# 数据库系统实验 8 数据库设计与应用开发大作业

组员 1：杨陈泽　　16337271

组员 2：姚振杰　　16337285

组员 3：张金涛　　16337304

## 1. 实验目的：

掌握综合运用数据库原理、方法和技术进行数据库应用系统分析、设计和开发的能力。

## 2. 实验内容和要求：

为某个部门或单位开发一个数据库应用系统，具体内容包括：对某个部门或单位业务和数据进行调查，系统分析，系统设计，数据库设计，数据库创建和数据加载，数据库应用软件开发，系统测试，系统分析设计和开发文档撰写，软件、文档和数据库提交，数据库应用系统运行演示和大作业汇报。

能够针对某个部门或单位的应用需求，通过系统分析，从数据库数据和应用系统功能两方面进行综合设计，实现一个完整的数据库应用系统。同时培养团队合作精神。要求 5~6 位同学组成一个开发小组，每位同学承担不同角色（例如：项目管理员、DBA、系统分析员、系统设计员、系统开发员、系统测试员）。撰写系统设计和开发文档；提交系统文档、数据库应用软件和数据库。每个小组进行 60 分钟的报告和答辩，讲解设计方案，演示系统运行，汇报分工与合作情况。

## 3. 实验重点和难点：

实验重点：数据库设计，数据库应用软件开发。

实验难点：综合运用系统分析与设计方法，从数据和功能两方面协调设计一个完整的数据库应用系统。熟练掌握和运用一个主流数据库应用开发工具进行数据库应用软件开发。

## 4. 实验过程：

（1） 需求分析

本项目需要实现一个外卖点餐系统，主要有客户、商家和骑手三个模式。

客户的需求包括：①注册/登录账户；②查看商家；③选择菜式；④提交订单；⑤评价订单。

商家的需求包括：①注册/登录账户；②处理订单；③添加菜式；④添加地址。

骑手的需求包括：①注册/登录账户；②查看订单；③配送订单；④完成订单。

（2） 数据库设计

① 数据库概念结构设计

实体表：

- 用户 Customer：用户 ID(varchar(20))，密码（加密）(char(40))，用户名(nvarchar(20))，身份属性(int)。
- 商家 Supplier：商家 ID，密码（加密），商家名，身份属性，平均分

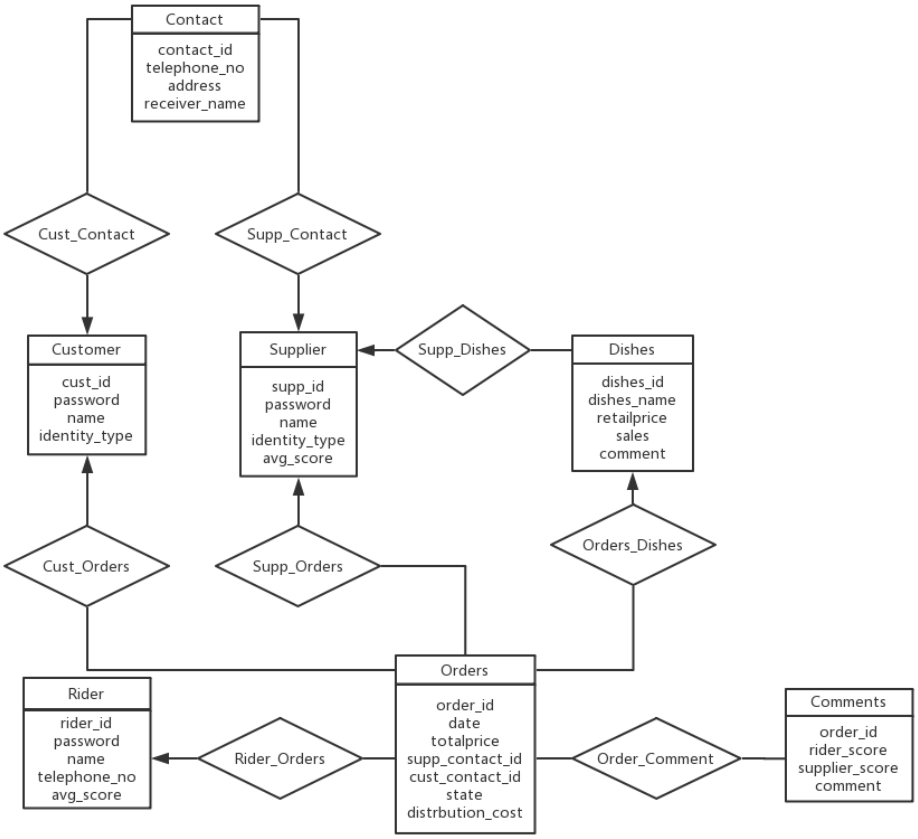
数(numeric(2, 1))。

- 订单 Orders：订单 ID(varchar(16))，时间(date)，金额(real)，商家联系表 ID(varchar(16))，用户联系表 ID(varchar(16))，订单状态(varchar(10))，配送费(real)。
- 商品 Dishes：商品 ID(varchar(16))，商品名字(nvarchar(20))，金额(real)，销售量(int)，comment(nvarchar(50))。
- 联系表 Contact：联系表 ID(varchar(16))，电话(char(11))，地址(nvarchar(50))，收件人名字(nvarchar(20))。
- 骑手 Rider：骑手 ID(varchar(16))，密码(加密)，骑手名字(nvarchar(20))，电话(char(11))，平均分数)。
- 评价 Comment：订单 ID（外码）(varchar(16))，骑手分数(int)，商家分数(int)，comment(nvarchar(50))。

