《数据库系统原理》课程设计

系统设计报告

题目名称：

学号及姓名：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

年 月 日

组内同学承担任务说明

|  |  |  |
| --- | --- | --- |
|  | 学生1 | 学生2 |
| 系统设计阶段 |  |  |
| 系统实现阶段 |  |  |
| 系统报告撰写 |  |  |

目录

[总目录 3](#_Toc468133744)

[一、系统需求分析 3](#_Toc468133745)

[1. 需求描述 3](#_Toc468133746)

[2. 数据流图 3](#_Toc468133747)

[**(1)** **顶层图** 3](#_Toc468133748)

[**(2)** **0层图** 3](#_Toc468133749)

[**(3)** **1层图** 3](#_Toc468133750)

[3. 数据字典 3](#_Toc468133751)

[二、系统概念模式 3](#_Toc468133752)

[1. 系统初步ER图 3](#_Toc468133753)

[2. 系统基本ER图 4](#_Toc468133754)

[三、系统逻辑模式 4](#_Toc468133755)

[1. 关系模式转化 4](#_Toc468133756)

[2. 关系模式范式等级分析 5](#_Toc468133757)

[3. 数据库设计优化 7](#_Toc468133758)

[**(1)** **建立索引** 7](#_Toc468133759)

[(2) **建立视图** 7](#_Toc468133760)

[四、最终版修改说明 7](#_Toc468133761)

# 一、系统需求分析

## 1. 需求描述

具备外卖系统所具有的菜品分类、推荐、搜索、购物车、订单、在线支付、配送、会员系统。

具体来说，用户可以在网上注册，成为会员，并以顾客、配送员、商家的身份之一使用系统；顾客可以通过浏览器获取菜品推荐，浏览菜品信息，浏览店铺信息，选取菜品加入购物车，获取并使用优惠券，可以提交订单并在线支付，可以评价菜品以及配送服务；商家可以管理自己店铺内的菜品，包括添加、删除、上架、下架等，可以处理订单，收取订单款；配送员可以处理配送任务，获得服务评价。

