# 校园外卖管理系统数据库设计报告

## 第一部分：课程设计内容

### 第一章 绪论

#### 1. 课题简介

随着校园生活的日益繁忙，校园外卖服务成为学生和教职员工日常生活中的重要部分。一个高效的校园外卖管理系统可以帮助商家、骑手和用户更好地进行外卖的点餐、配送和接收工作。本系统旨在通过数据库技术实现对校园外卖业务的全面管理，提高服务质量和用户体验。

#### 2. 设计目的

本设计的主要目的是：

* 通过数据库管理系统提高校园外卖服务的效率和准确性。
* 为商家提供方便的菜品管理和订单处理工具。
* 为骑手提供清晰的订单和配送信息。
* 为用户提供友好的点餐体验和订单跟踪功能。
* 通过数据库管理员（DBA）的角色确保系统的安全性和数据的完整性。

#### 3. 设计内容

设计内容包括：

* 需求分析：确定系统的功能需求和性能需求。
* 概念结构设计：使用E-R图描述数据的概念结构。
* 逻辑结构设计：将E-R图转换为关系模式，并进行详细设计。
* 物理结构设计：包括索引的建立和数据的存储结构。
* 系统实现：数据库的建立和系统功能的实现。
* 分析与总结：对系统进行分析，进行总结和展望。

### 第二章 需求分析

#### 1. 功能需求

**商家端功能：**

* 添加、修改、删除菜品信息。
* 查看当前订单及其状态。
* 查看并回复用户评价。

**骑手端功能：**

* 查看分配给自己的订单。
* 查看订单详情（取餐地点、送达地点）。
* 更新订单状态（如取餐、配送中、已送达）。

**用户端功能：**

* 查看所有商家及其菜品信息。
* 下订单并查看自己的订单状态。
* 查看并添加评价。

**DBA端功能：**

* 审批商家提交的菜品上架请求。
* 查看所有商家和菜品信息。
* 管理和维护系统数据库。

#### 2. 性能需求

* **响应时间**：系统应在用户操作后1秒内响应。
* **并发处理能力**：支持至少1000个用户同时在线使用。
* **数据存储**：系统应能存储至少100万条订单记录。
* **安全性**：确保用户数据和交易数据的安全，防止未授权访问。

### 第三章 概念结构设计

#### 1. 局部E-R图

**商家与菜品的关系**

plaintext

复制代码

商家 (Merchant)

|

|<--1:N--|

|

菜品 (Dish)

**用户与订单的关系**

plaintext

复制代码

用户 (User)

|

|<--1:N--|

|

订单 (Order)

**订单与骑手的关系**

plaintext

复制代码

订单 (Order)

|

|--N:1-->|

|

骑手 (Rider)

**审批关系**

plaintext

复制代码

审批 (Approval)

|

|--N:1-->|

|

菜品 (Dish)

#### 2. 全局E-R图

plaintext

复制代码

用户 (User) <---- 1:N ----> 订单 (Order) <---- N:1 ----> 商家 (Merchant)

|

|<---- N:1 ----> 骑手 (Rider)

|

|<---- 1:N ----> 配送 (Delivery)

|

|<---- 1:N ----> 审批 (Approval)

### 第四章 逻辑结构设计

#### 1. 关系模式

* **商家表 (Merchant)**: MerchantID, Name, ContactInfo, Address, Status
* **菜品表 (Dish)**: DishID, MerchantID, DishName, Image, Description, Price, Availability
* **订单表 (Order)**: OrderID, UserID, MerchantID, RiderID, OrderTime, OrderStatus
* **骑手表 (Rider)**: RiderID, Name, ContactInfo, Status
* **配送表 (Delivery)**: DeliveryID, OrderID, RiderID, PickupTime, DeliveryTime, DeliveryStatus
* **用户表 (User)**: UserID, Name, ContactInfo, Address
* **数据库管理员表 (DBA)**: DBAID, Name, ContactInfo
* **审批表 (Approval)**: ApprovalID, DishID, MerchantID, ApprovalStatus, ApprovalTime

#### 2. 数据库结构的详细设计

**商家表 (Merchant)**

sql

复制代码

CREATE TABLE Merchant (

MerchantID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

ContactInfo VARCHAR(100),

Address VARCHAR(200),

Status ENUM('营业中', '休息中') NOT NULL

);

