《数据库系统原理》大作业

系统设计报告

题目名称：　　“飨味通”点菜评价系统

学号及姓名：__20373516 曹伯炜___

__21373325 陈绍锋___

__21373421 张辰珏___

2023 年　　12 月　　15 日

组内同学承担任务说明

学生姓名	工作内容			工作量占比（组内同学总和为 1）
	子任务 1：系统功能设计与数据库设计	子任务 2：系统服务器端开发	子任务 3：系统客户端开发	
曹伯炜	数据库设计 设计报告	前端： 点菜界面 订单界面		33. 3%
陈绍锋	数据库设计 设计报告 实现报告	前端： 主页	后端： 模型建立 视图处理函数	33. 3%
张辰珏	数据库设计 设计报告	前端： 评价界面 管理员界面 账户管理界面		33. 3%

一、需求分析

1.1 需求描述

目前市面上点菜评价软件只允许在店铺评论区对店铺的某几道菜或店铺笼统的评价，无法提供对具体菜品的专属评价，且不提供筛选功能帮助用户寻找某道菜的评价。在此背景下，我们针对这一痛点萌生了"飨味通"点菜评价系统的想法。此外我们进一步延伸出四大特点：

- 具体菜品的专属评价列表：帮助用户针对性了解心仪菜品，参考他人对该菜品的消费感受，为消费决策提供细节性指导
- 一体化管理系统：提高餐厅管理效率并减少用户与系统的切换成本
- 实时评价和反馈系统：保障用户及时分享消费体验，为其他用户提供参考
- 多维度菜品信息：帮助用户全面了解菜品以及菜品背后的故事。其中菜品组和菜品标签有两点区别。其一，菜品组用于点菜界面分类查看不同菜品组中的菜品，菜品标签则无法在点菜界面作为分类标准查看不同标签的菜品。其二，一个菜品只属于一个菜品组，而一个菜品则可以同时拥有多样菜品标签。举例：菜品组(主食、炒菜)，菜品标签(甜、辣)

用户行为分析：用户在“飨味通”的行为主要包括浏览菜品信息，查看菜品评价，点菜，查看订单，收藏菜品和评价菜品。看看点评与推荐，已成为当下许多年轻人的日常操作，在选择菜品这一和生活息息相关的消费决策上，用户通过查看他人图文结合、真实走心的高质量点评获得关于菜品的反馈，以此为参考作出决策。用餐后，用户主动发布自己关于菜品的点评，分享自己的消费体验，为其他用户提供参考。

管理员在“飨味通”的行为主要包括添加菜品，修改库存，添加菜品组，添加标签。管理员通过参考用户关于菜品的评价，调整菜品的库存满足用户需求，为用户提供更好的就餐体验。此外，管理员推陈出新不断上架新菜品，丰富用户用餐选择；新建新菜品组，打造多谱系美食；增加新标签，为不同口味的用户提供参考以便选择更合适的菜品。通过对产品、管理和服务的调整和改进，提升用户满意度和口碑，提升品牌知名度。通过瀑布流和菜品分类的形式，使得菜品一目了然，帮助用户快速发现美食。简约的分类导航栏，高度集成点菜评价相关功能，拒绝冗余，提高页面使用率，提升用户和管理员的使用体验。

1.1.1 用户相关功能

- 注册登录：
  - 用户注册(用户名相同显示"用户名已注册"，手机号相同显示"手机号已注册")
  - 用户名登录
  - 手机号登录
- 账户管理：
  - 展示个人信息
  - 更新个人用户名
  - 更新个人手机号
  - 更新个人密码
  - 更新个人头像(刚注册时头像为微信默认头像)
- 点菜：
  - 查看菜品
  - 查看菜品详细介绍(名字、评分和详情)
  - 根据菜品组分类查看不同种类菜品，如“炒菜”、“主食”
  - 点菜下单(请求购买超过库存数量的菜品收到"库存不足"提示，库存为0商品显示售罄且无法加入购物车)
  - 调整点菜数量
  - 显示购物车(购物车仅显示选购的菜品)
  - 删除购物车商品
  - 增加购物车商品
  - 查看购物车商品总数量
  - 查看购物车商品总价格
  - 结账(跳转主页)
- 评价：
  - 评价购买过的菜品
  - 查看他人评价
  - 删除自己对购买过的菜品的某一条评价
  - 收藏菜品
  - 取消收藏菜品
  - 为已结账菜品打分(0-10分)
  - 查看菜品评分
- 订单：
  - 查看订单信息