联系表：

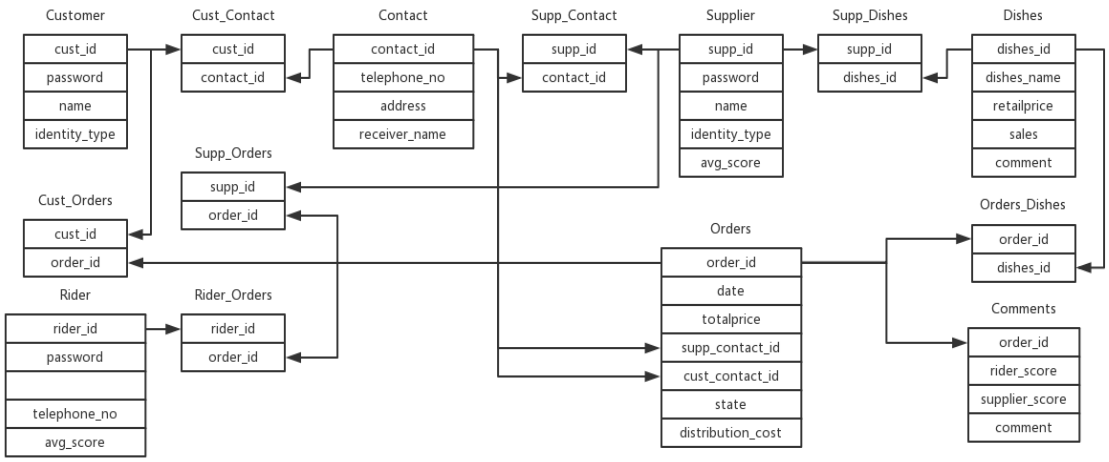
- 用户/商家-联系表 Cust_Contact/Supp_Contact（一对多）：用户/商家 ID，身份属性，联系表 ID。
- 商家-商品 Supp_Dishes（一对多）：商家 ID，商品 ID。
- 用户-订单 Cust_Orders（一对多）：用户 ID，订单 ID。
- 商店-订单 Supp_Orders（一对多）：商家 ID，订单 ID。
- 订单-商品 Orders_Dishes（一对多）：订单 ID，商品 ID。
- 骑手-订单 Rider_Orders（一对多）：骑手 ID，订单 ID。

实体-联系图（E-R 图）如下所示：




② 数据库逻辑结构设计

按照数据库中概念结构转化成逻辑结构的规则，每个实体转换成一个关系，实体之间的联系也转换成一个关系。因此，根据上述 ER 图可以设计出如下的数据库逻辑结构：



③ 数据库模式 SQL 语句生成；

下面是表的定义：

```
----# create Customer table
create table Customer(
    cust_id varchar(20) primary key,
    password char(40),
    name nvarchar(20),
    identity_type int default 1
);
----# create contact table
create table Contact(
    contact_id varchar(16) primary key,
    telephone_no char(11),
    address nvarchar(50),
    receiver_name nvarchar(20),
);
----# create contact table between customer and contact
create table Cust_Contact(
    cust_id varchar(20) references Customer(cust_id),
    contact_id varchar(16) references Contact(contact_id),
    primary key(cust_id, contact_id)
);
----# create supplier table
create table Supplier(
    supp_id varchar(20) primary key,
    password char(40),
    name nvarchar(20),
    identity_type int default 0,
    avg_score numeric(2, 1)
);
----# create contact table between supplier and contact
create table Supp_Contact(
    supp_id varchar(20) references Supplier(supp_id),
    contact_id varchar(16) references Contact(contact_id),
    primary key(supp_id, contact_id)
);
----# create dishes table
create table Dishes(
    dishes_id varchar(16) primary key,
    dishes_name nvarchar(20),
    retailprice real,
    sales int,
```

```
    comment nvarchar(50),
);
----# create contact table between supplier and dishes
create table Supp_Dishes(
    supp_id varchar(20) references Supplier(supp_id),
    dishes_id varchar(16) references Dishes(dishes_id),
    primary key(supp_id, dishes_id)
);
----# create orders table
create table Orders(
    order_id varchar(16) primary key,
    date DATE,
    totalprice real,
    supp_contact_id varchar(16) references Contact(contact_id),
    cust_contact_id varchar(16) references Contact(contact_id),
    state varchar(10) not null,
    distribution_cost real,
    check(state in ('to_do', 'to_deliver', 'delivering', 'done'))
);
----# create contact table between orders and dishes
create table Orders_Dishes(
    order_id varchar(16) references Orders(order_id),
    dishes_id varchar(16) references Dishes(dishes_id),
    primary key(order_id, dishes_id)
);
----# create comments table
create table Comments(
    order_id varchar(16) primary key references Orders(order_id),
    rider_score int,
    supplier_score int,
    comment nvarchar(50),
);
----# create contact table between orders and suppliers
create table Supp_Orders(
    supp_id varchar(20) references Supplier(supp_id),
    order_id varchar(16) references Orders(order_id),
    primary key(supp_id, order_id)
);
----# create contact table between customers and orders
create table Cust_Orders(
    cust_id varchar(20) references Customer(cust_id),
    order_id varchar(16) references Orders(order_id),
    primary key(cust_id, order_id)
);
```

```
----# create riders table
create table Rider(
    rider_id varchar(16) primary key,
    password char(40),
    name nvarchar(20),
    telephone_no char(11),
    avg_score numeric(2, 1),
);
----# create contact table between riders and orders
create table Rider_Orders(
    rider_id varchar(16) references Rider(rider_id),
    order_id varchar(16) references Orders(order_id),
    primary key(rider_id, order_id)
);
```

