数据库课程设计报告

沈渐之 20307130030
毛柯宇 20307130241
刘子瑞 20307130196

June 6, 2022

1 简介与小组分工

本报告就本小组的物流公司系统项目，进行前瞻设定、需求分析、功能介绍、系统设计、数据库设计、开发平台框架、系统安装部署、系统使用、OceanBase 构建设想以及演示数据的说明汇报。

1.1 小组分工

沈渐之：题目设定构想、功能设计、用户活动图及数据流图设计、ER 图设计、数据库设计（表结构创建、索引添加、存储过程创建、函数创建、触发器创建）、用户账户系统实现、发件人收件人用户视图实现、驿站管理端实现、游客功能实现、前端分页功能嵌入、前端联动下拉框功能嵌入、前端模板优化、筛选功能嵌入、数据可视化（热力图）设计、OceanBase 部署尝试、OceanBase 构建构想、系统测试、期中 ppt 制作、期中期末视频录制、课程汇报撰写。

毛柯宇：功能设计（基于原系统优化完善）、数据字典、数据库设计（参与表结构设计、函数创建）、快递员页面的前后端实现、货车司机页面的前后端实现、员工视图实现、超级管理员大部分页面的前后端实现、功能实现（员工接单、人事管理增删改查实现）、前端模板优化、系统测试、期中期末 ppt 制作、期中期末视频录制、课程汇报撰写。

刘子瑞：功能设计如加密存储、系统逻辑层的完善、数据字典设计、中国行政区划地址收集导入、用户信息、员工信息、线路信息、包裹信息等测试数据集的随机生成与导入、期中部分 ppt 制作、视频录制。

2 前瞻与设定

2.1 行业前瞻

网购快递已经成为人们消费生活的重要组成部分，每天都有数以亿计的快递物流来往于全国，伴随着寄存柜、寄存驿站等新兴经济的兴起，每个环节都将产生、查询并管理大量的数据。如何合理地对这些数据进行处理，存放，利用在这个时代显得尤为重要。因此，数据库技术得以在这一领域大展拳脚。

本课题拟从一个快递公司的角度出发，分析快递公司在人事管理、用户管理和物流管理方面存在的各种各样的需求，并设计实现数据库以行使相关功能，以练习实践在本课程中学习到的数据库知识。

2.2 课题设定

我们在课题中先行对设计的数据库系统（快递物流系统）进行了简要刻画，对诸多方面进行了假设或者理想化。虽然我们涉及的数据库系统不能完全刻画现有的快递物流数据库系统，毕竟现实中快递物流操作的进行离不开一些实际的操作，如：核验包裹大小与重量、核验身份码取件等等，但是我们会用一些操作来替代这些实际操作来确保该快递物流数据库系统的完整性。

本课题的拟定主要分为以下的了 5 个方面：

1. 每条物流有相应的物流号、发件人、收件人、发出地、目的地、发货时间、包裹内容等信息。根据出发地点有承运的货车司机（理论上讲每两个城市之间会安排至少一位货车司机）与上门取件快递员，根据目的地有对应的暂存驿站、配送货快递员。
2. 发件人、收件人有对应的用户系统，需要先注册，系统会自动生成用户 id 和取件码。用户可以实时查看收发包裹的各种信息。有对应的员工系统，由快递公司统一录入并管理，系统会自动生成员工 id，用快递公司提供的初始密码登录后可以修改密码。各方均可通过手机和密码登录系统，快递员、货车司机、驿站管理的视图根据工作内容决定其查询权限。
3. 在发件人的用户视图中，可向某公司发起快递物流，在发件后查询有权限的相关信息。收件人有特定的身份码，在收件人的用户视图中，可以查看寄向自己包裹的各种信息。包裹到达菜鸟驿站后会实时通知收件人，之后可进行用身份码对应取件码取件。
4. 在暂存驿站中，物流根据规格有相应的货架和层号，优先放入当前件数最少的层，并生成取件码。包裹的寄存费用取决于存放时间。在驿站的管理系统中，实现输入包裹规格显示最合适位置，并在取件时结算寄存费用。
5. 快递公司管理系统可以录入并管理员工信息，发出上门取件单，录入包裹信息，修改人员信息与员工工作信息（如工作地点、薪资），计算补助与奖金。快递公司还提供游客视图，查看时效承诺配送时间与单价。