1.1.2 管理员相关功能

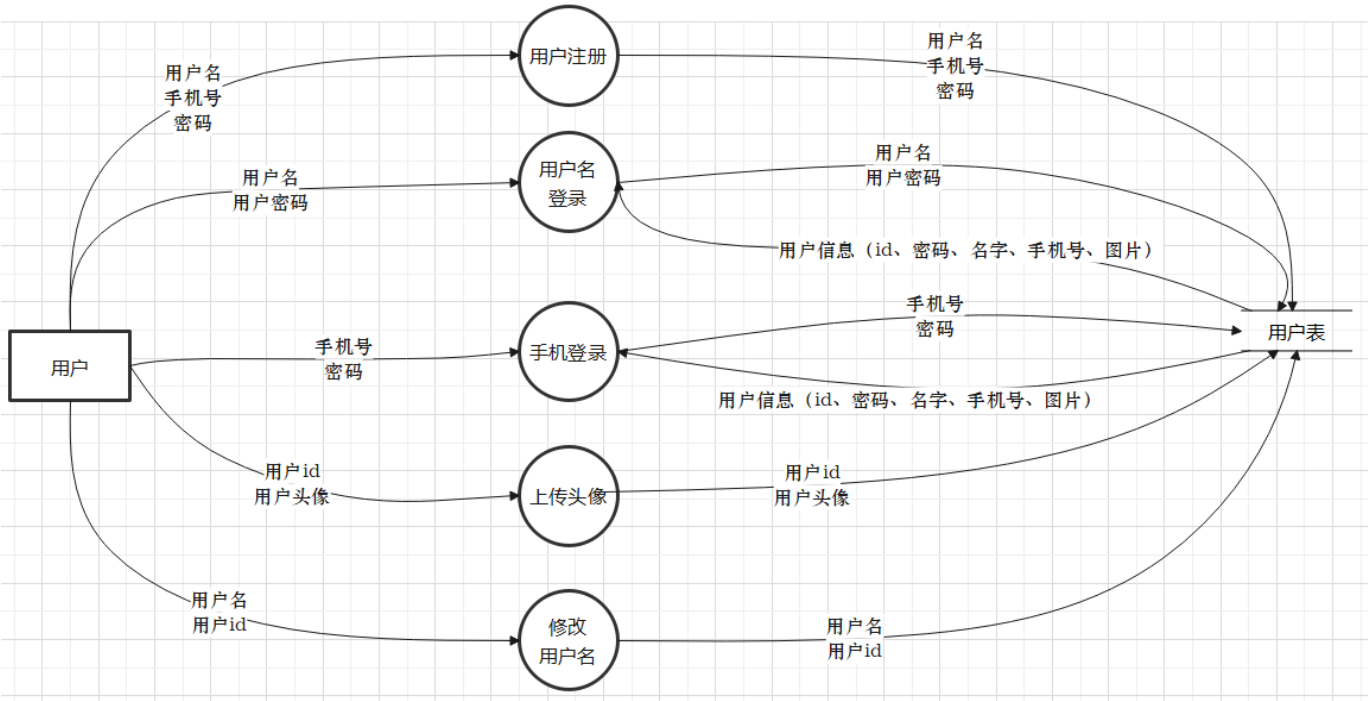
- 管理员：
  - 新建菜品组
  - 新建菜品标签
  - 新建菜品(必须选择菜品组)
  - 为已添加菜品添加标签

