品编号，统计该商品的销售总量；调用该存储函数查询销售总量大于4的商品编号、商品名称及销售数量。

SQL语句（声明存储函数）：

```
SET GLOBAL log_bin_trust_function_creators = 1;
DROP FUNCTION IF EXISTS quantitySum;
delimiter //
CREATE FUNCTION quantitySum(product_no VARCHAR(9))
    RETURNS INT
    BEGIN
    DECLARE quantity_sum INT;
        SELECT SUM(OrderDetail.quantity) INTO quantity_sum
        FROM OrderDetail
        WHERE OrderDetail.productNo=product_no;
    RETURN quantity_sum;
    END //
delimiter ;
```

SQL语句（调用存储函数）：

```
SELECT DISTINCT Product.productNo,
       Product.productName,
       quantitySum(Product.productNo) AS quantitySum
FROM Product, OrderDetail
WHERE quantitySum(Product.productNo)>4 AND
      Product.productNo=OrderDetail.productNo;
```

查询结果：

productNo	productName	quantitySum
P20050001	32M DRAM	20
P20050003	120GB硬盘	8
P20050004	3.5寸软驱	7
P20050005	键盘	6
P20060002	网卡	7
P20060003	Pentium100CPU	12
P20070001	1G DDR	15
P20070002	52倍速光驱	12
P20070003	计算机字典	11
P20070004	9600bits/s调制解调	5
P20080001	Pentium主板	7
P20080002	硕泰克SL—K8AN-R...	7

Q3、创建触发器，完成以下任务：

Q3.1

题目：当插入一项商品时，如果商品价格大于1000，则将其设置为1000。

SQL语句（创建触发器）：

```
DROP TRIGGER IF EXISTS insertProduct;
delimiter //
CREATE TRIGGER insertProduct BEFORE INSERT ON Product
FOR EACH ROW
BEGIN
    IF NEW.productPrice > 1000 THEN
        SET NEW.productPrice = 1000;
    END IF;
END //
delimiter ;
```

SQL语句（测试）：

```
# 插入价格不大于1000的商品，价格不变
INSERT INTO Product VALUES('P20090001','鼠标','设备',200.00);
# 插入价格大于1000的商品，价格变为1000
INSERT INTO Product VALUES('P20090002','耳机','设备',2000.00);
# 输出插入后的商品基本信息表
SELECT *
FROM Product;
```

查询结果：

productNo	productName	productClass	productPrice
P20050001	32M DRAM	内存	80.70
P20050002	17寸显示器	显示器	700.00
P20050003	120GB硬盘	存储器	300.00
P20050004	3.5寸软驱	设备	35.00
P20050005	键盘	设备	100.60
P20060001	VGA显示卡	显示器	1200.60
P20060002	网卡	设备	66.00
P20060003	Pentium 100CPU	处理器	200.00
P20070001	1G DDR	内存	256.00
P20070002	52倍速光驱	设备	200.00
P20070003	计算机字典	图书	100.00
P20070004	9600bits/s调...	设备	320.00
P20080001	Pentium主板	主板	890.00
P20080002	硕泰克SL—K8...	主板	1100.00
P20080003	龙基777FT纯...	显示器	900.00
P20090001	鼠标	设备	200.00
P20090002	耳机	设备	1000.00

可见插入的鼠标价格不变，耳机价格变为1000，说明触发器生效。

Q3.2

题目：当员工完成一个新的订单时，薪水增加5%；如果该员工是1992年前入职的，则再增加3%。

SQL语句（创建触发器）：

```
DROP TRIGGER IF EXISTS insertOrder;
delimiter //
CREATE TRIGGER insertOrder AFTER INSERT ON OrderMaster
FOR EACH ROW
BEGIN
    UPDATE Employee
    SET Employee.salary= Employee.salary*1.05
        WHERE Employee.employeeNo=NEW.employeeNo;
```

```
    UPDATE Employee
    SET Employee.salary = Employee.salary*1.03
        WHERE Employee.employeeNo=NEW.employeeNo AND
            Employee.hireDate < '19920101';
END //
delimiter ;
```

