

《数据库课程设计》

课程设计报告

|  |  |
| --- | --- |
| 设计题目： | 饭店点餐系统 |
| 院 系： | 信息工程学院 |
| 组 长： |  |
| 组 员： |  |
| 指导教师： |  |
| 时 间： | 2023.6.23 |

**目录**

[一、设计目标与具体分工 3](#_Toc138984060)

[1.1设计目标 3](#_Toc138984061)

[1.2具体分工 3](#_Toc138984062)

[二、需求分析 3](#_Toc138984063)

[2.1数据字典 3](#_Toc138984064)

[2.2数据流图 4](#_Toc138984065)

[2.3系统功能需求分析 6](#_Toc138984066)

[2.4其他需求分析 7](#_Toc138984067)

[三、总体设计 7](#_Toc138984068)

[3.1系统组成 7](#_Toc138984069)

[3.2开发工具 8](#_Toc138984070)

[3.3开发环境 8](#_Toc138984071)

[四、详细设计 8](#_Toc138984072)

[4.1概念结构设计 8](#_Toc138984073)

[4.2逻辑结构设计 9](#_Toc138984074)

[4.3 物理结构设计 12](#_Toc138984075)

[4.4数据库的实施与维护 13](#_Toc138984076)

[4.5视图设计 14](#_Toc138984077)

[4.6触发器设计 15](#_Toc138984078)

[五、成果展示 16](#_Toc138984079)

[5.1登录界面 16](#_Toc138984080)

[5.2管理员端界面 17](#_Toc138984081)

[5.3用户端设计 20](#_Toc138984082)

[5.4后端接口列表 23](#_Toc138984083)

[六、总结 24](#_Toc138984084)

## 设计目标与具体分工

### 1.1设计目标

本课设目标是完成一个食堂点餐的APP，分为用户界面和管理员界面。用户登录之后可以查看菜品列表、根据菜品列表点餐。点餐提交订单之后，可以查看订单的状态，订单完成之后，用户可以对该订单添加评论。管理员登录之后，可以修改菜品的信息，包括菜品的名称、种类、描述文字、图片；可以查看用户的订单，修改用户订单的状态。

完成整个APP的设计，包括前端界面设计、后端接口编写和数据库设计。

### 1.2具体分工

|  |  |  |  |
| --- | --- | --- | --- |
|  | 组长 | 组员一 | 组员二 |
| 姓名 |  |  |  |
| 学号 |  |  |  |
| 主要任务 | 主要负责前端界面搭建和部分后端接口设计 | 编写后端接口、设计数据库 | 编写后端接口、设计数据库 |

## 二、需求分析

### 2.1数据字典

|  |  |
| --- | --- |
| 数据结构名： | 菜品（Food） |
| 别名： | 菜品基本信息 |
| 描述： | 这是饭店点餐系统的主要数据结构之一，定义了饭店所有菜品的基本信息。 |
| 组成： | 菜品编号（Food\_id）、菜品名称（Food\_name）、菜品类型（Food\_typied）、菜品价格（Food\_price）  菜品描述（Food\_discrption）、菜品图片的Url（Food\_img\_url）、菜品销售热度（Food\_sale） |

|  |  |
| --- | --- |
| 数据结构名： | 订单（Order） |
| 别名： | 订单基本信息 |
| 描述： | 这是饭店点餐系统的主要数据结构之一，定义了顾客点餐的所有基本信息。 |
| 组成： | 订单编号（Order\_id）、订单价格（Order\_price）、订单时间（Order\_time）、订单状态（Order\_status）用户编号（User\_id）、对应的评论编号（Comment\_id） |

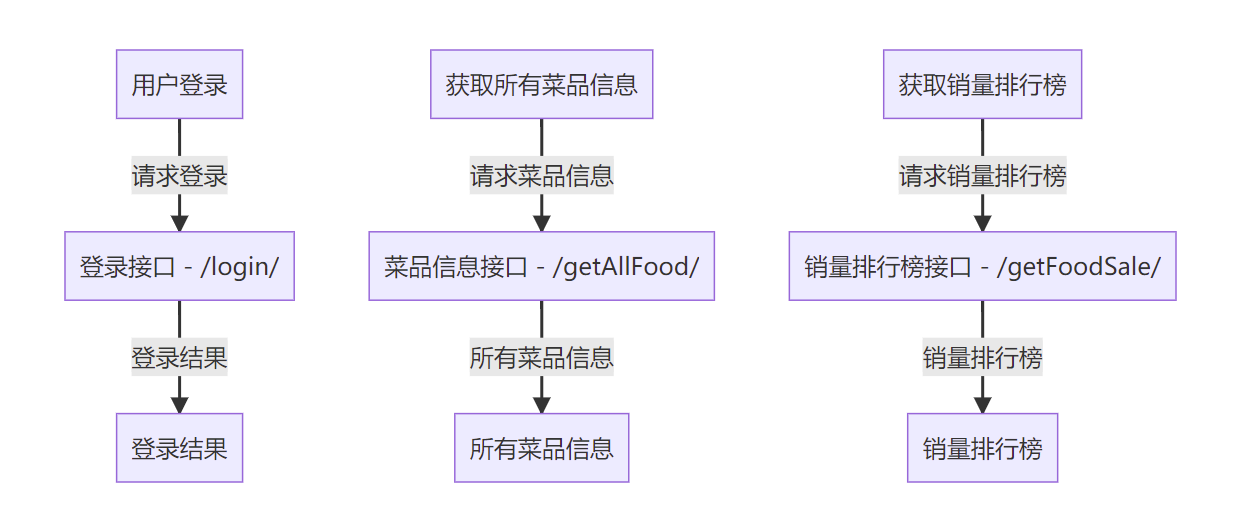
|  |  |
| --- | --- |
| 数据结构名： | 用户（User） |
| 别名： | 用户基本信息 |
| 描述： | 这是饭店点餐系统的主要数据结构之一，定义了使用该系统用户的的所有基本信息。 |
| 组成： | 用户编号（User \_id）、用户名称（User\_name）、用户密码（User\_password）、用户类别（User \_status） |