- 调整菜品库存

1.2 数据流图

1.2.1 用户登录数据流图

用户可以实现注册，两种登录方式并展示和更新个人信息

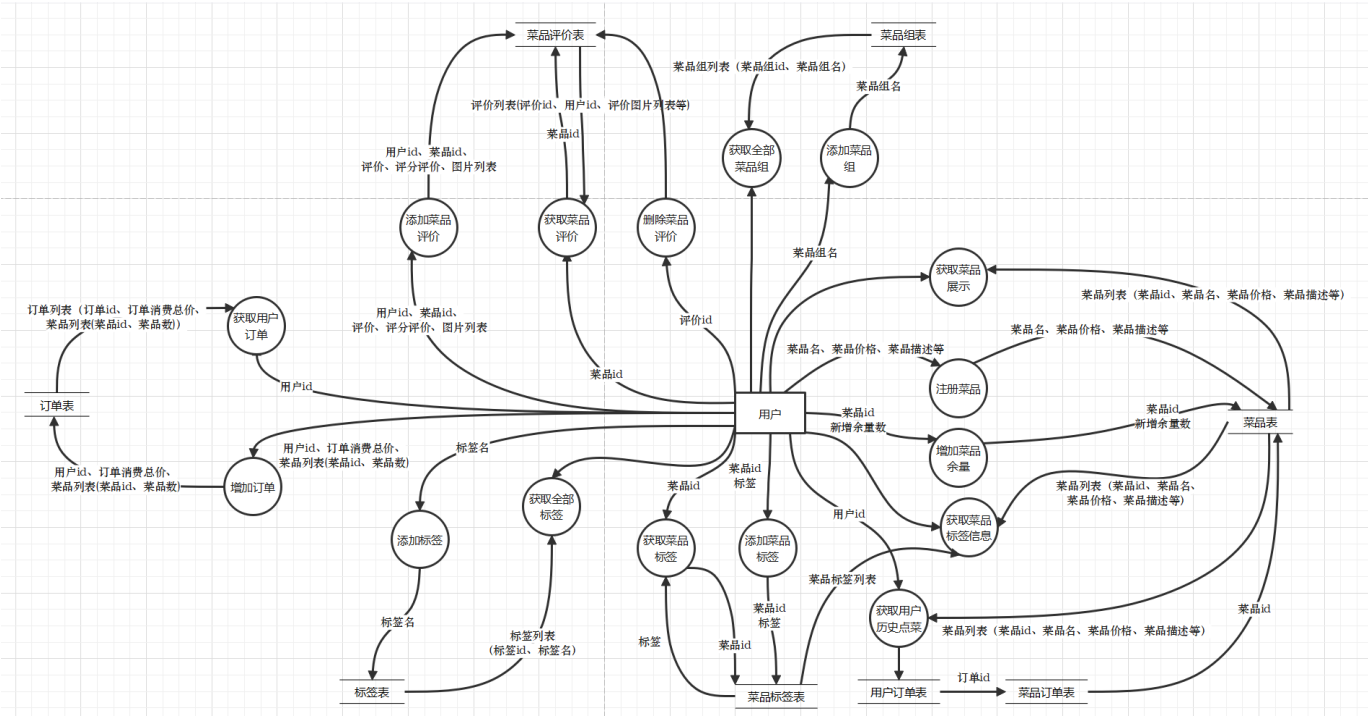


1.2.2 菜品相关数据流图

菜品相关数据分为管理员和用户两种角色。

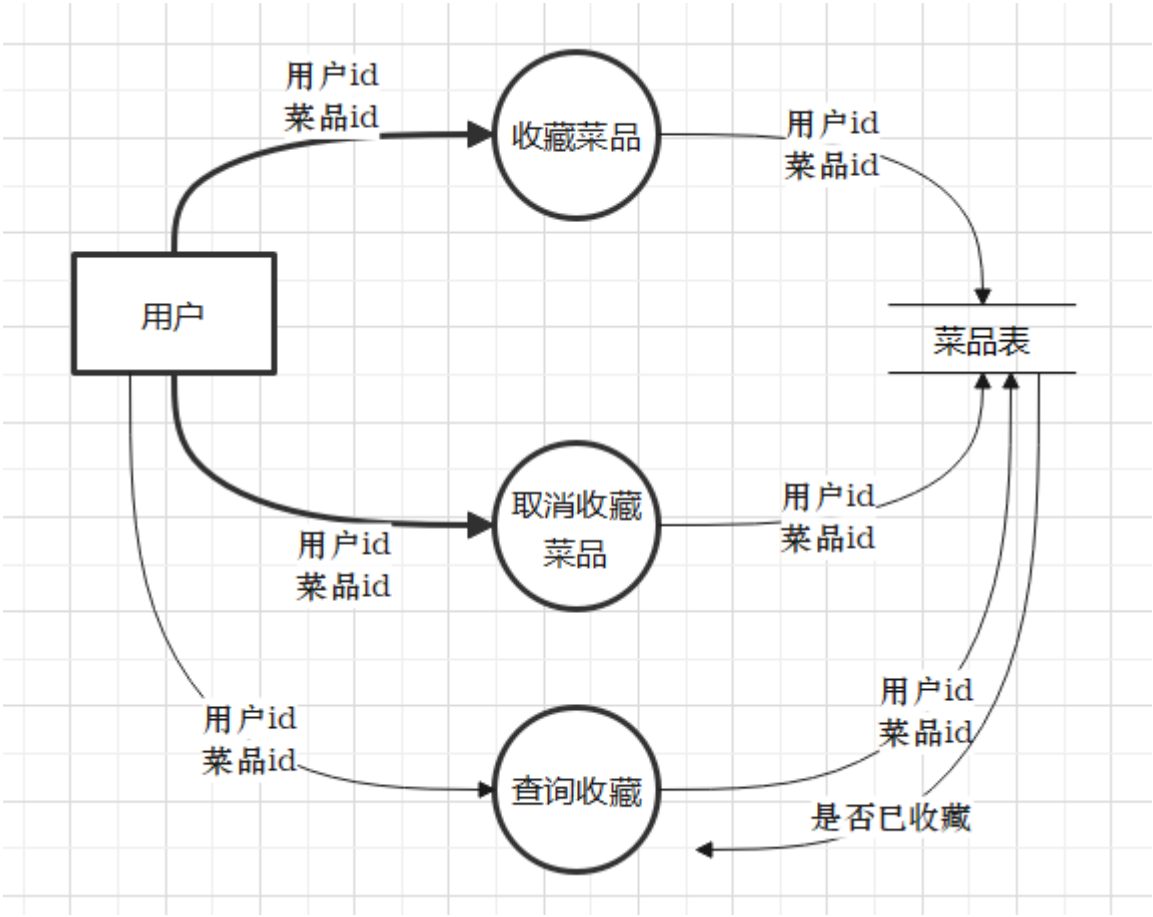
管理员可以进行添加菜品组，注册菜品，增加菜品余量，添加标签等改变菜品属性的操作。

用户可以获取全部菜品组，获取菜品展示，获取菜品标签，获取用户订单等围绕菜品展开的一系列活动，但不能改变除评分以外的菜品属性。



1.2.3 用户收藏数据流图

用户可以收藏或取消或查询收藏菜品。



1.3 数据元素表

1.3.1 用户表(表名为User)