SQL语句（测试）：

```
# 输出插入订单前的员工表
SELECT Employee.employeeNo, Employee.salary AS oldSalary
FROM Employee
WHERE Employee.employeeNo='E2005003' OR Employee.employeeNo='E2005004';
# 插入订单，该业务员在1992年后入职
INSERT OrderMaster
VALUES('200812080001','C20080001','E2005003','20080717',0.00,'I000000011');
# 插入订单，该业务员在1992年前入职
INSERT OrderMaster
VALUES('200812080002','C20050002','E2005004','20080816',0.00,'I000000012');
# 输出插入订单后的员工表
SELECT Employee.employeeNo, Employee.salary AS newSalary
FROM Employee
WHERE Employee.employeeNo='E2005003' OR Employee.employeeNo='E2005004';
```

查询结果：

插入订单前：

employeeNo	oldSalary
E2005003	2600.00
E2005004	4100.00
NULL	NULL

插入订单后：

employeeNo	newSalary
E2005003	2730.00
E2005004	4434.15

可见入职时间在1992年后的员工工资增长了5%，入职时间在1992年前的员工工资还多加了3%，说明触发器生效。

Q4、使用高级程序设计语言访问SQL并执行如下的查询、插入、删除、更新：

Q4.1

题目：查询职工工资按从高到低排序的前20的职工编号、职工姓名和工资。

python语句：

```
# 打开数据库连接
db = pymysql.connect(host='localhost',
                     user='root',
                     passwd='123456',
                     database='OrderDB')
```

```
# 使用cursor()方法获取操作游标
cursor = db.cursor()
# SQL查询语句
sql = """
      SELECT Employee.employeeNo, Employee.employeeName, Employee.salary
      FROM Employee
      ORDER BY Employee.salary DESC
      LIMIT 0,21
      """
# 执行SQL语句
cursor.execute(sql)
# 获取所有记录列表
result = cursor.fetchall()
# 打印结果
for row in result:
    print(row)
# 关闭数据库连接
db.close()
```

查询结果：

```
('E2005001', '喻自强', Decimal('5800.00'))
('E2008005', '张小梅', Decimal('5000.00'))
('E2005004', '张露', Decimal('4100.00'))
('E2006001', '陈辉', Decimal('4000.00'))
('E2008004', '李虹冰', Decimal('3400.00'))
('E2008001', '陈诗杰', Decimal('3200.00'))
('E2008003', '黄梅莹', Decimal('3100.00'))
('E2008002', '张良', Decimal('2700.00'))
('E2005003', '张小娟', Decimal('2600.00'))
('E2006002', '韩梅', Decimal('2600.00'))
('E2006003', '刘风', Decimal('2500.00'))
('E2007001', '吴浮萍', Decimal('2500.00'))
('E2005002', '张小梅', Decimal('2400.00'))
('E2007002', '高代鹏', Decimal('2000.00'))
('E2005005', '张小东', Decimal('1800.00'))
```

Q4.2

题目：为客户表插入一条新的客户信息，客户编号“C20080002”，客户名称“泰康股份有限公司”，客户电话“010-5422685”，客户地址“天津市”，客户邮编“220501”。

python语句：

```
# 打开数据库连接
db = pymysql.connect(host='localhost',
                     user='root',
                     passwd='123456',
                     database='OrderDB')
# 使用cursor()方法获取操作游标
cursor = db.cursor()
# SQL插入语句
sql = """
      INSERT INTO Customer
      VALUES ('C20080002','泰康股份有限公司','010-5422685','天津市','220501')
      """
# 执行SQL语句
```

```
cursor.execute(sql)
# 提交到数据库执行
db.commit()
# SQL查询语句
sql = """
      SELECT *
      FROM Customer
      """
# 执行SQL语句
cursor.execute(sql)
# 获取所有记录列表
result = cursor.fetchall()
# 打印结果
for row in result:
    print(row)
# 关闭数据库连接
db.close()
```