|  |  |
| --- | --- |
| 数据结构名： | 订单记录（Order\_content） |
| 别名： | 订单内容的基本信息 |
| 描述： | 这是饭店点餐系统的主要数据结构之一，定义了所有顾客的订单信息。 |
| 组成： | 订单编号（Order \_id）、菜品编号（Food\_id）、菜品数量（Food\_number） |

|  |  |
| --- | --- |
| 数据结构名： | 订单评论（User\_comment） |
| 别名： | 订单内容的基本信息 |
| 描述： | 这是饭店点餐系统的主要数据结构之一，定义了所有顾客的订单信息。 |
| 组成： | 评论编号（comment \_id）、星级（comment\_star）、评论的内容（comment\_content） |

### 2.2数据流图

用户在前端发送get和post请求，后端会发送数据到前端，具体的接口数据流图如下：



**图 2.2.1：前后端数据流交互图1**

图示

描述已自动生成

**图 2.2.2：前后端数据流交互图2**

图形用户界面

中度可信度描述已自动生成

**图 2.2.3：前后端数据流交互图3**

图形用户界面

描述已自动生成

**图 2.2.4：前后端数据流交互图4**

图片包含 图示

描述已自动生成

**图 2.2.5：前后端数据流交互图5**

### 2.3系统功能需求分析

实现一个适用于某个饭店的点餐系统（仅支持线下点餐），顾客可以通过该系统选择菜品以及数量进行下单，同时能够对加入购物车内的菜品实现增加、删除和修改操作；管理员通过登录系统对该系统的菜品做相应的调整，比如增加新菜品，修改菜品信息，删除菜品信息等；餐厅工作人员可以查看、处理和更新订单状态。