触发器的定义：

```
CREATE TRIGGER TRI_Comment_Score_INSERT
ON Comments
        AFTER INSERT
AS
DECLARE @new_order_id INTEGER,
                @supp_id INTEGER,
                @rider_id INTEGER,
                @supp_sum_score REAL,
                @rider_sum_score REAL,
                @supp_order_count INTEGER,
                @rider_order_count INTEGER;
BEGIN
        SELECT @new_order_id = order_id FROM inserted;
        SELECT @supp_id = supp_id FROM Supp_Orders WHERE order_id = @new_order_id;
        SELECT @rider_id = rider_id FROM Rider_Orders WHERE order_id = @new_order_id;
        SELECT @supp_sum_score = SUM(supplier_score),
                        @supp_order_count = COUNT(supplier_score)
        FROM Supp_Orders, Comments
        WHERE Supp_Orders.supp_id = @supp_id and Comments.order_id = Supp_Orders.order_id;
        SELECT @rider_sum_score = SUM(rider_score), @rider_order_count = COUNT(rider_score)
        FROM Rider_Orders, Comments
        WHERE Rider_Orders.rider_id = @rider_id and Comments.order_id = Rider_Orders.order_id;
        UPDATE Supplier
        SET avg_score = @supp_sum_score / @supp_order_count
        WHERE Supplier.supp_id = @supp_id;
```

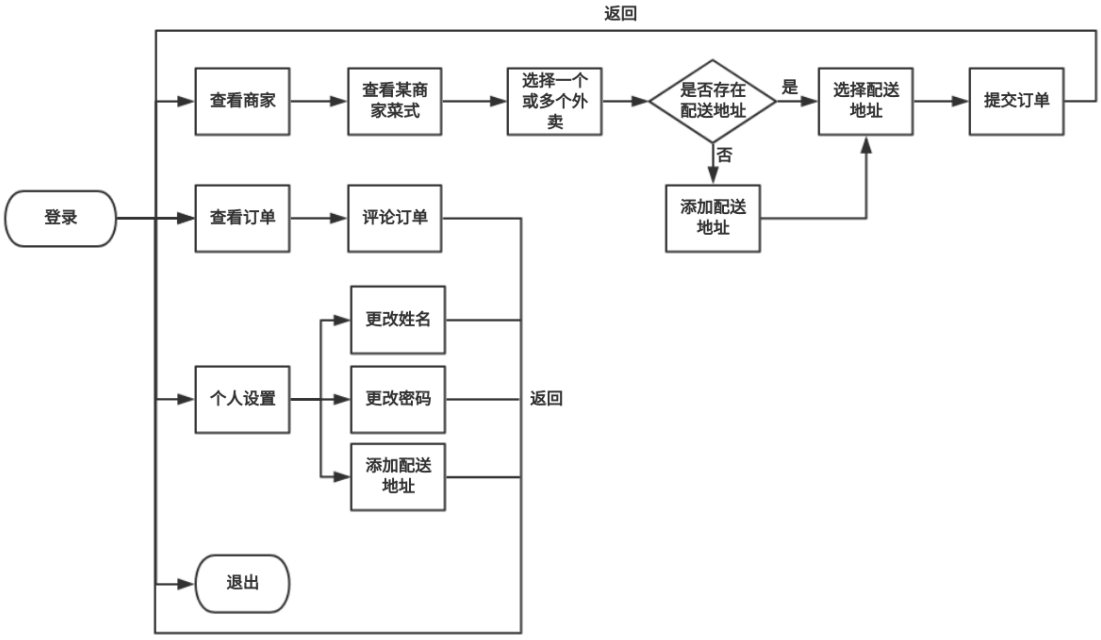
```
        UPDATE Rider
        SET avg_score = @rider_sum_score / @rider_order_count
        WHERE Rider.rider_id = @rider_id;
END

CREATE TRIGGER TRI_Orders_Dishes_INSERT
on Orders
        AFTER UPDATE
AS
DECLARE @dishes_totalprice REAL;
BEGIN
        SELECT @dishes_totalprice = SUM(retailprice)
        FROM inserted i, Orders_Dishes OD, Dishes D
        WHERE i.order_id = OD.order_id and OD.dishes_id = D.dishes_id;
        UPDATE Orders
        SET totalprice = @dishes_totalprice + O.distribution_cost
        FROM Orders O, inserted i
        WHERE O.order_id = i.order_id;
END
```