3 需求分析

3.1 需求描述与分析

近几年，快递行业迅速发展，本小组详细调查了物流管理系统的运行机制、参与人员以及行业发展。我们考察了使用物流管理服务用户的各种需求，详细研究了物流管理系统的各个环节的信息传递。

基于手机到的信息我们绘制了相应的数据流图，并整理了完备的数据字典，进而绘制 ER 图。这些工作将帮助我们构建物流管理系统的数据库以及基于此数据库的快递收发系统。根据各方面的需求，该系统期望实现：用户可以填写信息发起寄件单，物件在经过快递员上门取件、快递公司管理人员核验后成功发出，发件人与收件人均可随时查看物流信息。而工作人员根据其身份的不同拥有大相径庭的视图，每个工作人员有相对固定的工作区域，可以该区域进行接单，并可以随时查看已接单和历史接单。超级管理员则拥有最广的权限，可以管理工作人员信息、核验物流等。

3.2 用户活动及系统范围分析

我们对用户与员工的操作与活动进行了分析，根据此产生了用户活动图（业务流程图）。这些图会更形象地介绍每一个操作涉及的步骤，发生的信息传递。在此处我们将要举几个代表性的例子来进行说明。

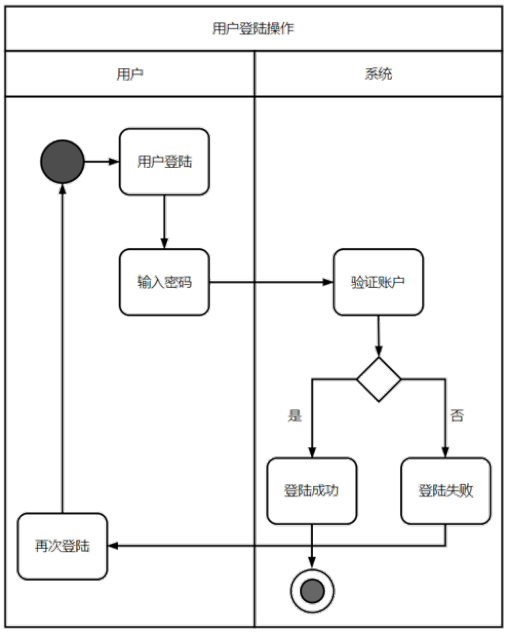


Figure 1: 用户登录操作

图1形象地表示了用户登录操作的流程。其中黑色实心圆圈表示一个操作的发起，而黑白相间的圆圈表示一个行为的结束。在这个登录操作中，用户需要输入账号与密码，系统会对账号与密

码进行验证并做出判定。若判定输入正确则登录成功，进入目标页面，否则登陆失败，需要重新进行操作。当然除了登录这种操作以外，我们不妨分析一些更复杂的（比如涉及多种结果的）操作，比如发起订单操作。

在发起订单操作中（见图2），首先由用户选择一个身份（身份在本系统中可以随意创建）发起订单，填写相关信息（包括包裹信息与收件方的信息）。完成之后会自动在系统中生成上门取件单，上门取件单根据地址呈现在相应地区的快递员的视图中。在快递员接单并进行上门取件之后，包裹将送至快递公司由管理员进行核验，此时会记录包裹的大小与重量。如果不可承运则取消此次物流，如果通过了核验则进一步创建订单，并更新物流状态。