## 2. 数据流图

1. **顶层图**

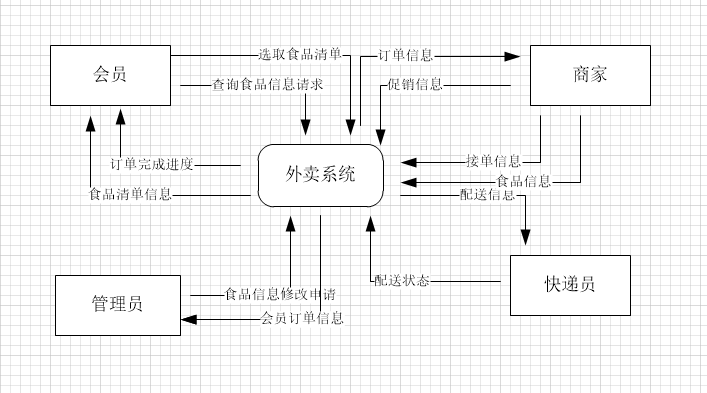


图1.1.1 数据流图顶层图

1. **0层图**

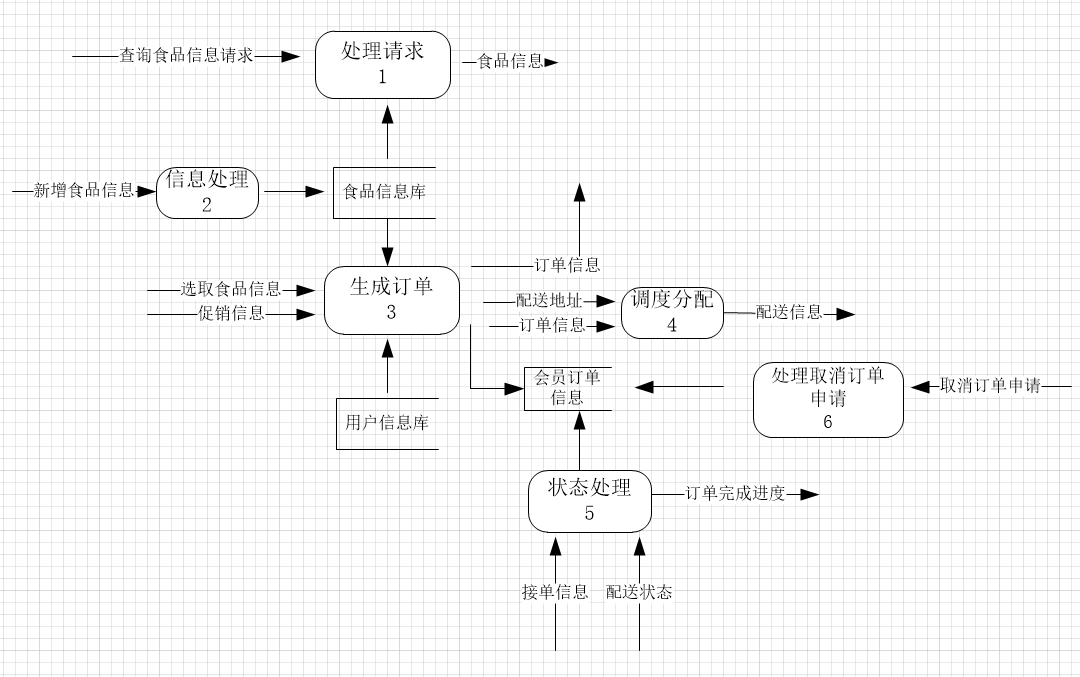


图1.2.1 数据流图0层图

1. **1层图**

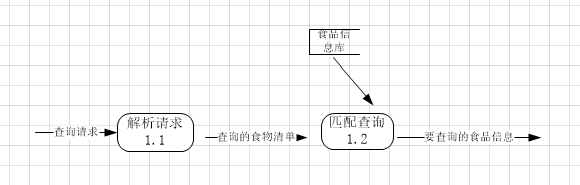


图1.3.1 数据流图1层图（1）

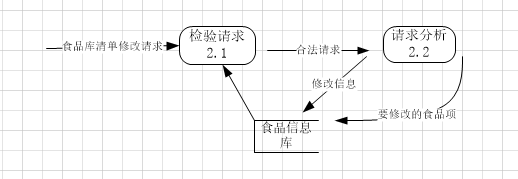


图1.3.2 数据流图1层图（2）

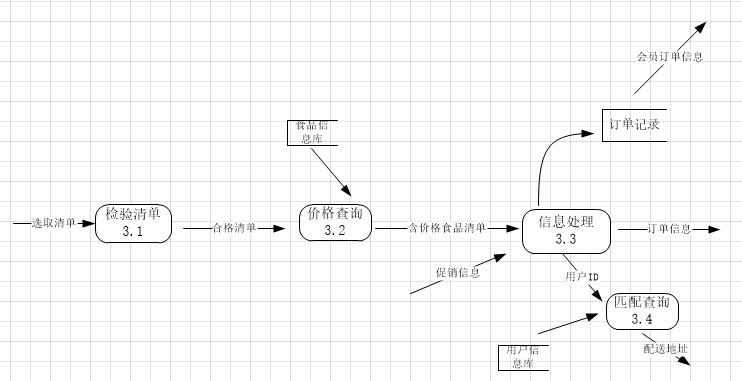


图1.3.3 数据流图1层图（3）

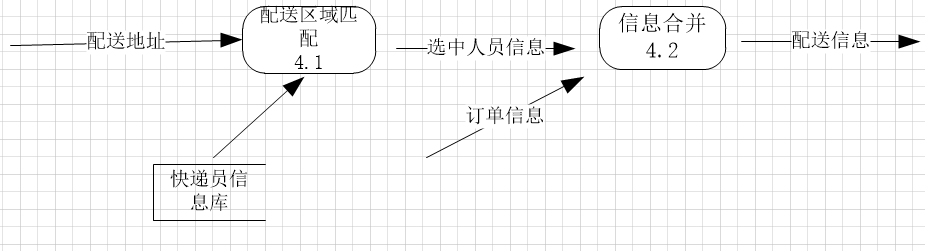


图1.3.4 数据流图1层图（4）

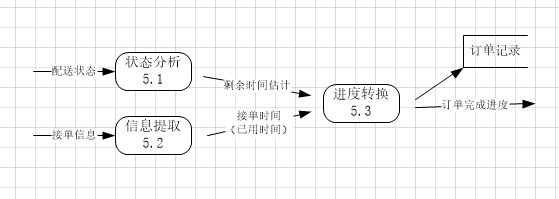


图1.3.5 数据流图1层图（5）

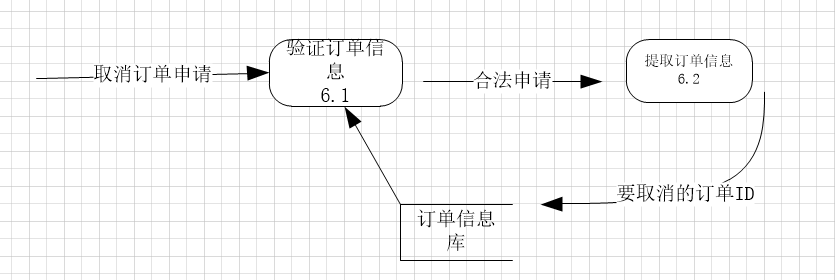


图1.3.6 数据流图1层图（6）

## 数据字典

从上述数据流图中，我们总结出以下数据。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 数据组名：用户（客户，商家，快递员）数据 | | | | | | | | |
| 特征 | 用户ID | 用户名 | 密码 | 类型 | 邮箱 | 手机号 | 地址 | 评分 |
| 用户标识符 | 用户自选名 | 用户密码 | 用户类型 | 验证邮箱 | 手机号码 | 长期地址 | 快递用户评分 |
| 数据类型 | 数值型 | 字符型 | 字符型 | 数值型 | 字符型 | 数值型 | 字符型 | 数值型 |
| 数据宽度 | 10 | 10 | 32 | 1 | 20 | 15 | 100 | 1 |
| 值约束 | 无 | 无特殊字符 | 无 | [0,3] | 符合格式 | 无 | 无 | [0,5] |
| 允许空值否 | 否 | 否 | 否 | 否 | 否 | 是 | 是 | 是 |