（3） 应用开发

① 客户模式：

客户程序的流程图如下：




客户端程序主要负责的重点在于为客户个人设置（注册、登录、更改名字、

更改密码、添加配送地址）、查看商家和商家的所有菜式以及进行下单、还有查看自己以往的订单然后进行订单的评价（或者查看订单的配送状态）。

- 注册账号

  用户在注册账号，需要填入自己的登录 ID，密码和姓名。登录 ID 在数据库中是属于用户表的主码，所以具有唯一性，而密码在存入数据库之前，会采用 sha1 单向加密算法进行加密，单向加密算法同时也保证了用户密码的安全性，不会被泄露出去。同时，在登录的时候，会将用户输入的密码进行 sha1 加密，再与数据库中所存的密文进行匹配，只有相匹配时才允许登录。

- 添加配送地址

  配送地址包含联系电话、配送的目的地以及联系人名字。而配送地址是为了用户在下单时，可以进行选择让商家进行配送的目的地，一个用户可以有多个配送地址，所以对于用户-配送地址而言，这是一个一对多的联系集。

- 查看商家

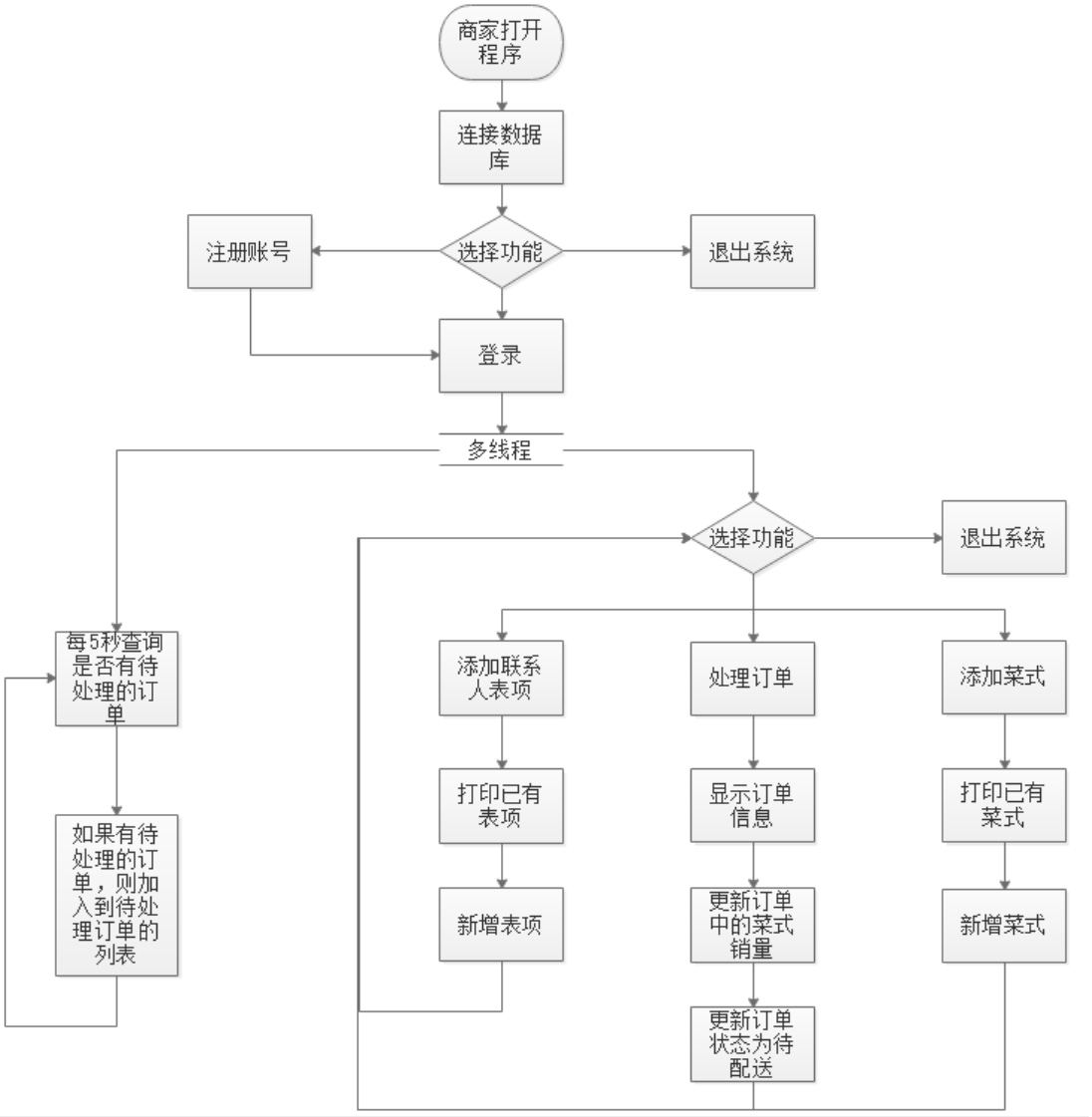
  用户在查看商家时，会显示所有注册的商家，以及用户给他们的平均评分。然后用户可以选择查看某个商家的所有菜式，包括该菜的名字、单价、销售量以及商家对它的描述。接着可以选择对哪个菜式进行添加到购物车，一次订单可以多个菜式，但是只能限定在一个商家的菜式里面。最后进行下单、选择配送地址，等待商家的配送。