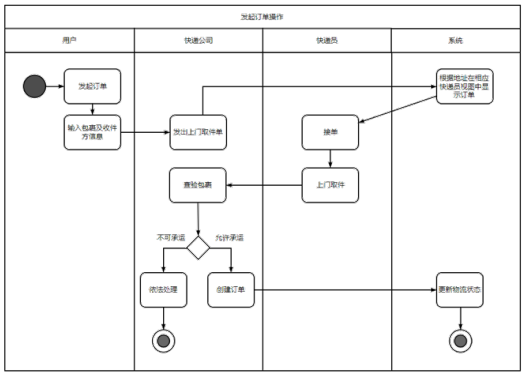
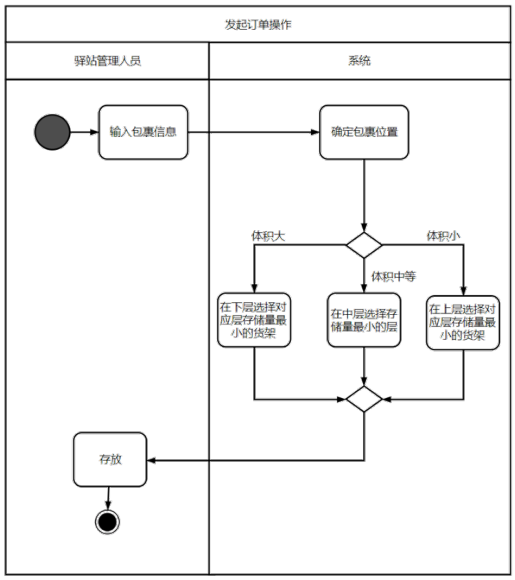
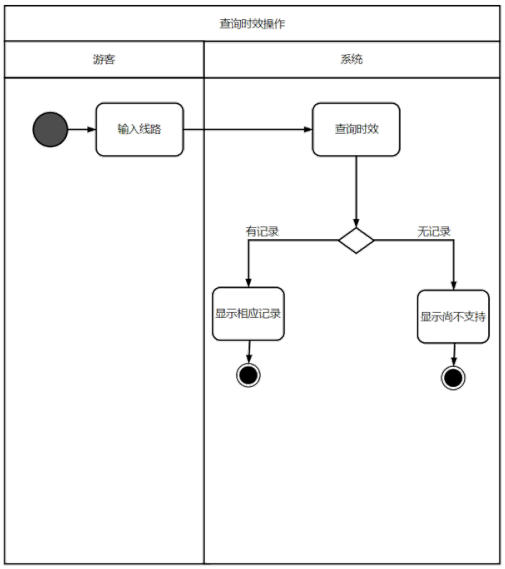


Figure 2: 发起订单操作（员工接单操作类似）

工作人员方面与无账号的游客方面也是类似，图3(a)中便简单地介绍了驿站管理员进行包裹分配的操作。首先输入包裹信息并确认包裹的位置，根据包裹的大小合理分配其在货架上的位置，这一步需要进行判定。而游客查询时效操作是所有查询操作的代表（类似的还有超级管理员通过工号查询员工），若有记录则返回查询结果，如无记录则提示其暂无相关数据。