查询结果：

```
('C20050001', '统一股份有限公司', '022-3566021', '天津市', '220012')
('C20050002', '兴隆股份有限公司', '022-3562452', '天津市', '220301')
('C20050003', '上海生物研究室', '010-2121000', '北京市', '108001')
('C20050004', '五一商厦', '021-4532187', '上海市', '210100')
('C20060001', '大地商城', '010-1165152', '北京市', '100803')
('C20060002', '联合股份有限公司', '021-4568451', '上海市', '210100')
('C20070001', '南昌市电脑研制中心', '0791-4412152', '南昌市', '330046')
('C20070002', '世界技术开发公司', '021-4564512', '上海市', '210230')
('C20070003', '万事达股份有限公司', '022-4533141', '天津市', '220400')
('C20080001', '红度股份有限公司', '010-5421585', '北京市', '100800')
('C20080002', '泰康股份有限公司', '010-5422685', '天津市', '220501')
```

Q4.3

题目：删除员工表中薪水高于5000的员工信息。

python语句：

```
# 打开数据库连接
db = pymysql.connect(host='localhost',
                     user='root',
                     passwd='123456',
                     database='OrderDB')
# 使用cursor()方法获取操作游标
cursor = db.cursor()
# SQL删除语句
sql = """
      DELETE
      FROM Employee
      WHERE Employee.salary>5000
      """
# 执行SQL语句
cursor.execute(sql)
# 提交到数据库执行
db.commit()
# SQL查询语句
```

```
sql = """
      SELECT Employee.employeeNo, Employee.employeeName, Employee.salary
      FROM Employee
      """
cursor.execute(sql)
# 获取所有记录列表
result = cursor.fetchall()
# 打印结果
for row in result:
    print(row)
# 关闭数据库连接
db.close()
```

查询结果：

```
('E2005002', '张小梅', Decimal('2400.00'))
('E2005003', '张小娟', Decimal('2600.00'))
('E2005004', '张露', Decimal('4100.00'))
('E2005005', '张小东', Decimal('1800.00'))
('E2006001', '陈辉', Decimal('4000.00'))
('E2006002', '韩梅', Decimal('2600.00'))
('E2006003', '刘风', Decimal('2500.00'))
('E2007001', '吴浮萍', Decimal('2500.00'))
('E2007002', '高代鹏', Decimal('2000.00'))
('E2008001', '陈诗杰', Decimal('3200.00'))
('E2008002', '张良', Decimal('2700.00'))
('E2008003', '黄梅莹', Decimal('3100.00'))
('E2008004', '李虹冰', Decimal('3400.00'))
('E2008005', '张小梅', Decimal('5000.00'))
```

Q4.4

题目：更新商品基本信息表中价格超过1000的商品价格变为原来的50%。

python语句：

```
# 打开数据库连接
db = pymysql.connect(host='localhost',
                     user='root',
                     passwd='123456',
                     database='OrderDB')
# 使用cursor()方法获取操作游标
cursor = db.cursor()
# SQL更新语句
sql = """
      UPDATE Product
      SET productPrice=productPrice*0.5
      WHERE productPrice>1000
      """
# 执行SQL语句
cursor.execute(sql)
# 提交到数据库执行
db.commit()
# SQL查询语句
sql = """
      SELECT *
      FROM Product
```

```
    """
# 执行SQL语句
cursor.execute(sql)
# 获取所有记录列表
result = cursor.fetchall()
# 打印结果
for row in result:
    print(row)
# 关闭数据库连接
db.close()
```

查询结果：

```
('P20050001', '32M DRAM', '内存', Decimal('80.70'))
('P20050002', '17寸显示器', '显示器', Decimal('700.00'))
('P20050003', '120GB硬盘', '存储器', Decimal('300.00'))
('P20050004', '3.5寸软驱', '设备', Decimal('35.00'))
('P20050005', '键盘', '设备', Decimal('100.60'))
('P20060001', 'VGA显示卡', '显示器', Decimal('600.30'))
('P20060002', '网卡', '设备', Decimal('66.00'))
('P20060003', 'Pentium100CPU', '处理器', Decimal('200.00'))
('P20070001', '1G DDR', '内存', Decimal('256.00'))
('P20070002', '52倍速光驱', '设备', Decimal('200.00'))
('P20070003', '计算机字典', '图书', Decimal('100.00'))
('P20070004', '9600bits/s调制解调', '设备', Decimal('320.00'))
('P20080001', 'Pentium主板', '主板', Decimal('890.00'))
('P20080002', '硕泰克SL-K8AN-RL主板', '主板', Decimal('550.00'))
('P20080003', '龙基777FT纯平显示器', '显示器', Decimal('900.00'))
```