字段	名称	类型	约束
Uid	Uid	char(8)	PRIMARY KEY
Uname	用户名	varchar(20)	NOT NULL
phone	用户手机	char(11)	NOT NULL
pwd	用户密码	varchar(20)	NOT NULL
Pid	用户头像	char(8)	FOREIGN KEY

1.3.2 菜品表(表名为Dish)

字段	名称	类型	约束
Did	Did	char(8)	PRIMARY KEY
Dname	菜品名	varchar(20)	NOT NULL
price	菜品价格	decimal(5,1)	NOT NULL
description	菜品描述	varchar(500)	NOT NULL
score	菜品评分	decimal(3,2)	NOT NULL
star	菜品收藏量	decimal(5,0)	NOT NULL
remain	菜品剩余	decimal(5,0)	NOT NULL
Pid	菜品图片	char(8)	FOREIGN KEY
Gid	菜品菜品组	char(8)	FOREIGN KEY

1.3.3 菜品组表(表名为Group)

字段	名称	类型	约束
Gid	Gid	char(8)	PRIMARY KEY
Gname	菜品组名	varchar(20)	NOT NULL

1.3.4 评价表(表名为Comment)

字段	名称	类型	约束
Cid	Cid	char(8)	PRIMARY KEY
Uid	评论用户	char(8)	FOREIGN KEY
Did	评论菜品	char(8)	FOREIGN KEY
content	评论内容	varchar(500)	NOT NULL
score	评论评分	decimal(3,2)	NOT NULL

1.3.5 标签表(表名为Tag)

字段	名称	类型	约束
Tid	Tid	char(8)	PRIMARY KEY
Tname	标签名	varchar(20)	NOT NULL

1.3.6 订单表(表名为Order)

字段	名称	类型	约束
Oid	Oid	char(8)	PRIMARY KEY
Uid	订单用户	char(8)	FOREIGN KEY
total	订单总价	decimal(5,1)	NOT NULL
time	下单时间	date	NOT NULL

1.3.7 图片表(表名为Picture)

字段	名称	类型	约束
Pid	Pid	char(8)	PRIMARY KEY
path	图片数据	longBLOB	NOT NULL

1.3.8 菜品标签表(表名为DishTag)

字段	名称	类型	约束
Tid	Tid	char(8)	PRIMARY KEY & FOREIGN KEY
Did	Did	char(8)	PRIMARY KEY & FOREIGN KEY

1.3.9 用户收藏表(表名为UserStarDish)

字段	名称	类型	约束
Uid	Uid	char(8)	PRIMARY KEY & FOREIGN KEY
Did	Did	char(8)	PRIMARY KEY & FOREIGN KEY

1.3.10 菜品订单表(表名为DishOrder)

字段	名称	类型	约束
Oid	Oid	char(8)	PRIMARY KEY & FOREIGN KEY
Did	Did	char(8)	PRIMARY KEY & FOREIGN KEY
number	点菜数量	decimal(5,0)	NOT NULL

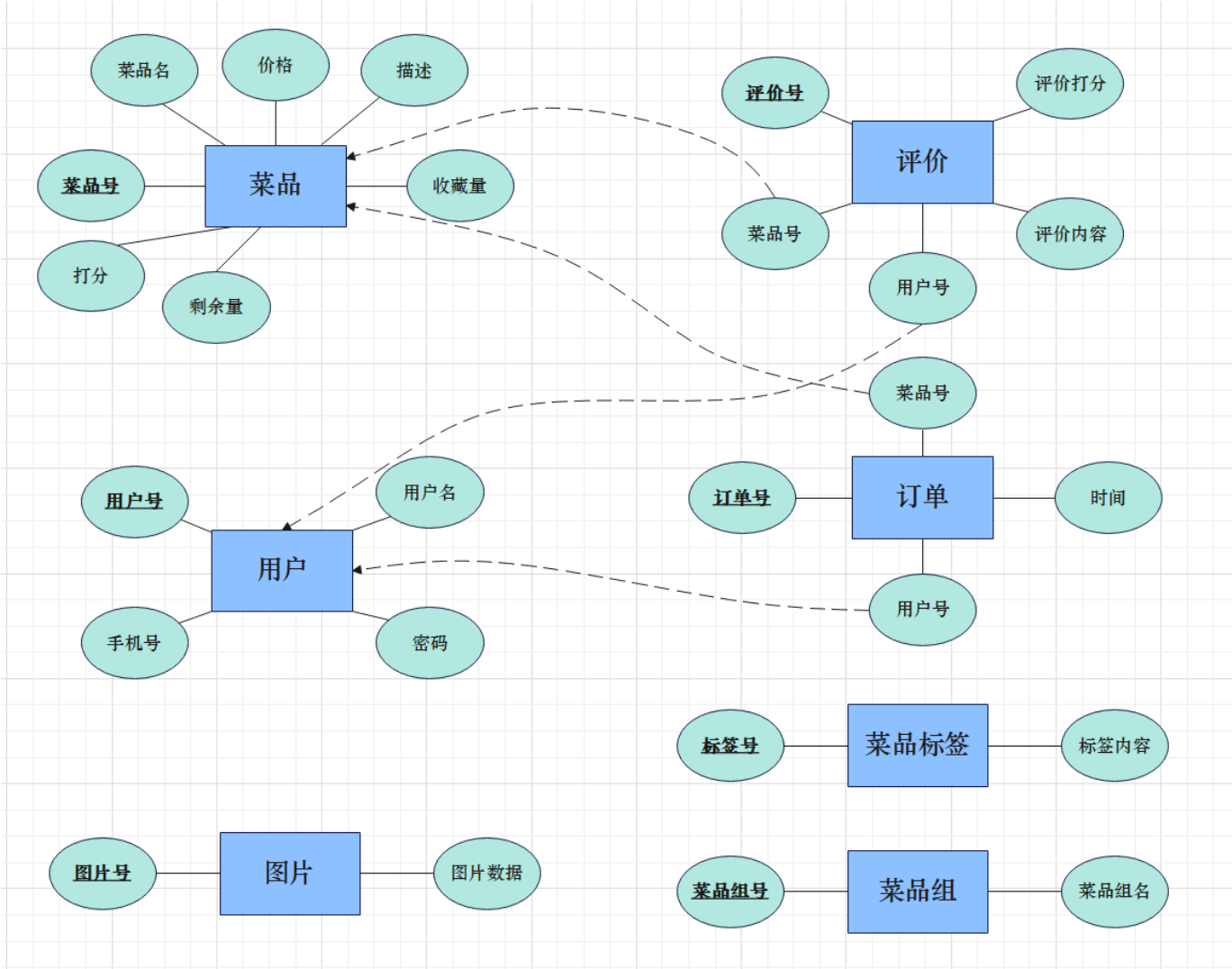
1.3.11 评价图片表(表名为CommentPicture)