**顾客：**

1. 点餐
2. 增加菜品
3. 删除菜品
4. 修改菜品

**餐厅工作人员：**

1. 查看订单状态
2. 修改订单状态

**管理员：**

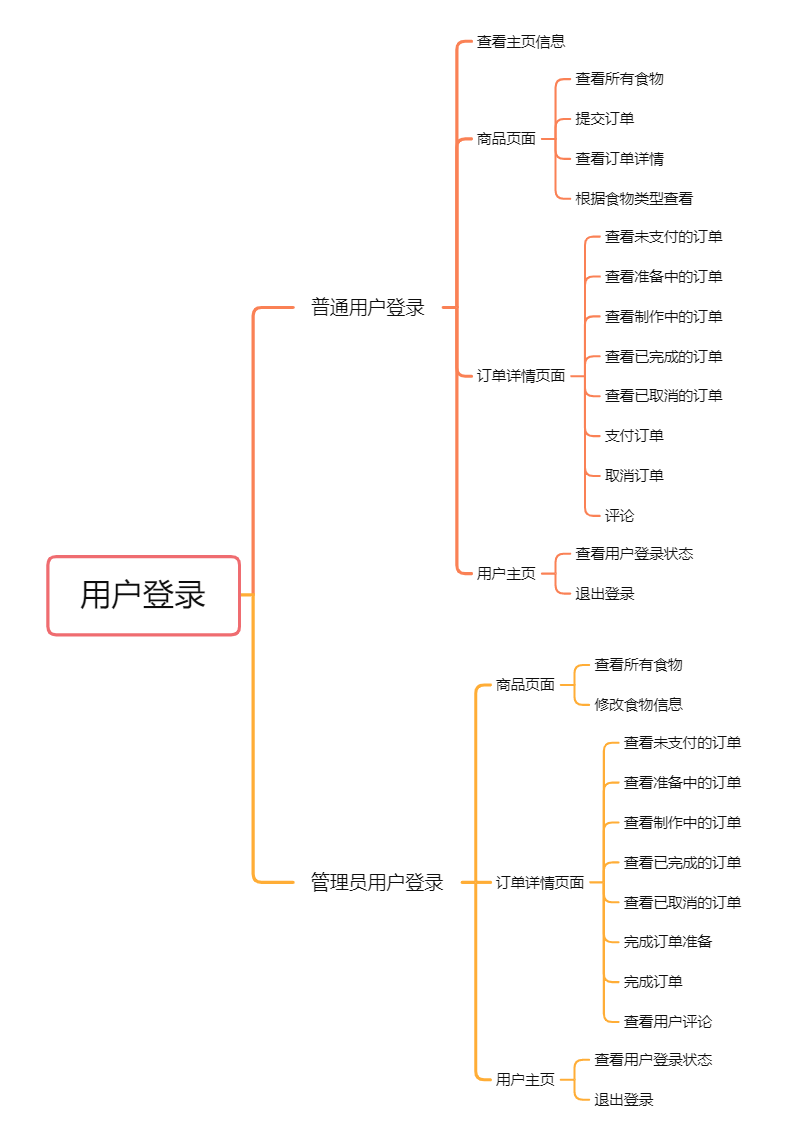
1. 增加菜品
2. 删除菜品
3. 修改菜品

### 2.4其他需求分析

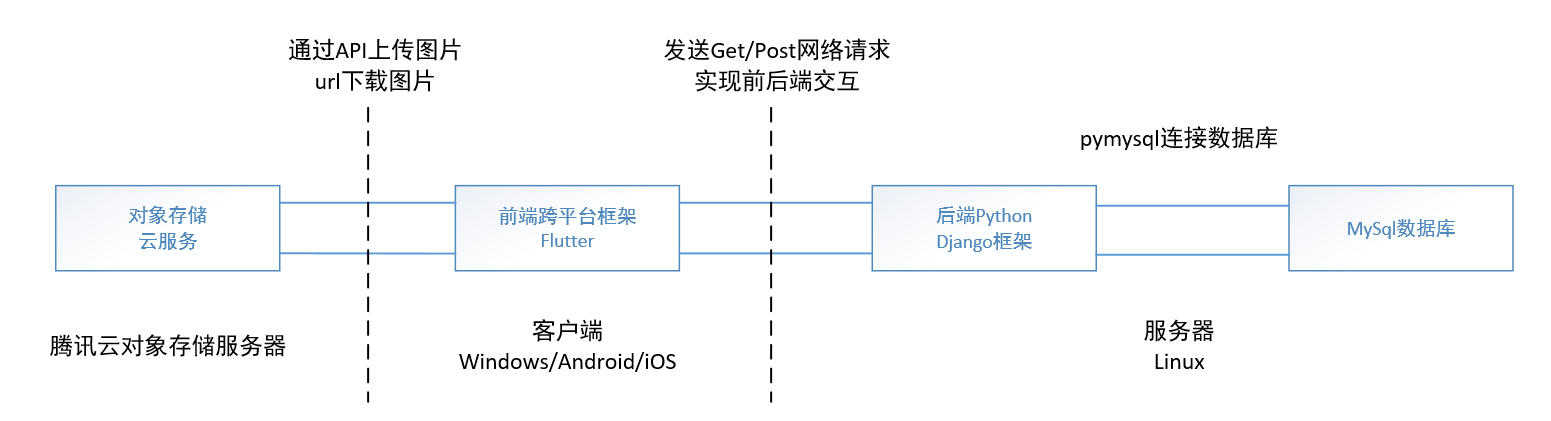
1. 操作错误时进行提示
2. 用户和管理员拥有不同的使用权限，保证数据的安全性
3. 建立触发器、存储过程，防止数据不一致
4. 建立折扣商品和热门商品的视图
5. 对数据库进行备份以便恢复

## 三、总体设计

### 3.1系统组成



**图 3.1.1：前端界面交互逻辑图**



**图 3.1.2：前后端交互逻辑图**

### 3.2开发工具

XShell、XFtp

PyCharm 2023.1.2

VisualStudio Code

### 3.3开发环境

Mysql 5.7.39, for Linux (x86\_64) (服务器)

MySQL 5.7.24 Community Server (本地)