Q5、使用高级程序设计语言中的动态SQL功能完成如下的任务（不保留4中对表信息的修改）：

Q5.1

题目：为“业务科”（作为外部输入参数）所有员工增加200的薪水。

python语句：

```
# 外部输出参数
department = '业务科'
# 打开数据库连接
db = pymysql.connect(host='localhost',
                     user='root',
                     passwd='123456',
                     database='OrderDB')
# 使用cursor()方法获取操作游标
cursor = db.cursor()
# SQL更新语句
sql = """
      UPDATE Employee
      SET salary=salary+200
      WHERE department=%s
      """
# 执行动态SQL语句，传入外部参数department
cursor.execute(sql, department)
# SQL查询语句
sql = """
```

```
        SELECT Employee.employeeNo, Employee.employeeName, Employee.salary
        FROM Employee
        """
# 执行SQL语句
cursor.execute(sql)
# 获取所有记录列表
result = cursor.fetchall()
# 打印结果
for row in result:
    print(row)
# 关闭数据库连接
db.close()
```

查询结果：

```
('E2005001', '喻自强', Decimal('5800.00'))
('E2005002', '张小梅', Decimal('2600.00'))
('E2005003', '张小娟', Decimal('2800.00'))
('E2005004', '张露', Decimal('4300.00'))
('E2005005', '张小东', Decimal('2000.00'))
('E2006001', '陈辉', Decimal('4000.00'))
('E2006002', '韩梅', Decimal('2800.00'))
('E2006003', '刘风', Decimal('2700.00'))
('E2007001', '吴浮萍', Decimal('2700.00'))
('E2007002', '高代鹏', Decimal('2000.00'))
('E2008001', '陈诗杰', Decimal('3200.00'))
('E2008002', '张良', Decimal('2900.00'))
('E2008003', '黄梅莹', Decimal('3300.00'))
('E2008004', '李虹冰', Decimal('3600.00'))
('E2008005', '张小梅', Decimal('5000.00'))
```

Q5.2

题目：查询客户表中的客户名称、客户地址及客户电话并输出。（注：结合游标的使用）

python语句：

```
# 打开数据库连接
db = pymysql.connect(host='localhost',
                     user='root',
                     passwd='123456',
                     database='OrderDB')
# 使用cursor()方法获取操作游标
cursor = db.cursor()
# SQL查询语句
sql = """
        SELECT Customer.customerName, Customer.address, Customer.telephone
        FROM Customer
        """
# 执行SQL语句
cursor.execute(sql)
# 获取所有记录列表
result = cursor.fetchall()
# 打印结果
for row in result:
    print(row)
# 关闭数据库连接
```

```
db.close()
```

查询结果：

```
('统一股份有限公司', '天津市', '022-3566021')
('兴隆股份有限公司', '天津市', '022-3562452')
('上海生物研究室', '北京市', '010-2121000')
('五一商厦', '上海市', '021-4532187')
('大地商城', '北京市', '010-1165152')
('联合股份有限公司', '上海市', '021-4568451')
('南昌市电脑研制中心', '南昌市', '0791-4412152')
('世界技术开发公司', '上海市', '021-4564512')
('万事达股份有限公司', '天津市', '022-4533141')
('红度股份有限公司', '北京市', '010-5421585')
```

三、实验中遇到的困难及解决办法

1、修改数据库密码时发现无法连接数据库

解决办法：在环境变量 path 中配置 mysql 路径。

2、运行 sql 语句时出现未知报错。

解决办法：添加如下代码。

```
SET GLOBAL log_bin_trust_function_creators = 1;
DROP FUNCTION IF EXISTS averagePrice;
delimiter //
CREATE FUNCTION averagePrice(...)
    ...
    BEGIN
    ...
    END //
delimiter ;
```

四、参考文献及致谢

1、课件PPT

2、Python3 MySQL 数据库连接 – PyMySQL 驱动 | 菜鸟教程 (runoob.com)