- 查看订单

  用户可以查看自己的所有订单，包含订单的时间、外卖总价、配送费以及订单的状态（待接单、待配送、配送中、已完成）。然后用户可以对自己已完成的订单进行评价，包括对商家的打分评价、骑手的打分评价，还有进行文字评论。

② 商家模式：

商家程序的流程图如下：



商家程序的重点在于账户设置（包括注册账号，添加联系人表项等）、添加菜式，处理订单：

- 注册账号

  每个商家都有一个系统分配的账号（ID），这个账号从 0 开始，商家可以输入自己的密码和姓名，商家的密码经过 SHA1 加密后存储在数据库中，下次登录的时候从数据库中取出密文进行对比，正确的情况下才能成功登录；

- 添加联系人表项

  商家需要添加自己的联系人表项，包括联系人、电话、地址，商家可以有多个联系人表项，因此它是一个多值属性。商家的联系人表项信息将在客户提交订单的时候用到。

- 添加菜式

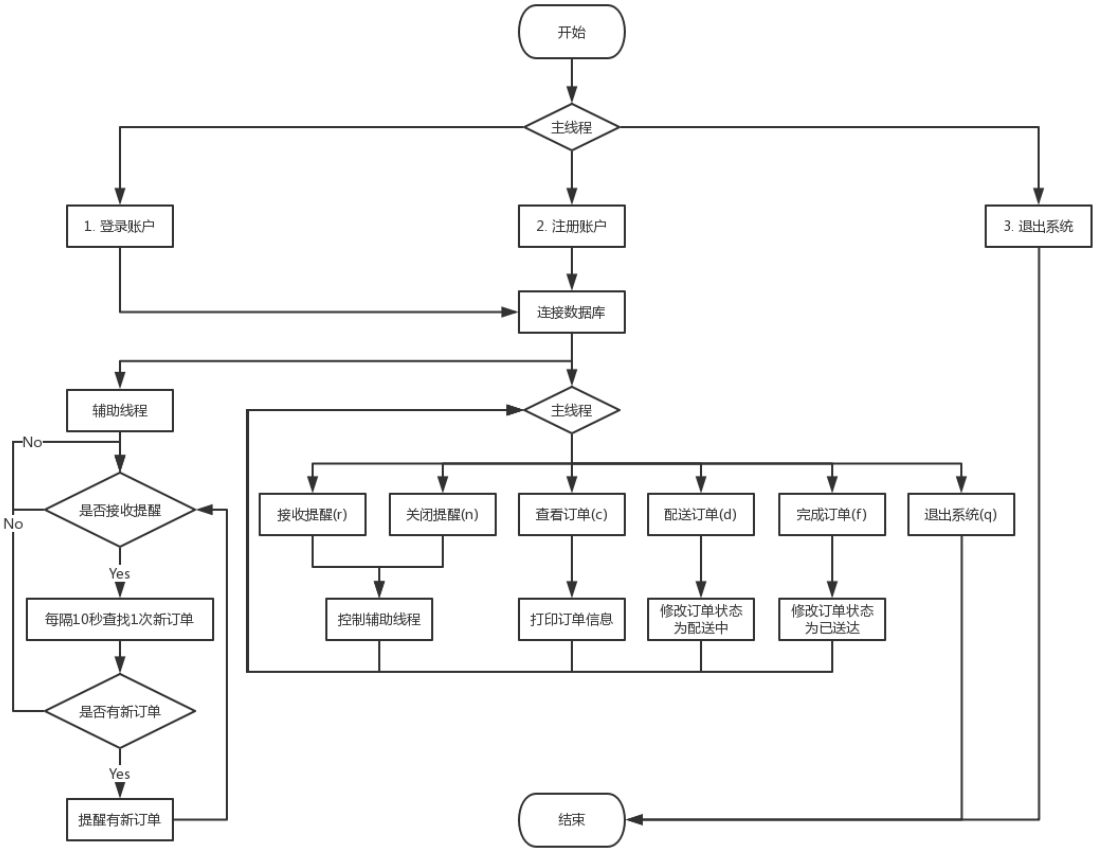
  商家需要添加自己商店的菜式，包括菜名、价格、备注等信息，客户进入商店后可以订购这些菜式；