Python 3.9

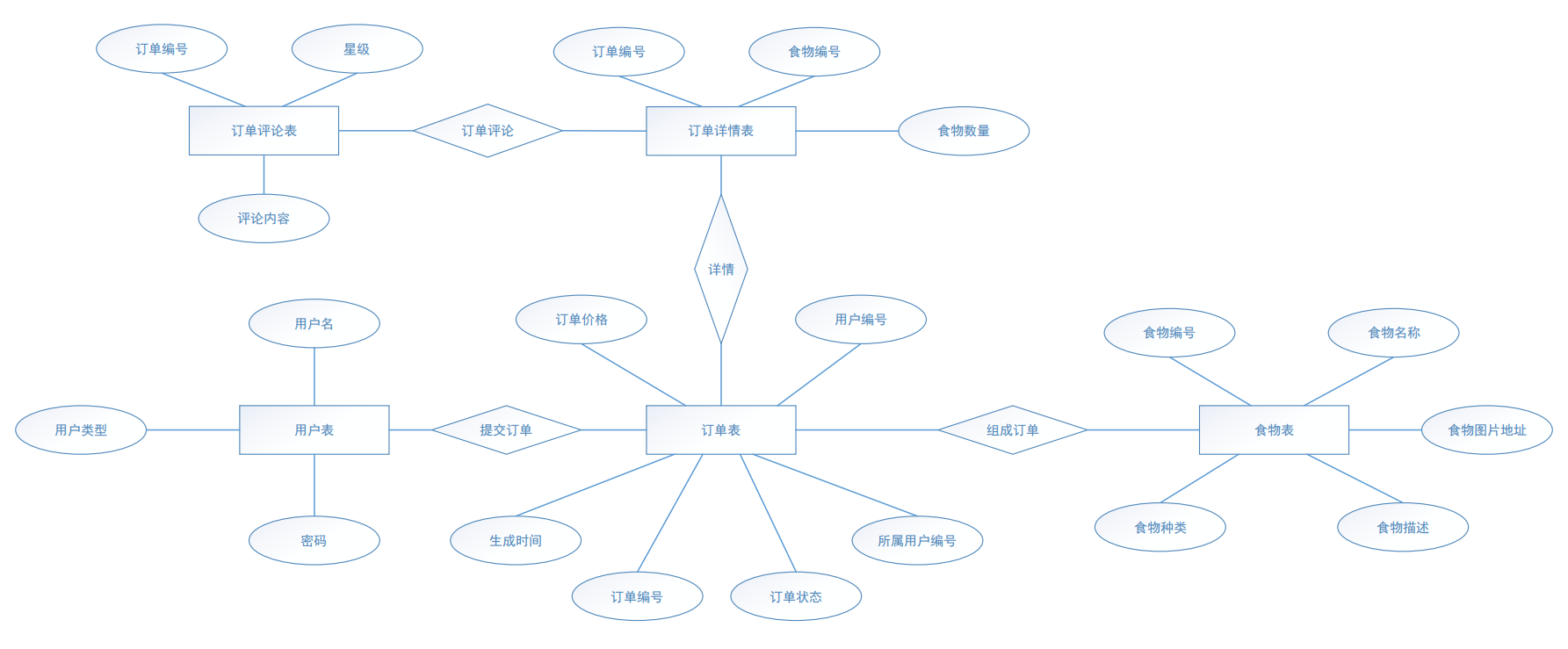
Flutter 3.4.0-17.2.pre

Dart 2.19.0

## 四、详细设计

### 4.1概念结构设计

|  |  |
| --- | --- |
| **图 4.1.1：菜品（Food）的E-R图** | 图示  描述已自动生成**图 4.1.2：订单（Order）的E-R图** |
| **图 4.1.3：订单记录（order\_content）的E-R图** | **图 4.1.4：订单评论（order\_comment）的E-R图** |
| **图 4.1.5：用户（User）的E-R图** | |

****

**图 4.1.6：总体的E-R图**

### 4.2逻辑结构设计

#### 4.2.1数据需求

1. 菜品表：包括菜品编号 菜品名称 菜品价格 菜品类型 菜品图片。
2. 用户表：包括用户编号 用户名 用户密码 用户类别。
3. 订单表：包括订单编号 用户编号 订单的状态 订单生成的时间 总价格。
4. 购物车\_订单内容（联系）:包括订单编号 菜品编号 菜品的数量。

#### 4.2.2事务需求

1. 点餐功能：顾客可以选择菜品、指定数量并下单。
2. 购物车管理：顾客可以将菜品添加到购物车，可以增加、删除或修改购
3. 物车中的项目。
4. 订单管理：餐厅工作人员可以查看、处理和更新订单状态。
5. 管理界面：管理员登录界面并实现菜品的增删改查操作

#### 4.2.3数据类型定义

**Food表**

| **列名** | **数据类型** | **可空** | **默认值** | **描述** |
| --- | --- | --- | --- | --- |
| Food\_id | int(11) | 否 |  | 菜品编号 |
| Food\_name | varchar(50) | 否 |  | 菜品名称 |
| Food\_typeid | varchar(20) | 否 |  | 菜品类型 |
| Food\_price | int(11) | 否 |  | 菜品价格 |
| Food\_discription | varchar(255) | 是 | NULL | 菜品描述 |
| Food\_img\_url | varchar(255) | 是 | NULL | 菜品图片链接 |
| Food\_sale | int(11) | 否 |  | 菜品销量 |

* 主键： Food\_id
* 唯一索引： food\_id\_index (Food\_id)

**Order表**

| **列名** | **数据类型** | **可空** | **默认值** | **描述** |
| --- | --- | --- | --- | --- |
| Order\_id | int(11) | 否 |  | 订单编号 |
| Order\_price | float(8, 2) | 否 |  | 订单价格 |
| Order\_time | datetime(2) | 否 |  | 订单时间 |
| Order\_status | int(11) | 否 |  | 订单状态 |
| User\_id | int(11) | 否 |  | 用户编号 |
| Comment\_id | int(11) | 否 |  | 评论编号 |

* 主键： Order\_id
* 唯一索引： order\_id\_index (Order\_id)
* 索引： user\_id\_index (User\_id)
* 索引： commit\_id\_fk (Comment\_id)

**外键约束：**

* **外键约束名称：** commit\_id\_fk
  + **外键列：** Comment\_id
  + **参考表：** user\_comment (comment\_id)
  + **删除操作：** RESTRICT
  + **更新操作：** RESTRICT
* **外键约束名称：** user\_fk
  + **外键列：** User\_id
  + **参考表：** user (User\_id)
  + **删除操作：** RESTRICT
  + **更新操作：** RESTRICT