(a) 驿站管理人员分配



(b) 游客查询时效

Figure 3: 用户活动及系统范围分析

3.3 数据流图

在经过了用户活动及系统范围分析之后我们可以统筹信息，将每一步数据的流动可视化表示，由此绘制出完备的数据流图（见图4）

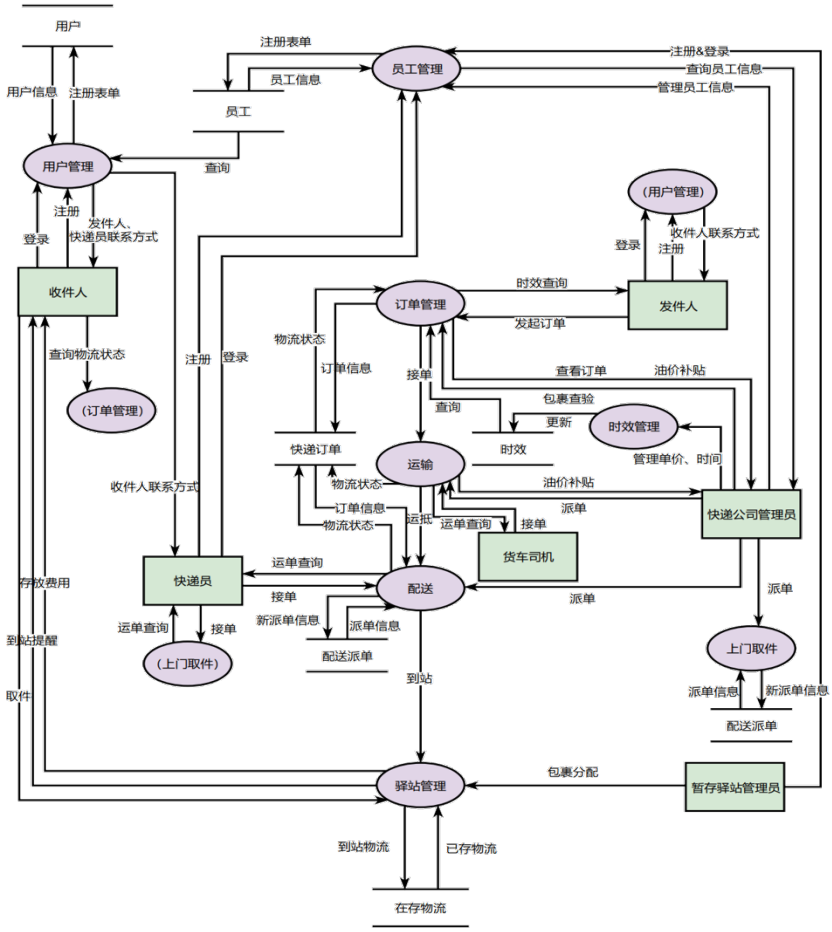


Figure 4: 数据流图

3.4 数据字典

数据字典是指以特定格式记录下来的、对系统数据流程图中各个基本要素（数据流、文件、加工等）的具体内容和特征所做的完整的定义和说明。它是结构化系统分析的另一重要工具，是对数据流程图的重要补充和注释。数据字典是进行详细的数据收集和数据分析所获得的主要成果，因此在数据库中占有很重要的地位。在本项目中，数据字典主要分为了以下几大部分。

3.4.1 数据项

3.4.2 数据结构简介

数据结构反映了数据之间的组合关系。一个数据结构可以由若干数据项组成，也可以由干个数据结构组成，或由若干数据项和数据结构混合组成。

1. 线路：由出发地和目的地组成，是快递的运输路线。每个地点又由市级和区级名称组成。
2. 时效：由时间和单价组成，是快递公司的服务承诺。
3. 存放位置：由货架号和层号组成，是包裹在某个驿站存放的位置。
4. 包裹规格：由重量和大小组成，是运费计量和位置分配的标准。
5. 用户信息：用户的 ID、手机、姓名、地址，是用户的相关信息。(员工信息类似)

3.4.3 数据流

数据流可以是数据项，也可以是数据结构，表示某一加工处理过程的输入或输出数据。在本次项目中，我们将主要存在的数据流总结如下：

3.4.4 数据储存

数据存储是处理过程中要存储的数据，可以是手工文档或手工凭单，也可以是计算机文档。在本次实验中，主要的数据储存归纳如下：

4 详细功能介绍

在引入到系统设计之前，我们需要先详细地介绍我们系统实现的功能。功能部分主要分为以下四个部分：用户方、快递公司员工方、快递公司管理员方（超级管理员）以及游客方。

4.1 用户方

用户方的功能主要如下：

1. 用户注册：系统储存注册信息（用户需要填写手机号、密码、姓名、地址等信息），系统会自动生成用户 id 与取件码；
2. 用户登录：用户输入注册时使用的手机号与设定的密码，系统验证账号与密码后进入用户视图界面；

数据流名	寄件信息表
说明	发件人联系快递公司并进行寄件
数据流来源	寄件管理人员
数据流去向	快递公司承运系统（物流系统）
数据组成	用户信息+寄件费用+物流号+收件人信息
平均流量	1000件/天
高峰流量	3000件/天

(a) 由发件人到快递公司的信息传递

数据流名称	派件安排表
说明	快递公司安排目的地的快递员进行派件
数据流来源	快递公司承运系统
数据流去向	快递公司派单系统
数据组成	物流号+驿站地址+快递员工号
平均流量	2000件/天
高峰流量	3000件/天

(b) 由快递公司到快递员信息传递

数据流名称	包裹信息录入
说明	驿站管理员进行信息录入
数据流来源	驿站管理员
数据流去向	在存物流
数据组成	物流号+入站时间+包裹编号
平均流量	2000件/天
高峰流量	3000件/天