- 处理订单

  商家需要处理客户订购的本商店的订单，根据订单中客户订购的菜式，更新本商店菜式的销售量，然后将订单的状态更新为待派送，等待骑手的派送。

③ 骑手模式：

骑手模式的流程图如下所示：




骑手模式实现的重点在于以下三个方面：

- 账户设置

  注册账户时的 ID 是由系统分配的，从 0 开始递增，以满足 Rider 的主键约束。密码、姓名、地址添加输入合法性判定，以免输入为空。

  注册过程中可以随时回退一步或者退出系统。

  登录账户时必须满足 ID 存在且密码匹配才能正常登录。

  账户密码先使用 SHA1 加密成 40 位的 16 进制数，再存储进数据库。

- 订单设置

  查看订单选项用于打印所有订单的客户地址和商家地址；

配送订单选项用于输入订单 ID 以接单，将订单状态（state）更改为配送中（delivering）；

完成订单选项用于输入订单 ID 以确定订单送达，将订单状态更改为已送达（done）。

- 提醒设置

  系统每隔 10 秒会查找一次数据库的订单信息，如果有新订单会打印提醒信息，以便骑手选择打印订单。

  骑手也可以自行选择接收提醒或关闭提醒。

## 5. 实验结果展示：

这里我们展示一下我们整个流程的实验结果：

首先，一个商家进行注册登录，然后添加自己的联系表以及两个菜式：

```
>> 处理订单请按 c， 添加菜式请按 d
>> 添加地址请按 a， 退出请按 q: a

>> 您目前还没有联系表!

>> 添加联系表请按 a; 退出请按 q:a
>> 请输入 电话， 想要退出请按 q.
>> 15627890035
>> 请输入 地址， 想要退出请按 q.
>> 贝岗村
>> 请输入 联系人， 想要退出请按 q.
>> 苏东坡
>> 插入联系表成功！
```

```
>> 处理订单请按 c， 添加菜式请按 d
>> 添加地址请按 a， 退出请按 q: d

>> 您的商店目前没有菜式!

>> 添加菜式请按：a; 退出请按 q :a
>> 请输入 菜名， 想要退出请按 q.
>> 东坡肉
>> 请输入 零售价， 想要退出请按 q.
>> 15
>> 请输入 备注， 想要退出请按 q.
>> 完美东坡肉
>> 菜式 东坡肉 已插入！
```

```
>> 处理订单请按 c， 添加菜式请按 d
>> 添加地址请按 a， 退出请按 q: d

>> 您当前的菜式如下:
>>> 菜名: 东坡肉 , 零售价: 15.0 , 销量: 0 , 备注: 完美东坡肉

>> 添加菜式请按：a; 退出请按 q :a
>> 请输入 菜名， 想要退出请按 q.
>> 红烧肉
>> 请输入 零售价， 想要退出请按 q.
>> 13
>> 请输入 备注， 想要退出请按 q.
>> 正宗红烧肉
>> 菜式 红烧肉 已插入！
```

然后一个用户登录并查看了所有商家（目前一共只有两家商家注册），选择了“东坡肉店”（上图展示的就是“东坡肉店”商家的操作），可以看到一共有 3 款菜式(“五花肉”的插入操作便没有展示，因为感觉重复了)，用户订购了“红烧肉”。

```
****************************************
************* 1：查看商家 *************
************* 2：查看订单 *************
************* 3：个人设置 *************
************* q：退出     *************
****************************************
请输入您的选项：1

****************************************
目前一共有2家商家注册：
1. "qq"，评分：4.0
2. "东坡肉店"，评分：5.0
****************************************

请选择需要查看商家的所有菜式（1 到 2，'q'：返回上一层）：
2
```

```
****************************************
"东坡肉店"商家一共有 3 款菜式：

****************************************
1. 名字：东坡肉，单价：15.0，销售量：0，描述：完美东坡肉
2. 名字：红烧肉，单价：13.0，销售量：0，描述：正宗红烧肉
3. 名字：五花肉，单价：10.0，销售量：0，描述：无

****************************************
请选择你想要订购的快餐（1 到 3）：
2
添加 "红烧肉" 成功！
```