**菜品表 (Dish)**

sql

复制代码

CREATE TABLE Dish (

DishID INT PRIMARY KEY AUTO\_INCREMENT,

MerchantID INT,

DishName VARCHAR(100) NOT NULL,

Image VARCHAR(255),

Description TEXT,

Price DECIMAL(10, 2) NOT NULL,

Availability ENUM('已上架', '待审批', '已下架') NOT NULL,

FOREIGN KEY (MerchantID) REFERENCES Merchant(MerchantID)

);

**订单表 (Order)**

sql

复制代码

CREATE TABLE `Order` (

OrderID INT PRIMARY KEY AUTO\_INCREMENT,

UserID INT,

MerchantID INT,

RiderID INT,

OrderTime DATETIME NOT NULL,

OrderStatus ENUM('已下单', '处理中', '配送中', '已完成', '已取消') NOT NULL,

FOREIGN KEY (UserID) REFERENCES User(UserID),

FOREIGN KEY (MerchantID) REFERENCES Merchant(MerchantID),

FOREIGN KEY (RiderID) REFERENCES Rider(RiderID)

);

**骑手表 (Rider)**

sql

复制代码

CREATE TABLE Rider (

RiderID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

ContactInfo VARCHAR(100),

Status ENUM('空闲', '配送中') NOT NULL

);

**配送表 (Delivery)**

sql

复制代码

CREATE TABLE Delivery (

DeliveryID INT PRIMARY KEY AUTO\_INCREMENT,

OrderID INT,

RiderID INT,

PickupTime DATETIME,

DeliveryTime DATETIME,

DeliveryStatus ENUM('已取餐', '配送中', '已送达') NOT NULL,

FOREIGN KEY (OrderID) REFERENCES `Order`(OrderID),

FOREIGN KEY (RiderID) REFERENCES Rider(RiderID)

);

**用户表 (User)**

sql

复制代码

CREATE TABLE User (

UserID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

ContactInfo VARCHAR(100),

Address VARCHAR(200)

);

**数据库管理员表 (DBA)**

sql

复制代码

CREATE TABLE DBA (

DBAID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

ContactInfo VARCHAR(100)

);

**审批表 (Approval)**

sql

复制代码

CREATE TABLE Approval (

ApprovalID INT PRIMARY KEY AUTO\_INCREMENT,

DishID INT,

MerchantID INT,

ApprovalStatus ENUM('待审批', '已审批', '已拒绝') NOT NULL,

ApprovalTime DATETIME NOT NULL,

FOREIGN KEY (DishID) REFERENCES Dish(DishID),

FOREIGN KEY (MerchantID) REFERENCES Merchant(MerchantID)

);

### 第五章 物理结构设计

#### 1. 建立索引

为了提高查询效率，需要在一些重要字段上建立索引。例如：

sql

复制代码

CREATE INDEX idx\_order\_userid ON `Order`(UserID);

CREATE INDEX idx\_order\_merchantid ON `Order`(MerchantID);

CREATE INDEX idx\_order\_riderid ON `Order`(RiderID);

CREATE INDEX idx\_dish\_merchantid ON Dish(MerchantID);

#### 2. 存储结构

数据存储使用InnoDB引擎，以支持事务和外键约束。表的物理存储结构根据数据量和查询需求进行优化，例如使用分区表存储历史订单数据。

### 第六章 系统实现

#### 1. 数据库的建立

使用SQL脚本创建所有的表和索引：

sql

复制代码

-- 创建商家表

CREATE TABLE Merchant (

MerchantID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

ContactInfo VARCHAR(100),

Address VARCHAR(200),

Status ENUM('营业中', '休息中') NOT NULL

);

-- 创建菜品表

CREATE TABLE Dish (

DishID INT PRIMARY KEY AUTO\_INCREMENT,

MerchantID INT,

DishName VARCHAR(100) NOT NULL,

Image VARCHAR(255),

Description TEXT,

Price DECIMAL(10, 2) NOT NULL,

Availability ENUM('已上架', '待审批', '已下架') NOT NULL,

FOREIGN KEY (MerchantID) REFERENCES Merchant(MerchantID)