(c) 驿站管理员向驿站录入信息

数据流名称	驿站取件
说明	收件人凭身份码进行取件
收件人方数据	身份码取件+寄存费用
驿站方数据	取出时间+计算寄存费用+库存中移除包裹与编号
平均流量	1500件/天
高峰流量	2500件/天

(d) 收件人前往驿站进行取件

Figure 5: 数据字典–数据流

3. 身份创建与管理：用户可以任意创建、编辑身份，寄件与收件的对象以用户创建的身份为目标。创建身份后自动生成独立于用户 id 的 iid。一个用户可以同时拥有多个身份；

4. 寄件：用户可凭借其创建的身份向某快递公司发起快递物流，需要填写寄件人对应身份的地址与手机号（或其他信息），并支付系统计算出的运费。物流在通过快递公司管理员审查后成功发起，寄件发起后可以查询加密的收件人的联系方式、物流状态等信息。在查询界面可以输入物流号对物件进行筛查；

5. 收件：包裹抵达驿站之后生成对应取件码（取件码格式会表示所在的货架位置）。收件人利用身份码对取件进行确认并支付寄存费用。收件人可以查询加密的发件人联系方式、包裹信息（规格与内容）、快递员联系方式、物流状态、取件码、寄存费用（寄存费用会从到件开始计算）等信息。在查询界面可以输入物流号对物件进行筛查。

4.2 快递公司员工方

首先，所有员工的账号由快递公司管理员分配而不是自行申请，员工账号拥有统一的初始密码，可在登录后进行修改。

数据存储名	用户
说明	用户的身份信息及登录信息
输入数据流	用户向系统注册
输出数据流	用户向系统查询
数据组成	用户信息，密码
数据量	1000
存取频度	100次/天
存取方式	联机处理、检索且更新、随机检索

(a) 用户

数据存储名	订单
说明	物流信息
输入数据流	发件人发起、员工接单
输出数据流	用户查询、员工查询
数据组成	物流号、线路、规格、内容、物流状态等
数据量	10000
存取频度	1000次/天
存取方式	联机处理、检索且更新、随机检索

(b) 订单

数据存储名	在存物流
说明	某驿站存有的包裹信息
输入数据流	驿站管理员录入到站物流
输出数据流	收件人取件
数据组成	物流号、在存位置、存放时间等
数据量	500
存取频度	50次/天
存取方式	联机处理、检索且更新、随机检索

(c) 在存物流

数据存储名	时效
说明	线路和承诺时间、单价的对应信息
输入数据流	快递公司物流管理向系统输入
输出数据流	用户（游客）向系统查询
数据组成	线路、时间、单价
数据量	500
存取频度	50次/天
存取方式	联机处理、检索且更新、随机检索

(d) 时效

Figure 6: 数据字典–数据储存

4.2.1 快递员方

快递员页面的主要功能如下：

1. 快递员可以查看自己已经接的正在进行中的单，通过地区匹配查看其所在地能够接的单，以及历史接单（当该物流的状态进入下一个阶段后视为其已完成工作，进而从正在进行中的单中消失，加入到历史接单中）；
2. 快递员可以在接单页面继续接单，一旦点击接单便会更新该物流的对应信息（比如对应的快递员 id），接了的单不能删除、退回。接单后会实时更新快递员已接单信息与其（未领取的）奖金；
3. 快递员可以查看自己的个人信息与员工信息，可以查看已接单数与实时奖金。奖金是关于已接单数的函数，但是会在领取之后清零，因此两者在样本中没有固定的关系。

4.2.2 货车司机方

货车司机页面的主要功能如下：

1. 货车司机可以查看自己已经接的正在进行中的单，通过地区匹配查看其所在地能够接的单，以及历史接单（判定与快递员相同）。区别于快递员的是货车司机往往是往返于两地的，因此其可接单所在的地域是有变化的；

2. 货车司机可以在接单页面继续接单，一旦点击接单便会更新该物流的对应信息（比如对应的货车司机 id），接了的单不能删除、退回。接单后会实时更新货车司机已接单与其（未领取的）油价补贴；