表1.1 用户数据

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据组名：店铺数据 | | | | |
| 特征 | 店铺ID | 信用 | 收入 | 简介 |
| 用户标识符 | 信用值 | 商家用户收入 | 店铺的简介 |
| 数据类型 | 数值型 | 数值型 | 实数型 | 字符型 |
| 数据宽度 | 11 | 1 | 10,2 | 1024 |
| 值约束 | 无 | [0,5] | 无 | 无 |
| 允许空值否 | 否 | 否 | 否 | 是 |

表1.2 商店数据

|  |  |  |  |
| --- | --- | --- | --- |
| 数据组名：快递员数据 | | | |
| 特征 | 快递员ID | 评分 | 状态 |
| 用户标识符 | 快递用户评分 | 快递员当前是否处于空闲状态 |
| 数据类型 | 数值型 | 数值型 | 数值型 |
| 数据宽度 | 11 | 1 | 1 |
| 值约束 | 无 | [0,5] | [0,1] |
| 允许空值否 | 否 | 否 | 否 |

表1.3 快递员数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 数据组名：订单数据 | | | | | | | |  |  |
| 特征 | 订单ID | 用户ID | 商家ID | 下单时间 | 金额 | 状态 | 评价 | 菜品ID | 折扣规则 |
| 订单标识符 | 用户标识符 | 商家标识符 | 下订单的时刻 | 订单金额 | 订单完成状态 | 用户评价 | 所选菜品ID | 适用的折扣 |
| 数据类型 | 数值型 | 数值型 | 数值型 | 数值型 | 数值型 | 数值型 | 字符型 | 数值型 | 数值型 |
| 数据宽度 | 10 | 10 | 10 | 10 | 20 | 1 | 100 | 100 | 10 |
| 值约束 | 无 | 无 | 无 | 符合时间格式 | 无 | [0,4] | 无 |  | (0,1] |
| 允许空值否 | 否 | 否 | 否 | 否 | 否 | 是 | 是 | 否 | 否 |