字段	名称	类型	约束
Cid	Cid	char(8)	PRIMARY KEY & FOREIGN KEY
Pid	Pid	char(8)	PRIMARY KEY & FOREIGN KEY

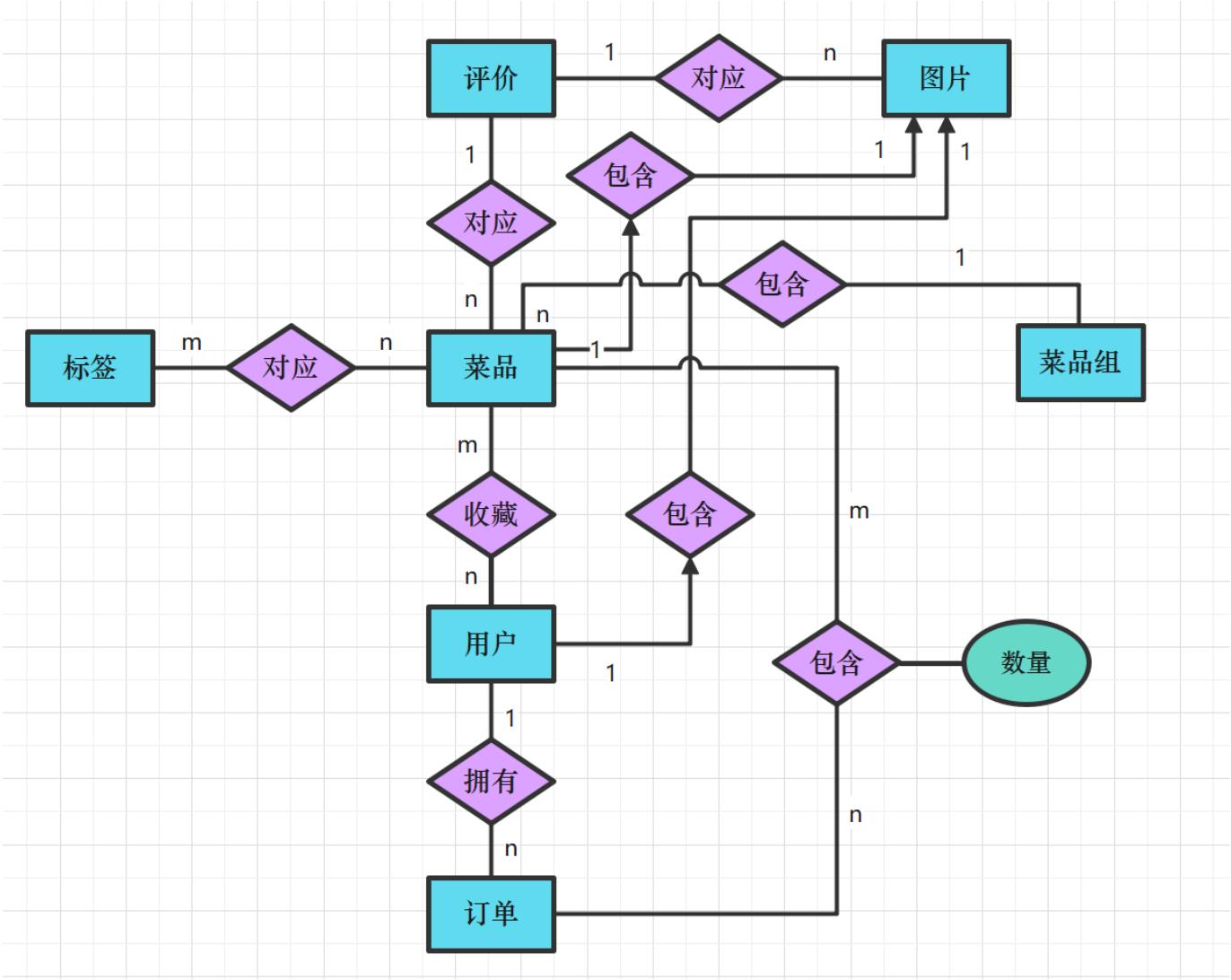
注释：菜品组和菜品标签有两点区别。其一，菜品组用于点菜界面分类查看不同菜品组中的菜品，菜品标签则无法在点菜界面作为分类标准查看不同标签的菜品。其二，一个菜品只属于一个菜品组，而一个菜品则可以同时拥有多样菜品标签。举例：菜品组(主食、炒菜)，菜品标签(甜、辣)