   （补充：货车司机接单在现实中是不存在的，我们将“对包裹扫码登记并装载”这一行为进行了抽象处理以实现系统的完整性）

3. 货车司机在到达指定目的地后需要手动更新其所在地。在规定中货车司机来往于两地，因此这个操作会交换其出发地与目的地。其可以在当前位置继续接单。

4. 货车司机同样可以查看个人信息与员工信息，可以查询实时油价补贴。

4.2.3 驿站管理员方

驿站管理员页面的主要功能如下：

1. 驿站管理员可以查看到站的报告，点击“分配位置”按钮系统会自动为包裹分配合适的位置。需要驿站管理员手动将该包裹放置到指定位置。

2. 驿站管理员系统同时也是收件人进行取件的接口。收件人取件时，需要在点击对应包裹后输入自己的身份码，经过核验后成功取出。

   补充：取件过程仍然是对现实操作的一个抽象，具体的描述请见系统设计部分。

4.3 快递公司管理员方（超级管理员）

4.3.1 原本实现

快递公司管理员的账号由快递公司专门提供，其不能修改密码与账号。快递公司管理员是一个拥有各种权限的高级账号，由多个人同时使用，因此没有相应的个人信息。

快递公司管理员页面的主要功能如下：

1. 快递公司管理员可以查看所有员工的所有信息，可以通过工号对指定员工进行查询；

2. 快递公司管理员可以添加、修改、删除员工信息。添加界面需要明确员工的类型、工作区域等信息，添加完成后会自动在数据库的员工登录对应的表格中添加新的账号，该账号拥有默认密码。快递公司管理员没有权限查看、修改员工的密码。删除操作进行后，也会同时删除登录表格中对应的员工信息，被删除账号便不能再登录，需要重新申请。

3. 快递公司管理员可以查看物流的所有信息，可以通过物流号查询相关物流。在用户发起的寄件经过快递员上门取件操作之后需要管理员对包裹进行核验。通过核验的包裹需要记录大小、重量等信息，记录后可以被货车司机接单。未通过核验的包裹（违禁品）会被删除，会进一步联系发件人或直接会交予相关部门处理。快递公司管理员没有权限对包裹的内容等用户填写的信息进行修改；

4. 快递公司管理员系统会提供热力图供管理员直观地了解近期各地的物流量等信息。

4.3.2 汇报后改进

在完成了报告之后，我们小组认识到本系统，尤其是是在快递公司管理员方，仍在功能上存在可以进一步优化的地方。根据老师在汇报中对我们提出的建议、并综合考量了一些其他组同学的优良之处之后我们对超管系统进行进一步的改进。

首先，我们在快递公司管理员的人事管理界面中完善了快递员与货车司机的“结算功能”。在我们的假设中，快递员的“奖金”与货车司机的“油价补贴”均是随接单数发生变化的变量。当该员工（快递员或货车司机）进行了奖金或油价补贴领取之后，奖金或油价补贴会被清零。之前之歌“清零”的操作被我们抽象化为线下结算，但是这样的假设是不合理的，因此我们在相关界面增设了“结算”的选项。点击“结算”并点击额“确定”则会将该名员工对应的奖金或油价补贴清零。

其次，在老师的建议下，我们进一步完善了物流的检索功能。我们先将对于物流号得精确检索独立出来，用于单个包裹及其相关信息进行精确搜索。进一步，我们设定完备的筛选功能，使用者（即快递公司的管理员）可以在以下选项中及进行选择：

1. 设定一个具体的位置（要求为城市 + 区，eg：上海市-虹口区，不选择时默认为“不限制”，eg：上海市-不限制）
2. 进一步对限制这个地址的性质为出发地或者是到达地（不选择时为“不限制”）
3. 对物流的状态信息进行限制，比如可以设置为只查看状态为“运输中”的包裹

管理员可以对该三种限制方式进行组合使用（没有被使用的选项会默认为“无限制”），对需要统一查看的包裹进行筛选，分类。这样的处理方式不需要超级管理员在查询某地相关包裹时记忆复杂的行政区划代码，大大地提升了快递公司管理员在处理物流信息时的效率。