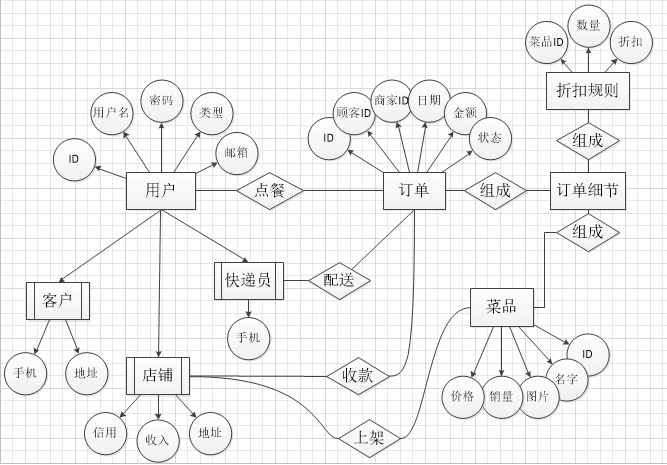
表1.4订单数据

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 数据组名：菜品数据 | | | | | |
| 特征 | 菜品ID | 菜品名 | 菜品图片（地址） | 销量 | 金额 |
| 菜品标识符 | 菜品名称 | 菜品图片地址 | 菜品销量 | 菜品单价 |
| 数据类型 | 数值型 | 字符型 | 字符型 | 数值型 | 数值型 |
| 数据宽度 | 10 | 20 | 30 | 10 | 10 |
| 值约束 | 无 | 无 | 无 | 符合时间格式 | 无 |
| 允许空值否 | 否 | 否 | 是 | 否 | 否 |