二、数据库概念模式设计

2.1 实体初步E-R图



2.2 系统初步E-R图



分析系统初步 E-R 图，我们发现可以消除以下 1:n 联系：

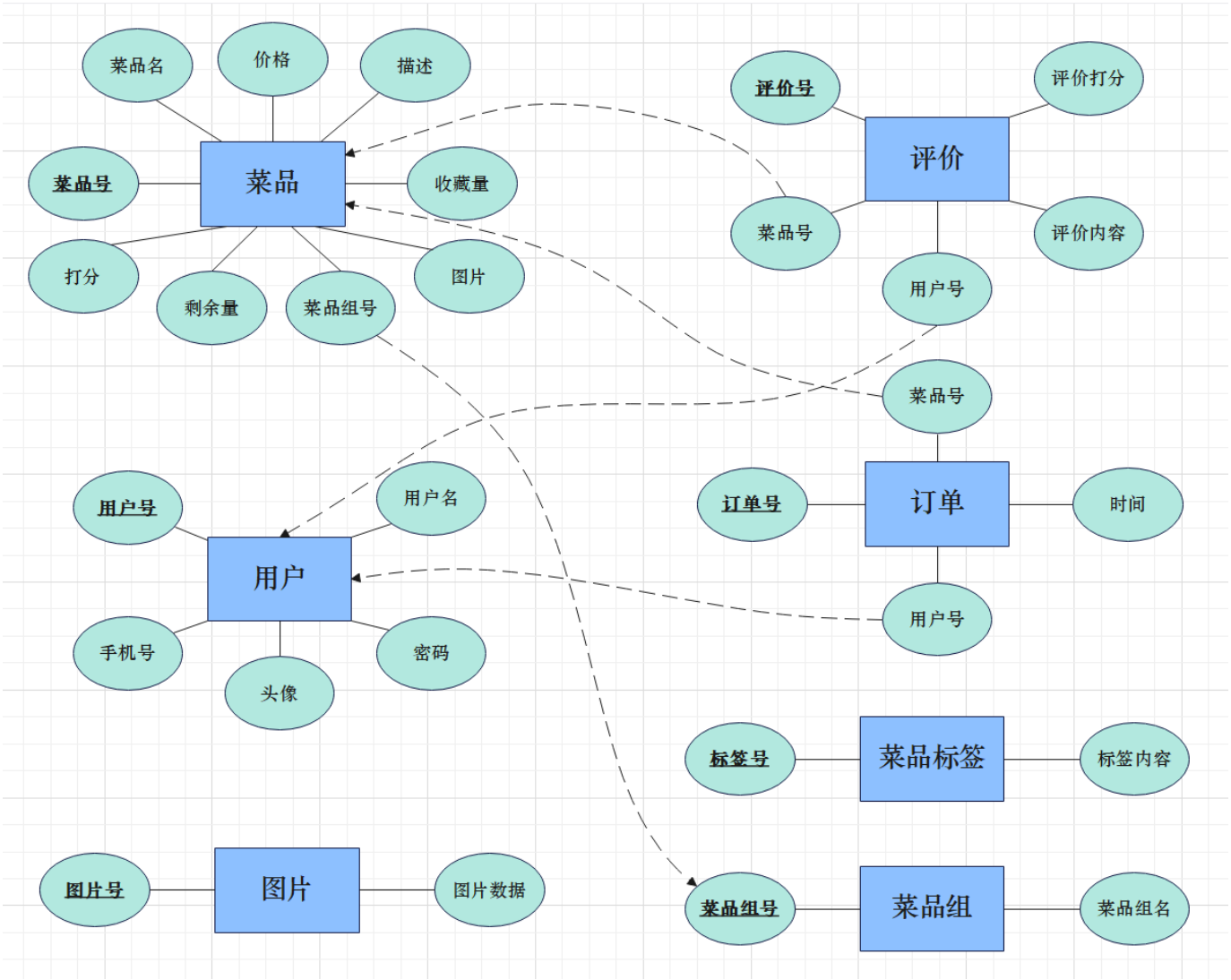
菜品组与菜品是 n:1 的关系，因此在实现的时候没有必要将菜品组和菜品联系表示出，将其作为菜品的个一个属性即可。