**order\_content表**

| **列名** | **数据类型** | **可空** | **默认值** | **描述** |
| --- | --- | --- | --- | --- |
| Order\_id | int(11) | 否 |  | 订单编号 |
| Food\_id | int(11) | 否 |  | 菜品编号 |
| Food\_number | int(11) | 否 |  | 菜品数量 |

* **主键：** (Order\_id, Food\_id)
* **索引：** food\_id\_fk (Food\_id)

**外键约束：**

* **外键约束名称：** food\_id\_fk
  + **外键列：** Food\_id
  + **参考表：** food (Food\_id)
  + **删除操作：** RESTRICT
  + **更新操作：** RESTRICT
* **外键约束名称：** order\_id\_fk
  + **外键列：** Order\_id
  + **参考表：** order (Order\_id)
  + **删除操作：** RESTRICT
  + **更新操作：** RESTRICT

**user表**

| **列名** | **数据类型** | **可空** | **默认值** | **描述** |
| --- | --- | --- | --- | --- |
| User\_id | int(11) | 否 |  | 用户编号 |
| User\_name | varchar(20) | 否 |  | 用户名称 |
| User\_type | int(11) | 是 | NULL | 用户类型 |
| User\_password | char(20) | 否 |  | 用户密码 |

* **主键：** User\_id
* **唯一索引：** user\_id\_index (User\_id)
* **索引：** user\_name\_index (User\_name)

**user\_comment表**

| **列名** | **数据类型** | **可空** | **默认值** | **描述** |
| --- | --- | --- | --- | --- |
| comment\_id | int(11) | 否 |  | 评论编号 |
| comment\_star | int(11) | 是 | NULL | 评论星级 |
| comment\_content | varchar(255) | 是 | NULL | 评论内容 |

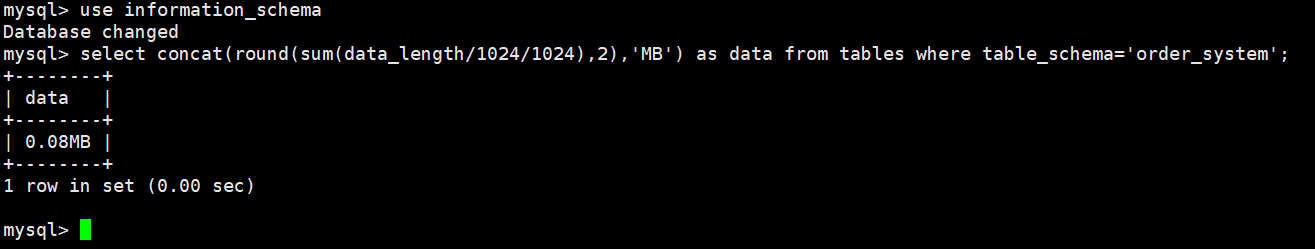
* **主键：** comment\_id
* **唯一索引：** user\_comment\_index (comment\_id)

### 4.3 物理结构设计

数据库存储位置（本地）: D:\mysql-5.7.24-winx64\data

数据库存储位置（服务器）: /www/server/data

数据库大小:



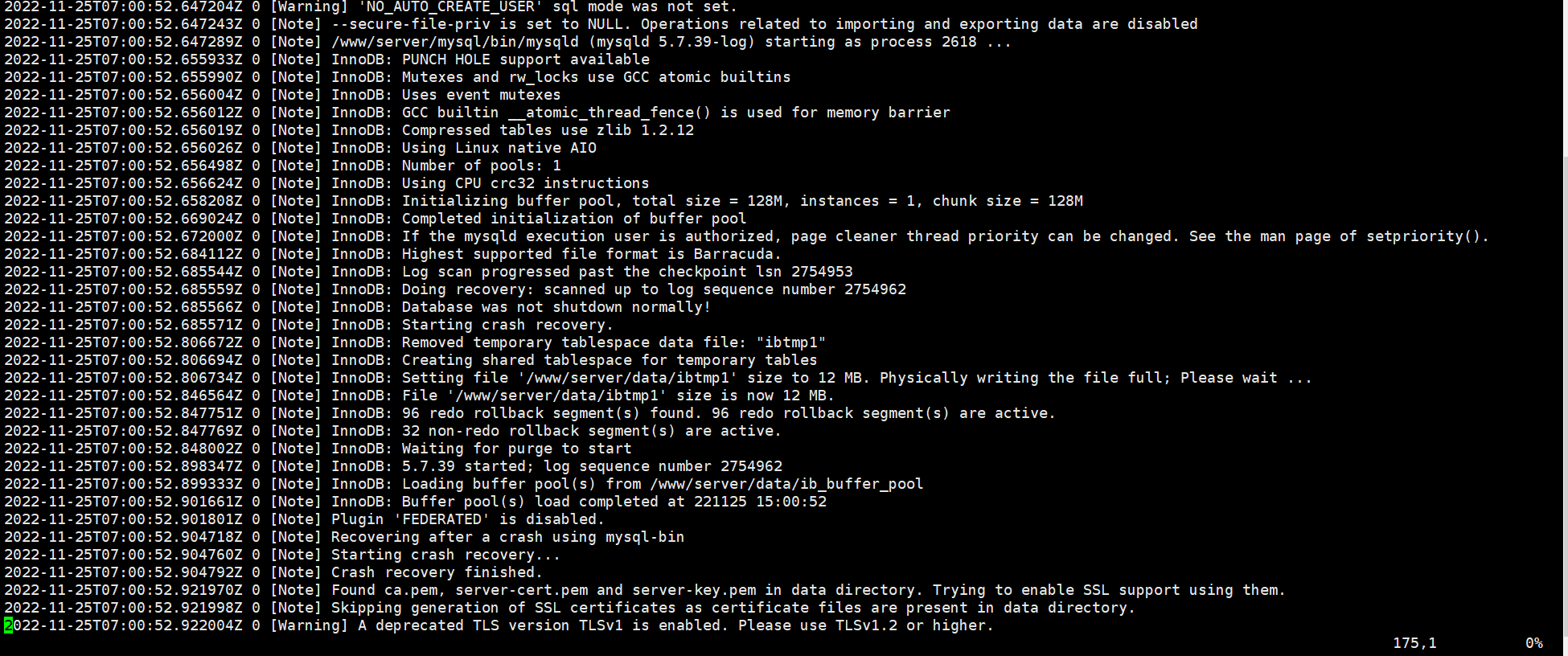
**图 4.3.1：查看数据库的大小**

Mysql中采用Innodb 引擎储存，理论最大值为64TB，实际当前数据库占用空间为0.08MB

数据库日志存储位置（本地）: D:\mysql-5.7.24-winx64\data\mysql-bin.000016

数据库日志存储位置（服务器）: /www/server/data/mysql-bin.000019

查看服务器上的错误日志/www/server/data/VM-12-4-centos.err



**图 4.3.1：查看数据库的错误日志**

### 4.4数据库的实施与维护

#### 4.4.1数据备份

在mysql的配置文件中加入命令

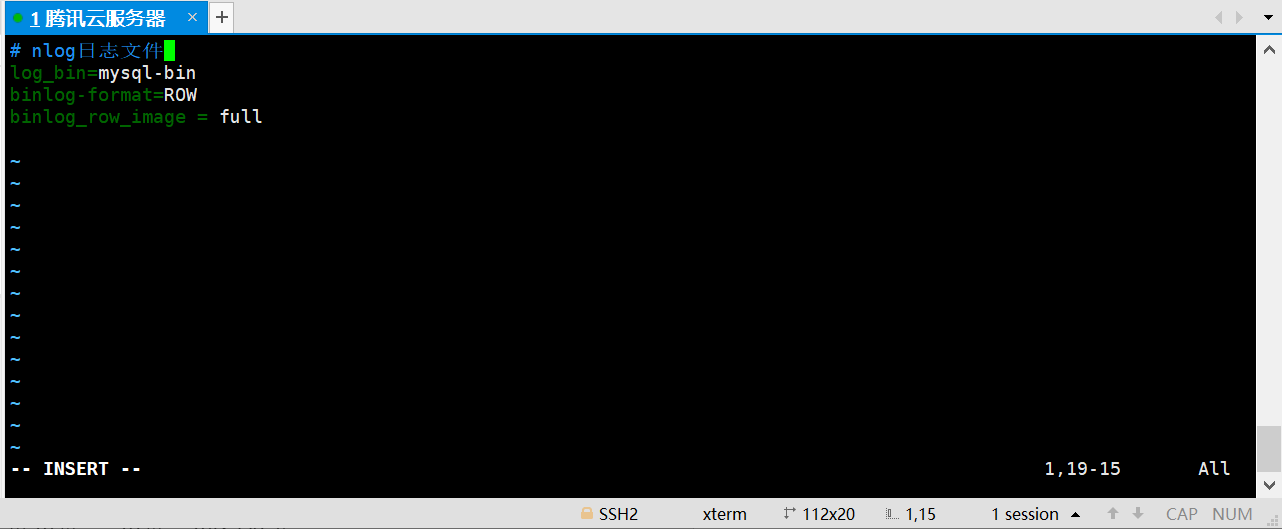
#binlog日志文件

log\_bin=mysql-bin

binlog-format=ROW

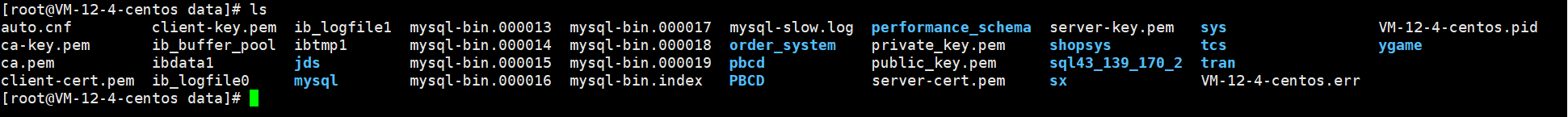
binlog\_row\_image = full

启用binlog文件，存储所有对当前数据库修改的操作以及修改内容



**图 4.4.1 改写数据库配置文件**

重启mysql服务，生成binlog文件

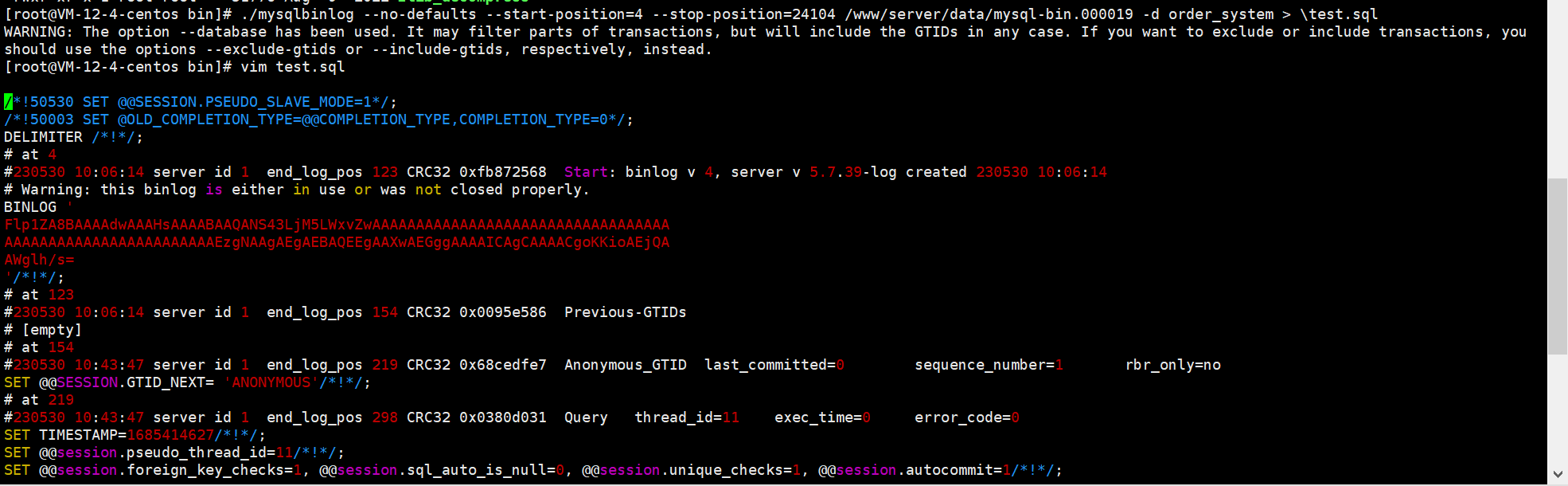


**图 4.4.2 生成备份的文件**

数据库备份工作完成

#### 4.4.2根据日志文件还原备份状态时sql语句

|  |
| --- |
| ./mysqlbinlog --no-defaults --start-position=4 --stop-position=24104  /www/server/data/mysql-bin.000019 -d order\_system > \test.sql |



**图 4.4.3 根据备份文件生成sql**

|  |
| --- |
| source /www/server/mysql/bin/test.sql |

#### 文本 描述已自动生成

**图 4.4.4 通过sql语句恢复数据**

启动数据库服务并登录，切换到需要恢复的数据库，执行语句source D:\test\_pot.sql等待执行完成就完成了对数据库的恢复工作

### 4.5视图设计

热销食物表视图

|  |
| --- |
| CREATE VIEW hot\_food AS  SELECT \* FROM food WHERE Food\_sale = 1; |

特价食物表视图

|  |
| --- |
| CREATE VIEW hot\_food AS  SELECT \* FROM food WHERE Food\_sale = 2; |

### 4.6触发器设计

不在工作时间，不允许增加，修改，删除表

|  |
| --- |