表1.5 菜品数据

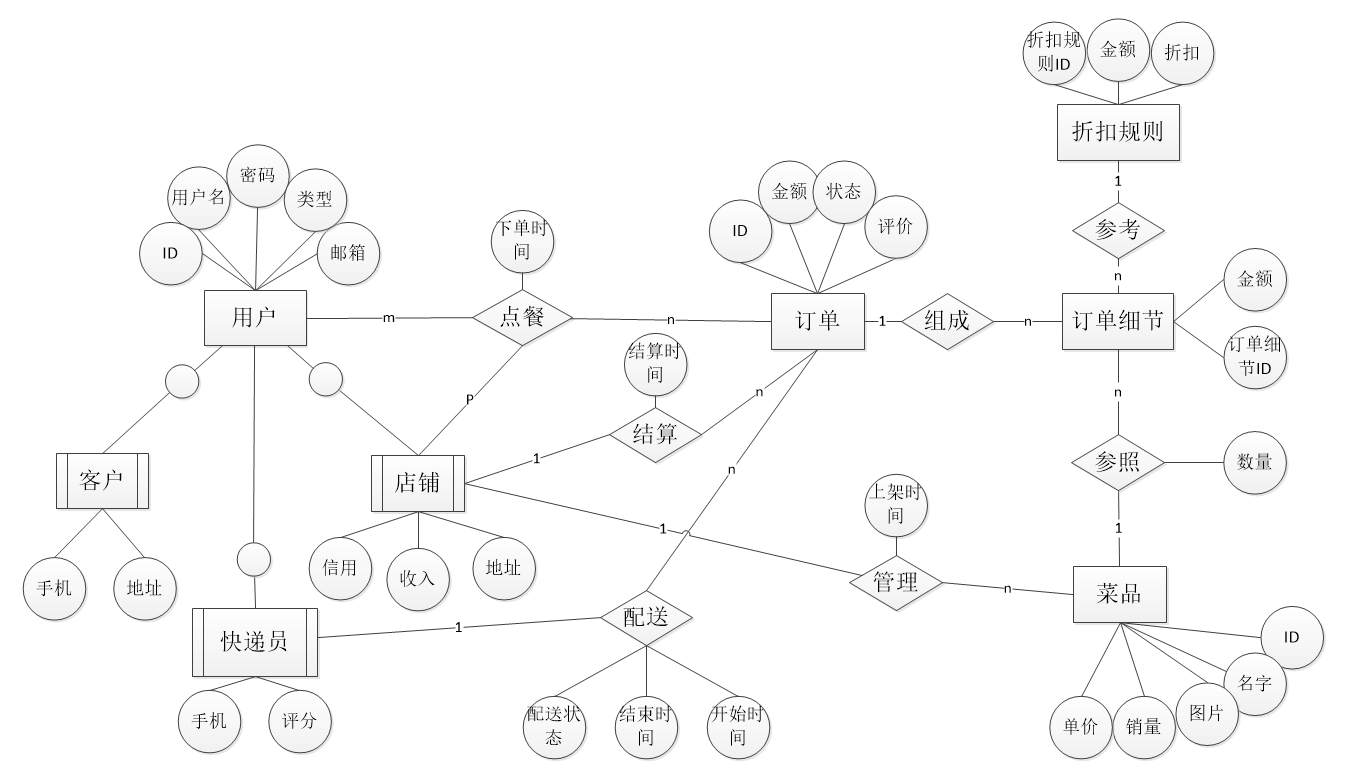
# 二、系统概念模式

## 1. 系统初步ER图



系统初步ER图

## 2. 系统基本ER图



系统设计ER图

# 三、系统逻辑模式

## 1. 关系模式转化

将上述ER图转化为关系模型，并进行相应的合并，获得以下8个关系模式（函数依赖集均已转化为极小依赖集）：

1. 用户<（用户ID，用户名，密码，类型，邮箱）,（用户ID->用户名，用户ID->密码，用户ID->类型，用户ID->邮箱，邮箱->用户ID，用户名->用户ID）>

码：{(用户ID),(用户名)，（邮箱）}

1. 客户<（客户ID（用户ID），手机号，地址），（客户ID->手机号，客户ID->地址，手机号->客户ID）>

码：{(客户ID)，（手机号）}

1. 店铺<（店铺ID（用户ID），信用，收入，地址），（店铺ID->信用，店铺ID->收入，店铺ID->地址）>

码：{(店铺ID)}

1. 快递员<（快递员ID（用户ID），手机，评分），（快递员ID->手机号，快递员ID->评分，手机号->快递员ID）>

码：{(快递员ID)，（手机号）}

1. 订单<（订单ID，用户ID，店铺ID，快递员ID，金额，下单时间，状态，评价，结算时间，开始配送时间，送达时间，配送状态），（订单ID->用户ID，订单ID->店铺ID，订单ID->快递员ID，订单ID->金额，订单ID->下单时间，订单ID->状态，订单ID->评价，订单ID->结算时间，订单ID->开始配送时间，订单ID->配送状态，订单ID->送达时间）>

码：{(订单ID)}

1. 折扣规则<（规则ID，金额，折扣），（规则ID->金额，规则ID->折扣，金额->规则ID）>

码：{(规则ID)}

1. 订单细节<（订单细节ID，订单ID，折扣规则ID，菜品ID，数量，金额），（订单细节ID->折扣规则ID，订单细节ID->菜品ID，订单细节ID->订单ID，订单细节ID->数量，（折扣规则ID，菜品ID，数量）->金额），（（订单ID， 菜品ID）->订单细节ID）>

码：{(订单细节ID)，（订单ID，菜品ID）}

1. 菜品<（菜品ID，名字，图片，销量，单价，店铺ID，上架时间）,（菜品ID->名字，菜品ID->图片，菜品ID->销量，菜品ID->单价，菜品ID->店铺ID，菜品ID->上架时间）>

码：{(菜品ID)}

## 关系模式范式等级分析

对于关系模式1,2,3,4,5,6,8，在我们上面列出的最小依赖集中，其任一函数依赖的左部都是关系模式的码，因此已经达到了BCNF。

而对于关系模式7，其具有函数依赖（折扣规则ID，菜品ID，数量）->金额，因而可以找出：订单细节ID->（折扣规则ID，菜品ID，数量）,（折扣规则ID，菜品ID，数量）->金额，而（折扣规则ID，菜品ID，数量）无法推出订单细节ID，从而证明函数依赖不是3NF，而不存在非码元素对于码的部分函数依赖，所以关系模式7是2NF。

首先，保持函数依赖将其规范到3NF，根据其函数依赖左部将其分为两组R1=<（订单细节，折扣规则ID，菜品ID，数量，订单ID），（订单细节ID->折扣规则ID，订单细节ID->菜品ID，订单细节ID->数量，订单细节ID->订单ID）>，R2=<（折扣规则ID，菜品ID，数量，金额），（（折扣规则ID，菜品ID，数量）->金额）>。两关系模式属性集的交集是R2的码，因此该分解也具有无损连接性。

**规范到3NF的关系模式**

根据上述分析，最终我们得到的关系模式为（所有函数依赖集均已转化为极小依赖集）：

1. 用户<（用户ID，用户名，密码，类型，邮箱）,（用户ID->用户名，用户ID->密码，用户ID->类型，用户ID->邮箱，邮箱->用户ID，用户名->用户ID）>