此外，我们还可以消除以下1:1联系：

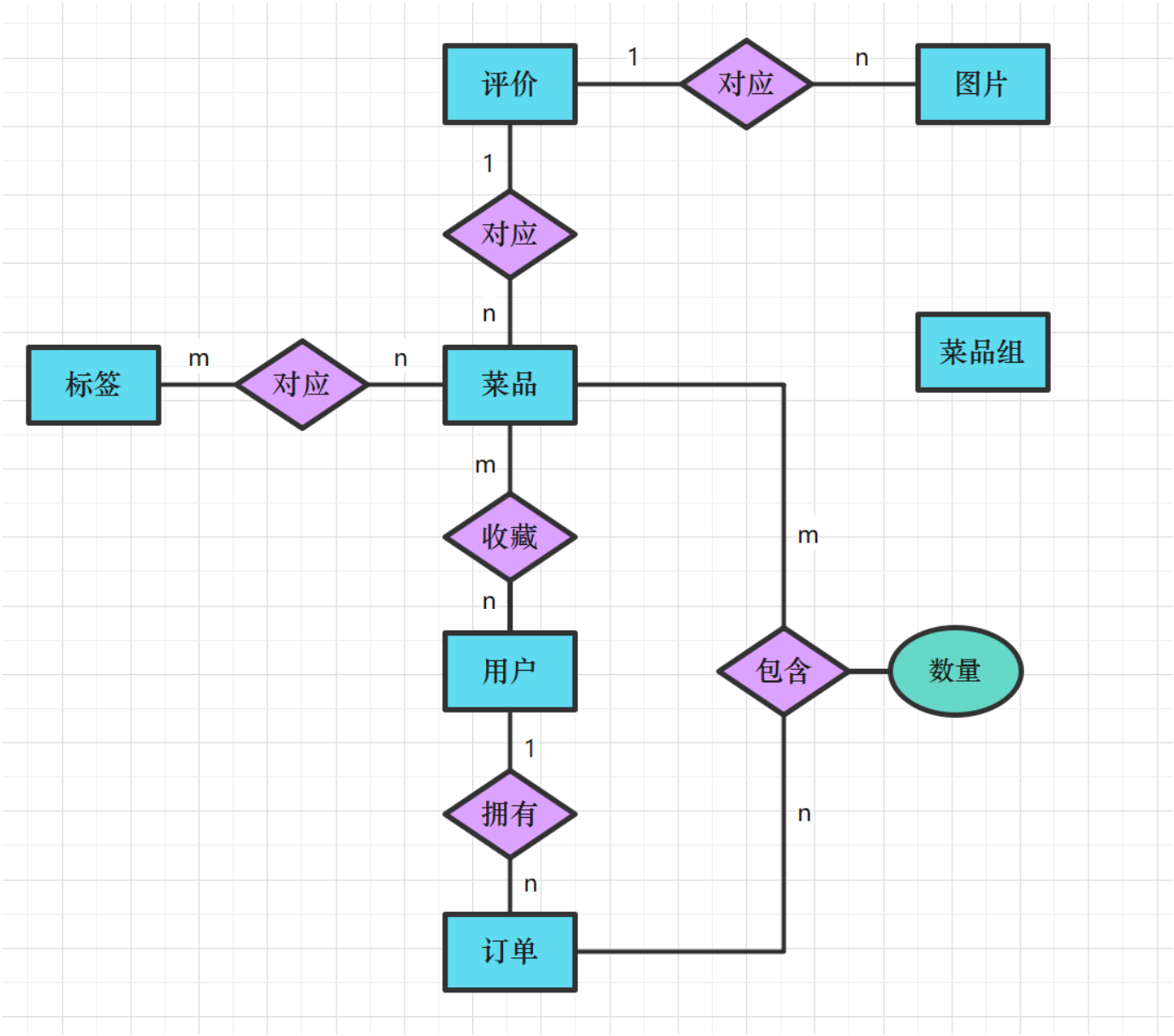
菜品和图片是1:1的关系，我们可以不表示菜品与图片的联系，将图片作为菜品的属性之一。

用户和图片是1:1的关系，因此我们可以不显示图片和用户的联系，将图片作为用户的属性之一。

2.3 实体基本E-R图


菜品名
价格
描述
菜品号
菜品
收藏量
打分
剩余量
菜品组号
图片
评价号
评价打分
评价
菜品号
评价内容
用户号
菜品号
订单号
订单
时间
用户号
用户号
用户名
用户
手机号
密码
头像
标签号
菜品标签
标签内容
图片号
图片
图片数据
菜品组号
菜品组
菜品组名