| DROP TRIGGER IF EXISTS `secure\_order\_INSERT`;  delimiter ;;  CREATE TRIGGER `secure\_order\_INSERT` BEFORE INSERT ON `order` FOR EACH ROW BEGIN  IF (HOUR(NOW()) NOT BETWEEN 9 AND 17) THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Orders can only be INSERT during business hours.';  END IF;  END  ;;  delimiter ; |
| DROP TRIGGER IF EXISTS `secure\_order\_UPDATE`;  delimiter ;;  CREATE TRIGGER `secure\_order\_UPDATE` BEFORE UPDATE ON `order` FOR EACH ROW BEGIN  IF (HOUR(NOW()) NOT BETWEEN 9 AND 17) THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Orders can only be UPDATE during business hours.';  END IF;  END  ;;  delimiter ; |
| DROP TRIGGER IF EXISTS `secure\_order\_DELETE`;  delimiter ;;  CREATE TRIGGER `secure\_order\_DELETE` BEFORE DELETE ON `order` FOR EACH ROW BEGIN  IF (HOUR(NOW()) NOT BETWEEN 9 AND 17) THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Orders can only be UPDATE during business hours.';  END IF;  END  ;;  delimiter ; |

制作中的订单不允许取消

|  |
| --- |
| DROP TRIGGER IF EXISTS `order\_prepare`;  delimiter ;;  CREATE TRIGGER `order\_prepare` BEFORE UPDATE ON `order` FOR EACH ROW BEGIN  IF NEW.order\_status = 4 AND OLD.order\_status = 2 THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Cannot update orders with order\_status = 4 and previous order\_status = 2.';  END IF;  END  ;;  delimiter ; |