4.4 游客方

游客方的功能不需要使用者拥有任何账号，在系统的主页面可以直接进入。因此无论有无账号均可以使用该页面的功能。

游客方主要提供的功能是时效查询与驿站查询。在驿站查询功能中，选择下拉框中对应的地址可以查询该地区的驿站详细地址。理论上讲每一个区分配一个驿站。而在时效查询功能中，用户需要选择两地，系统会给出两地寄件的单价与预估时效。预估时效的大小基本取决于两地的之间的距离，距离越长承诺的配送时间也就相应更长。

5 系统设计

系统组成请见 Figure7。

5.1 用户账户系统

功能：用户账户系统包含用户注册、登录、修改密码等功能。

实现：UserLogin 表储存了每个用户的 ID、手机和 MD5 加密后的密码，通过 Django 的 hashers 模块实现密码的加密储存与检验。

5.2 员工账户系统

功能：员工账户系统包含员工登录、修改密码、人事管理功能（新员工的创建、快递员奖金与货车司机补贴的查询）。

实现：StaffLogin 表储存了每个员工的手机、MD5 密码、类型和编号。每个类型的员工都有一张表（Courier、Driver、Manager），记录了相应的人事信息，快递员的奖金、货车司机的油价补贴在每次接单时会更新。

5.3 物流管理系统

功能：物流管理系统可以实现整个快递业务流程的可视化管理。包括快递员上门取件接派单、物流管理部门核验包裹、货车司机运输接派单、快递员配送接派单、驿站寄存、核码取件。

实现：快递员可以查询 Package 表中有关自己所负责区域的订单，并选择接单。货车司机同理，但是考虑到实际人为勾选操作非常低效，本系统仅提供了一个接口，在真实生产环境中，可以通过扫描包裹二维码实现快速接单。快递员的接单会产生高并发，本项目采用以下策略：Package 的 “status”（包裹状态）列作为乐观锁，在快递员接单的事务中，若检测到 “status” 已变化（说明已被接单），则系统会提示错误并停止更新。物流管理部门核验包裹中录入包裹规格数据的过程也是自动化接口，生产环境中可通过机器扫描和称量实现自动输入，在核验包裹后，通过 Service 表中对应的单价和重量生成运费。驿站寄存通过 Store 表记录驿站每架每层存放数目，实现自动分配至当前负载最小的层。该功能由存储过程实现。Package 表上创建的触发器可以自动更新 Store 表。驿站存放时会随机生成身份码和带有架号层号的取件码，会显示在收件人界面，收件人可凭取件码找到包裹，并凭身份码进行取件。每个过程对 Package 表进行更新的规则是：

发件人发起订单时：新建记录。pid 物流号,content 包裹内容，sender_id 发件人用户 ID，sender_iid 发件人身份 ID（身份是发件人自定义的用户信息）,receiver_id 收件人用户 ID，receiver_iid 收件人身份 ID，start_time 开始时间，pdeparture 包裹出发地编号，pdestination 包裹目的地编号，expected_time 预计时间，station_id 目的地对应驿站编号，status 此时为 “未发货”；

快递员接上门取件单时：courier_a_id 上门取件快递员 ID，send_time 发货时间，status 更新为 “已发货”；

管理员核验时：weight 包裹重量，size 包裹体积，express_price 运费，status 更新为 “已接单”；

货车司机接单：driver_id 货车司机 ID，status 更新为 “运输中”；

快递员接配送单：courier_b_id 配送快递员 ID，status 更新为“配送中”；

驿站存放：shelf 货架号，layor 层号，arrival_time 到站时间，picker_id 身份码，status 更新为“已到站”；

收件人取件：pick_time 取件时间，station_price 存放价格，status 更新为“已收货。订单结束。

结束超过两个月的订单会被自动清理。

5.4 物流服务系统

功能：用户创建和管理身份，收件人和发件人查看包裹相关信息，物流号查询，发起订单，游客查询地区驿站信息和各路线时效信息。

实现：考虑到用户收发包裹可能有多个地址和多个身份，本系统引入 Identity 表来存储用户创建的个人身份信息，包括称呼、可与登录手机不同的联系方式、地址。在发起订单时，只需填写收发双方的身份标识即可。这些身份信息可以被与该订单相关的其他用户或员工所查询，用户也可查到相应的快递员信息，因此需要复杂的联表查询，本系统通过存储过程实现。游客查询地区驿站信息和各线路时效信息，则通过 Station 驿站信息表、Service 时效信息表和 Map 地区表的连接。各表将在下一板块数据库设计详细说明。