他又继续添加了“五花肉”，然后进行结账，但由于他没有配送地址，所以需要先添加配送地址。

```
****************************************
************* 1：结账     *************
************* 2：继续添加 *************
************* q：退出     *************
****************************************
2

****************************************
1. 名字：东坡肉，单价：15.0，销售量：0，描述：完美东坡肉
2. 名字：红烧肉，单价：13.0，销售量：0，描述：正宗红烧肉
3. 名字：五花肉，单价：10.0，销售量：0，描述：无

****************************************
请选择你想要订购的快餐（1 到 3）：
3
添加 "五花肉" 成功！
```

```
****************************************
************* 1：结账     *************
************* 2：继续添加 *************
************* q：退出     *************
****************************************
1
对不起，你还没有配送地址，请先增加配送地址：
请输入联系电话：13729030065
请输入配送地址：至二
请输入联系人名字：要整洁
增加成功！
```

最后用户选择结账，一共选购了两样快餐，需要 25 元。

```
****************************************
1. 联系人名字：要整洁，联系电话：13729030065，配送地址：至二

****************************************
请选择配送地址（1 到 1）：1

****************************************
你一共选购了 2 样快餐：
1. 名字：红烧肉，单价：13.0
2. 名字：五花肉，单价：10.0
一共需要支付 25.0 (2元配送费) 元：

****************************************
************* 1：确认订单 *************
************* 2：取消订单 *************
************* q：退出     *************
****************************************
1

****************************************
支付成功！请耐心等待商家的配送。
```

用户返回来查看订单时，可以看到此时订单还是待接单的状态。

```
****************************************
************* 1：查看商家 *************
************* 2：查看订单 *************
************* 3：个人设置 *************
************* q：退出     *************
****************************************
请输入您的选项：2

****************************************
1. 时间：2018-12-24，联系人：要整洁，总价：25.0，配送费：2.0，状态：待接单。
```

“苏东坡肉店”商家发现有人下单，然后进行接单，准备好外卖以后，等待骑手接单进行配送。

```
>> 处理订单请按 c， 添加菜式请按 d
>> 添加地址请按 a， 退出请按 q: c
>> 处理订单 ('1', '1', '1', datetime.date(2018, 12, 24), 25.0, '2', '3', 'to_do', 2.0) 成功，订单待派送！
```

此时，用户可以查看到他的订单由原来的“待接单”状态变为“待配送”状态。

```
****************************************
************* 1：查看商家 *************
************* 2：查看订单 *************
************* 3：个人设置 *************
************* q：退出     *************
****************************************
请输入您的选项：2

****************************************
1. 时间：2018-12-24，联系人：要整洁，总价：25.0，配送费：2.0，状态：待配送。
```

接着，一个骑手进行注册登录以后，可以看到服务器提醒他有新订单可以进行配送，他可以看到该订单的商家地址和顾客的配送地址，从而决定是够需要接单。

```
>> 欢迎回来。
>> 请输入您的选择
>> 'c'：查看订单; 'd'：配送订单; 'f'：完成订单; 'r'：接收提醒; 'n'：关闭提醒; 'q'：退出系统
>> 您有新订单可以配送。
>> n
>> 请输入您的选择
>> 'c'：查看订单; 'd'：配送订单; 'f'：完成订单; 'r'：接收提醒; 'n'：关闭提醒; 'q'：退出系统
>> c
>> 订单号：1，商家地址：贝岗村，顾客地址：至二
>> 请输入您的选择
>> 'c'：查看订单; 'd'：配送订单; 'f'：完成订单; 'r'：接收提醒; 'n'：关闭提醒; 'q'：退出系统
>> d
>> 请输入订单号：
>> 1
>> 您已成功接单。
```

当骑手接单以后，用户可以查看到自己的订单由“待配送”的状态转变为“配送中”的状态。

```
****************************************
************* 1：查看商家 *************
************* 2：查看订单 *************
************* 3：个人设置 *************
************* q：退出     *************
****************************************
请输入您的选项：2

****************************************
1. 时间：2018-12-24，联系人：要整洁，总价：25.0，配送费：2.0，状态：配送中。
```

在骑手完成外卖的配送以后，他在软件系统中确定自己已完成该订单的配送。

```
>> 请输入您的选择
>> 'c'：查看订单；'d'：配送订单；'f'：完成订单；'r'：接收提醒；'n'：关闭提醒；'q'：退出系统
>> f
>> 您所配送的订单号：['1']
>> 请输入您已完成配送的订单号：
>> 1
>> 您已成功送达该订单，谢谢。
```