## 五、成果展示

### 5.1登录界面

|  |  |
| --- | --- |
| **图 5.1.1 登录界面** | **图 5.1.2 注册界面** |

### 5.2管理员端界面

#### 5.2.1菜品管理

|  |  |
| --- | --- |
| **图 5.2.1 管理菜品** | **图 5.2.2 修改菜品信息** |

#### 5.2.2订单管理

|  |  |
| --- | --- |
| **图 5.2.1 管理员订单管理1** | **图 5.2.2 管理员订单管理2** |

#### 

#### 5.2.3.退出登录



**图 5.3.1 管理员退出登录**

### 5.3用户端设计

#### 5.3.1.首页

|  |  |
| --- | --- |
| **图5.3.1首页界面1** | **图 5.3.2 首页界面2** |

#### 5.3.2.点餐页面

|  |  |
| --- | --- |
| **图 5.1.4 点餐界面1** | **图 5.1.5 点餐界面2** |
| **图 5.3.6 点餐界面3** | **图 5.3.7 点餐界面4** |

#### 5.3.3.订单页面

|  |  |
| --- | --- |
| **图 5.3.8 订单界面1** | **图 5.3.9 订单界面2** |
| **图 5.3.10 订单界面3** | **图 5.3.11 订单界面4** |

#### 5.3.4.用户主页

****

**图 5.3.12 用户主页**

### 5.4后端接口列表

| **接口名称** | **请求方式** | **接口地址** | **参数** | **返回值** | **备注** |
| --- | --- | --- | --- | --- | --- |
| userLogin | POST | /login/ | username, password,type | 登录结果 | 用户登录 |
| getAllFood | GET | /getAllFood/ | 无 | 所有菜品信息 | 获取所有菜品信息 |
| getFoodSale | GET | /getFoodSale/ | 无 | 销量排行榜 | 获取销量排行榜 |
| getFoodHot | GET | /getFoodHot/ | 无 | 热度排行榜 | 获取热度排行榜 |
| getOrderInfo | GET | /getOrderInfo/ | order\_id | 订单信息 | 获取订单信息 |
| reg | POST | /reg/ | username,password,type | 注册结果 | 用户注册 |
| postOrder | POST | /postOrder/ | userId,orderData | 订单信息 | 提交订单 |
| pay | POST | /pay/ | order\_id | 支付结果 | 订单支付 |
| updateFood | POST | /foodEdit/ | food\_id,name,price, escription,img | 菜品信息 | 修改菜品信息 |
| getAllOrderInfo | GET | /getAllOrder/ | 无 | 所有订单信息 | 获取所有订单信息 |
| finishPreOrder | POST | /finishPreOrder/ | order\_id | 订单状态 | 完成预订单 |
| finishOrder | POST | /finishOrder/ | order\_id | 订单状态 | 完成订单 |
| cancelOrder | POST | /cancelOrder/ | order\_id | 订单状态 | 取消订单 |
| getComment | GET | /getComment/ | comment\_id | 评论信息 | 获取菜品评论信息 |
| postComment | POST | /postComment/ | comment\_content,  comment\_star | 评论结果 | 提交菜品评论 |

|  |  |  |
| --- | --- | --- |
|  |  |  |

## 六、总结

在本次数据库课设中，我们的目标是设计和实现一个功能较完善、高效可靠的饭店点餐系统。通过该课设，我们深入学习和应用了数据库设计原理、关系模型、SQL语言、触发器、数据库备份和恢复、性能优化等相关知识。

首先，我们进行了需求分析和概念设计阶段，确定了饭店点餐系统的功能需求和数据模型。在概念结构设计阶段，我们使用实体-关系模型进行了数据库的设计，包括确定实体、属性和关系，并绘制了实体关系图。

接下来，我们进行了逻辑结构设计和物理结构设计。在逻辑结构设计中，我们转换了概念结构设计的实体-关系模型为关系模式，并进行了范式分解，以提高数据库的数据一致性和效率。在物理结构设计中，我们考虑了存储结构、查询优化等方面，以确保数据库系统的高性能运行。

在实施阶段，我们利用navicat可视化MySql创建了数据库表格，并导入了相关数据。我们编写了各种SQL查询语句，以满足用户的查询需求。同时，我们还使用了触发器，并进行了性能调优，以提高数据库的响应时间。

此外，我们还进行了数据库的安全设计和数据备份策略制定。我们限制了用户的访问权限，以保护数据库的数据安全性。同时，我们也对数据库进行备份，在数据库运行过程中发生故障时实现对数据库的恢复，以确保数据库系统的可靠性和持久性。

在测试和评估阶段，我们进行了一些压力测试和异常情况测试，以验证系统的稳定性和可靠性。

最后，在总结和展望部分，我们对整个课设过程进行了总结和反思。我们回顾了课设中遇到的问题和挑战，并提出了改进的建议。同时，我们认识到数据库在现代信息系统中的重要性和广泛应用。

通过本次数据库课设，我们不仅掌握了数据库设计和实现的基本原理和方法，还培养了团队合作、问题解决和项目管理等能力。这次课设为我们今后在数据库领域的学习和工作打下了坚实的基础，使我们对数据库技术有了更深入的理解和实践经验。