码：{(用户ID),(用户名)，（邮箱）}

1. 客户<（客户ID（用户ID），手机号，地址），（客户ID->手机号，客户ID->地址，手机号->客户ID）>

码：{(客户ID)，（手机号）}

1. 店铺<（店铺ID（用户ID），信用，收入，地址），（店铺ID->信用，店铺ID->收入，店铺ID->地址）>

码：{(店铺ID)}

1. 快递员<（快递员ID（用户ID），手机，评分，状态），（快递员ID->手机号，快递员ID->评分，手机号->快递员ID）>

码：{(快递员ID)，（手机号）}

1. 订单<（订单ID，用户ID，店铺ID，快递员ID，金额，下单时间，状态，评价，结算时间，开始配送时间，送达时间，配送状态），（订单ID->用户ID，订单ID->店铺ID，订单ID->快递员ID，订单ID->金额，订单ID->下单时间，订单ID->状态，订单ID->评价，订单ID->结算时间，订单ID->开始配送时间，订单ID->配送状态，订单ID->送达时间）>码：{(订单ID)}
2. 折扣规则<（规则ID，金额，折扣），（规则ID->金额，规则ID->折扣，金额->规则ID）>码：{(规则ID)}
3. 订单细节1<（订单细节，折扣规则ID，菜品ID，数量，订单ID），（订单细节ID->折扣规则ID，订单细节ID->菜品ID，订单细节ID->数量，订单细节ID->订单ID）>

码：{(订单细节ID)}

1. 订单细节2<（折扣规则ID，菜品ID，数量，金额），（（折扣规则ID，菜品ID，数量）->金额）>

码：{(折扣规则ID，菜品ID，数量)}

1. 菜品<（菜品ID，名字，图片，销量，单价，店铺ID，上架时间）,（菜品ID->名字，菜品ID->图片，菜品ID->销量，菜品ID->单价，菜品ID->店铺ID，菜品ID->上架时间）>

码：{(菜品ID)}

## 3. 数据库设计优化

1. **建立索引**

出于加快查询速度的考量，为查询较多的表建立了索引，以此加快查询速度，具体添加索引的表及其索引如下：

用户：用户ID

客户：客户ID

店铺：店铺ID

快递员：快递员ID

菜品：店铺ID

对于用户、客户、店铺、快递员这四种角色，最多的操作就是根据id查找其相应属性，因此以ID为索引可以极大地加快查询速度。

而对于一个订单，与它有关的所有用户都要对其进行查询，包括客户、店铺以及快递员。以一个客户为例，其需查询所有与自己有关的订单，唯一的查询依据就是该客户的ID，对店铺和快递员也是同理。在这种情况下，单纯的索引并不能显著加快查询速度，反而会因为建立索引而减慢插入速度。因此没有对订单建立索引。

对于一个菜品，在我们系统中最常用到的操作有两个，一是根据店铺的ID查询及管理其所属的所有菜品，二是用户在一家店内选择菜品进行下单。这两个操作都需以商家的ID为依据，因此以商家ID为索引是一个很好的选择。因此我们最终以商家ID为索引。

1. **建立视图**

同样是出于加快查询速度的考量，建立了如下视图：

**空闲状态的快递员**：从所有快递员中选出处于空闲状态（即可进行送餐）的快递员。

在对订单进行分配时需要从空闲的快递员中选出合适的，建立空闲快递员视图可以即方便查询操作，也可以加快查询速度。

**用户信息**：为登陆及每次连接时都需访问的用户名、用户密码、用户角色建立视图。

用户信息时在整个系统中使用频率最高的信息。在用户每一次连接服务器时都需根据cookie信息查询该用户的用户名、用户所属角色，因此我们为这些数据建立了视图以优化查询速度。

**优秀店铺**：按信用从高到低的顺序选出信用等级高的店铺及其相关数据。

优秀店铺的信息主要用于首页推荐一栏，如前文所述店铺表是以店铺ID为索引，而此时我们需要根据每个店铺的信用等级对其进行排序并获取较为优秀的店铺推荐给用户，因此建立了相关的视图，以店铺的信用等级从高到低进行排序。

# 四、最终版修改说明

最终版设计文档中增添了对数据库查询的优化，包括建立索引和建立视图，另外修改了排版，增添了目录，重新整理了基本ER图。