2.4 系统基本E-R图



三、数据库逻辑模式设计

3.1 数据库关系模式

- 用户(**用户id**、用户名、用户手机、用户密码、用户头像)
- 菜品表(**菜品id**、菜品名、菜品价格、菜品描述、菜品评分、菜品收藏量、菜品剩余、菜品图片、菜品菜品组)
- 菜品组表(**菜品组id**、菜品组名)
- 评价表(**评价id**、评论用户、评论菜品、评论内容、评论评分)
- 标签表(**标签id**、标签名)
- 订单表(**订单id**、订单用户、订单总价、下单时间)
- 图片表(**图片id**、图片数据)
- 菜品标签表(**菜品标签id**、**菜品id**)
- 用户收藏表(**用户id**、**菜品id**)
- 菜品订单表(**订单id**、**菜品id**、点菜数量)
- 评价图片表(**评价id**、**图片id**)

3.2关系模式等级的判定和规范化

- 用户(**用户id**、用户名、用户手机、用户密码、用户头像)

一个用户只有一个独立的用户id为主键，其他属性都依赖主键id，且非主属性之间没有函数依赖，因此肯定是3NF，所有的函数依赖的左值都含有码，因此是BCNF。

- 菜品表(**菜品id**、菜品名、菜品价格、菜品描述、菜品评分、菜品收藏量、菜品剩余、菜品图片、菜品菜品组)

每个菜品只有一个独立的菜品id为主键，没有其他候选码，一定是2NF，所有的非主属性都不传递依赖id，因此一定是BCNF。

- 菜品组表(**菜品组id**、菜品组名)

每个菜品组只有一个独立的菜品组id为主键，没有其他候选码，一定是2NF，所有的非主属性都不传递依赖id，因此一定是BCNF。

- 评价表(**评价id**、评论用户、评论菜品、评论内容、评论评分)

每个评价只有一个独立的评价id为主键，没有其他候选码，一定是2NF，所有的非主属性都不传递依赖id，因此一定是BCNF。

- 标签表(**标签id**、标签名)

每个标签只有一个独立的标签id为主键，没有其他候选码，一定是2NF，所有的非主属性都不传递依赖id，因此一定是BCNF。

- 订单表(**订单id**、订单用户、订单总价、下单时间)

每个订单只有一个独立的订单id为主键，没有其他候选码，一定是2NF，所有的非主属性都不传递依赖id，因此一定是BCNF。

- 图片表(**图片id**、图片数据)

每个图片只有一个独立的图片id为主键，没有其他候选码，一定是2NF，所有的非主属性都不传递依赖id，因此一定是BCNF。

- 菜品标签表(**菜品标签id**、**菜品id**)

菜品标签id和菜品id完全函数依赖，因此一定是BCNF。

- 用户收藏表(**用户id**、**菜品id**)

用户id和菜品id完全函数依赖，因此一定是BCNF。

- 菜品订单表(**订单id**、**菜品id**、点菜数量)

订单id和菜品id完全函数依赖，且点菜数量完全依赖前两个码，因此一定是BCNF

- 评价图片表(**评价id**、**图片id**)

评价和图片id完全函数依赖，因此一定是BCNF。

3.3 数据库设计优化

3.3.1 建立索引

通过建立适当的索引，我们对数据库查询性能进行了优化，重点关注了主码和外码字段。对于主码，我们建立了相应的索引，以提高频繁查询的效率，在主码包含多个字段的情况下我们也根据所有主码字段进行了索引创建。此外为了优化联表查询，我们对外码字段同样添加了索引。在考虑其他查询方式时，我们根据实际情况添加了额外的索引。这样的优化措施有助于提高整体查询性能，同时确保不过度增加索引的情况下获得最佳效果。

3.3.2 使用外键

我们在关系模式设计中使用外键以确保数据完整性，增强数据库的性能和可维护性。我们为每个表增加了可能的所有外键，确保了引用表与主表的数据一致性。同时，外键能够为关联查询起到潜在的加速作用，数据库引擎在检查到外键时可能会更快地执行查询，或者自动创建索引。此外，外键的设计将有助于数据库关系模型的可读性，有助于开发过程中的错误察觉。

3.3.3 关系模式简化

我们对关系模式进行了一定程度的简化，对于所有的 1 对 n 关系，我们都将某实体id设计为另一实体的属性，省去了额外增加一张表的开销，如将用户头像id设计为用户属性，将菜品图片id设计为菜品属性，将菜品组id设计为菜品属性等。对于与同一实体相关且可以合并的表，我们也进行了合并，如将菜品收藏、菜品剩余并入菜品表，在保证数据完整性的基础上简化了关系模式。

3.3.4 数据库字段类型选择

对于数据库中的字符常量，我们在最大限度内使用了定长的 char 类型，避免使用变长的 varchar 类型，一定程度上加快了数据库访问。对于图片存储，我们使用了 BLOB 类型存储图片的原数据，实现了在数据库中的统一管理和原子访问，保持了数据的一致性，同时在前端渲染上，BLOB 类型也有访问方便容易的优点。

3.3.5 优化查询语句

在进行数据库查询时，我们尽量选择用已建立索引的字段进行检索，同时避免使用嵌套查询，而是使用更加高效的联表查询。此外，我们在查询中避免使用 LIKE 关键字，防止产生较低的查询性能。最后，通过使用优秀的 ORM 框架，可以自动帮助我们处理查询语句的优化，进一步提高性能。