最后，用户在拿到外卖后，查看自己刚刚订单，订单已经由“代配送”状态转变为“已完成”状态，用户对该订单进行评价，包括对商家、骑手的打分。以及重新查看“东坡肉店”商家，可以看出商家的评分确实变为“4.0”分了，以及“红烧肉”和“五花肉”的销售量由原来的“0”变为“1”了。

```
****************************************
************* 1：查看商家 *************
************* 2：查看订单 *************
************* 3：个人设置 *************
************* q：退出     *************
****************************************
请输入您的选项：2

****************************************
1. 时间：2018-12-24，联系人：要整洁，总价：25.0，配送费：2.0，状态：已完成。

****************************************
********** 1：评价订单      ************
********** q：返回上一层    ************
请输入你的选项：1

****************************************
请输入你要进行评价的订单：1
请给商家打分：4
请给骑手打分：4
是否需要进行评论（是：直接输入评论内容；否：输入小写“n”并按回车）：很好吃
评价成功！
```

```
****************************************
************* 1：查看商家 *************
************* 2：查看订单 *************
************* 3：个人设置 *************
************* q：退出     *************
****************************************
请输入您的选项：1

****************************************
目前一共有2家商家注册：
1. "qq"，评分：4.0
2. "东坡肉店"，评分：4.0
****************************************

请选择需要查看商家的所有菜式（1 到 2，'q'：返回上一层）：
2

****************************************
"东坡肉店"商家一共有 3 款菜式：

****************************************
1. 名字：东坡肉，单价：15.0，销售量：0，描述：完美东坡肉
2. 名字：红烧肉，单价：13.0，销售量：1，描述：正宗红烧肉
3. 名字：五花肉，单价：10.0，销售量：1，描述：无
```

## 6. 实验总结：

16337285 杨陈泽

在本次项目中，我们小组三个人，先确定我们的实验方向：实现一个外卖软件。然后再一起讨论，对此进行需求分析，接着到实体集的讨论：需要什么实体，每个实体需要哪些属性等，然后到联系集的讨论。确定了这些以后，再进行数据库部分的工作，完成表的定义、触发器的定义。接着，确定了使用 python 高级语言以及 pyodbc 模块进行与 sql server 的连接。再分工，每个人负责实现一份软件代码。为了让用户在进行订单的评价以后，保持商家评分和骑手评分的一致性，我设计了两个触发器来更新他们的评分。

对于软件实现方面，我主要是负责用户软件的实现，一开始实现发现逻辑有点混乱，而为了维持数据库的事务性，我是在检查插入或者修改合法以后，再进行一个 commit()，要不然整个事务都会失效。比如：在用户下订单时，需要检测订单的菜式、用户的配送地址等等合法以后，再进行订单表项的插入。而且在如何存取用户、商家、骑手的密码问题上，我们经过认真地考虑了安全性，最后才决定采用 sha1 的单向加密加密算法，只支持加密不支持解密，而在数据库里面只存取密文，这样可以保证密码的绝对安全性。

16337285 姚振杰

本次项目综合应用了前几次实验中的数据库定义、数据库查询、数据库更新、完整性语言、触发器、存储过程等内容，并利用 python 的 pyodbc 模块实现了高级编程语言与 SQL Server 数据库的连接，较为全面地完成了外卖系统的数据库设计和应用开发。

在数据库设计的过程中，由于实体集的内部约束和外部参照，完整性语言的使用以及触发器的设计就显得尤其重要。例如订单（Orders）的状态（state）只能限定取值范围为待接单（to_do）、待配送（to_deliver）、配送中（delivering）和已送达(done)；订单加入菜式时，需要使用触发器(TRI_Orders_Dishes_INSERT)来更新订单的总价格。

至于应用开发，个人负责 rider 模块的实现，逻辑思路较为简单，痛过主线程完成骑手的注册登录、配送订单等一系列操作，辅助线程用于提醒骑手有新订单。基本的查询更新操作一般直接执行 select 或 update 语句，但接单操作使用了存储过程，以免多个骑手同时接下同一份订单。另外，如果在配送过程中退出程序，需要把配送中的订单 ID 存储进 json 文件中，以免内存清理导致配送订单信息的丢失。

16337304　张金涛

我们的数据库应用是一个外卖系统，涉及到了需求分析，概念设计、逻辑设计、数据库的实施以及应用系统的开发。比较难的地方是设计 E-R 图和开发应用系统。

首先是需求分析，根据给定的应用场景，我们可以较快的分析出需要存储哪些数据和实现哪些功能。接下来是概念设计，难点在于需要根据需求分析需要哪些实体集、联系集以及它们有哪些属性。之后是逻辑设计，将 E-R 图转换为表，这里需要注意属性类型、取值范围，还要考虑表之间的依赖关系。最后将表用 SQL 语言实现即可。为了保证数据库的约束完整性和实现特定的功能，我们也实现了一些触发器。虽然这些触发器的功能可以由应用程序实现，但是保证约束完整性

应该是数据库的职责，而不能期望由应用程序完成。

对于应用程序的开发，我负责了商家的部分。我们使用的是 python 编程语言，连接数据库的模块是 pyodbc。应用程序连接数据库需要安装好驱动，进行相应的配置。商家最主要的功能是添加菜式和处理顾客的订单。商家可以为自己店铺的菜式命名、定价、描述菜式的特点等。对于订单的处理，商家需要处理订单，将其状态从待处理更新为待派送。这里另外起了一个线程每个 5 秒查询数据库一次，查看是否有待处理的订单，然后商家可以从订单列表里获取待处理订单的信息。