);

-- 创建订单表

CREATE TABLE `Order` (

OrderID INT PRIMARY KEY AUTO\_INCREMENT,

UserID INT,

MerchantID INT,

RiderID INT,

OrderTime DATETIME NOT NULL,

OrderStatus ENUM('已下单', '处理中', '配送中', '已完成', '已取消') NOT NULL,

FOREIGN KEY (UserID) REFERENCES User(UserID),

FOREIGN KEY (MerchantID) REFERENCES Merchant(MerchantID),

FOREIGN KEY (RiderID) REFERENCES Rider(RiderID)

);

-- 创建骑手表

CREATE TABLE Rider (

RiderID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

ContactInfo VARCHAR(100),

Status ENUM('空闲', '配送中') NOT NULL

);

-- 创建配送表

CREATE TABLE Delivery (

DeliveryID INT PRIMARY KEY AUTO\_INCREMENT,

OrderID INT,

RiderID INT,

PickupTime DATETIME,

DeliveryTime DATETIME,

DeliveryStatus ENUM('已取餐', '配送中', '已送达') NOT NULL,

FOREIGN KEY (OrderID) REFERENCES `Order`(OrderID),

FOREIGN KEY (RiderID) REFERENCES Rider(RiderID)

);

-- 创建用户表

CREATE TABLE User (

UserID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

ContactInfo VARCHAR(100),

Address VARCHAR(200)

);

-- 创建数据库管理员表

CREATE TABLE DBA (

DBAID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

ContactInfo VARCHAR(100)

);

-- 创建审批表

CREATE TABLE Approval (

ApprovalID INT PRIMARY KEY AUTO\_INCREMENT,

DishID INT,

MerchantID INT,

ApprovalStatus ENUM('待审批', '已审批', '已拒绝') NOT NULL,

ApprovalTime DATETIME NOT NULL,

FOREIGN KEY (DishID) REFERENCES Dish(DishID),

FOREIGN KEY (MerchantID) REFERENCES Merchant(MerchantID)

);

-- 创建索引

CREATE INDEX idx\_order\_userid ON `Order`(UserID);

CREATE INDEX idx\_order\_merchantid ON `Order`(MerchantID);

CREATE INDEX idx\_order\_riderid ON `Order`(RiderID);

CREATE INDEX idx\_dish\_merchantid ON Dish(MerchantID);

#### 2. 系统实现

系统实现包括前端和后端的开发。前端使用HTML、CSS、JavaScript进行用户界面设计，后端使用Java或Python等编程语言实现业务逻辑，并使用MySQL或其他关系数据库系统进行数据存储和管理。

前端功能：

* 用户界面设计：包括登录、注册、下单、查看订单等界面。
* 表单验证：确保用户输入数据的合法性。
* 数据交互：通过Ajax或Fetch API与后端进行数据交换。

后端功能：

* 用户认证与授权：包括登录、注册、权限管理等功能。
* 数据操作：包括对数据库的增删改查操作。
* 业务逻辑处理：包括订单处理、配送调度等功能。
* 接口设计：提供RESTful API接口供前端调用。

## 第三部分 分析与总结

#### 1. 数据库分析

通过数据库的建立和系统的实现，可以有效管理校园外卖业务。数据库结构设计合理，确保数据的完整性和一致性；通过建立索引，提高了查询效率；物理存储结构的优化，满足了系统的性能需求。

#### 2. 课题总结

本课题通过设计和实现一个校园外卖管理系统，深入理解了数据库设计和应用的基本原理和技术，掌握了数据库建模、关系模式设计、SQL编程等技能。

#### 3. 项目展望

未来可以进一步优化系统性能，提高用户体验。可以考虑引入更多智能化功能，如推荐系统、数据分析等。此外，还可以扩展系统功能，覆盖更多的校园服务。

#### 4. 心得与体会

在本次课程设计中，深入了解了数据库管理系统的设计和实现过程，增强了编程能力和项目管理能力。同时，体会到了团队协作的重要性，学会了如何有效地与团队成员沟通和合作。

## 参考文献

* 《数据库系统概论》，王珊，萨师煊
* 《数据库系统基础》，Elmasri & Navathe
* MySQL官方文档
* 相关学术论文和技术博客