6 数据库设计

数据库端共设计了 Package, User_Login, Identity, Staff_Login, Courier, Driver, Manager, Station, Store, Map, Service 十一个表，设置了 ‘time_price‘函数。创建了 ‘receiver_view‘, ‘sender_view‘, ‘service_map‘, ‘station_arrangement‘以及 ‘station_pricing‘五个存储过程。其中前两个实现的其实是仅查询含参视图，其他查询过程则较为简单，故不再单独设置视图。因为考虑到外键的性能影响，有关外键的完整性约束均通过应用层实现，数据库没有实际创建外键约束。此外，由于本系统在实际生产环境下查询的需求量远大于增删，因此给频繁作为查询条件的列如位置编码均添加了索引。在 Package 表上有触发器 ‘store_count‘。事务则主要体现在快递员接单的功能中，结合 Django 的 Transaction 模块，利用乐观锁来避免接单冲突。接下来，将详细介绍各表结构、存储过程及函数。相关 SQL 代码请见附件。数据库逻辑 ER 图请见 Figure8。

Package 表：由于订单实体和本系统涉及的其他所有实体几乎都有 1 对 n 的关系，因此 Package 表几乎囊括了订单流程的所有信息：各方 ID (**id)、各节点时间 (**time)、各价格 (**price) 以及与驿站存放相关的货架号、层号、取件码和身份码。在逻辑上，Package 表的各 ID 和位置编码与各自相关联的表都有参照完整性约束。这在 ER 图上有所体现。由于均通过原始关系模型按公式转化而来，尽管列数繁多，仍满足三范式。

User_Login 表：记录用户登录的凭证。每个用户有一个用于登录的手机、经过 MD5 加密的密码以及主键用户 ID（注意用户 ID 与身份 IID 的区别，一个用户可以创建不同的身份用于创建订单，因此用户与身份的关系是 1 对 n）。手机应满足唯一性约束，同时由于其频繁查询，创建了索引。

Identity 表：Identity 表存放了所有用户添加的身份信息，普通用户有对该表中各自用户 ID 对应的身份信息的操作权限。单个用户创建的身份标识 (iset) 应不同，由于 uid 作为查询条件更为频繁，根据 MySQL 的最左匹配原则，Identity 表创建了 (uid,iset) 唯一性索引。所在地 ilocation 与地图表的 ID 有参照完整性约束，联系方式 iphone 则不再有唯一约束和域限制，仅作为查询对象。

Staff_Login 表：与 User_Login 表相同，不过由员工类型（class）与对应类型的工号（sid）作为联合主键。

Courier 表、Driver 表、Manager 表：记录各员工信息，其中涉及的位置编码均与 Map 表的 ID 有参照完整性约束。cmoney、salary 分别是快递员的固定工资和奖金，dsalary 和 dcompensation 分别是货车司机的固定工资和油价补贴。Manage 表的 station_id 与 Station 表的 sid 有参照完整性约束。

Station 表、Store 表：各驿站信息和存放情况。slocation 也与 Map 表相关联。Store 表的 station_id 与 Station 的 sid 有参照性约束。

Map 表、Service 表：记录了快递公司的业务范围和各路线时效信息，是整个业务的基础。Service 表每条路线的信息可由 Map 表两地经纬度数据函数导出。

‘receiver_view‘存储过程、‘sender_view‘存储过程：输入相应的用户 ID，收件人视图可以查看该用户所收快件的时间、物流状态、快递员联系方式、寄存驿站等信息，发件人视图可以查看该用户所发快件的时间、物流状态、快递员联系方式、运费等信息，两个查询都相对复杂，又需要相应参数，而无需更新操作，因此创建了存储过程来实现。

‘service_map‘存储过程：可以根据 Map 表中两地的经纬度数据生成 Service 表中的预计时间 duration、运费单价 unit_price 数据。

‘station_arrangement‘存储过程：输入驿站编号 sid，包裹编号 pid，根据包裹规格及驿站存放情况，自动将包裹分配至合适的当前存放量最小的层，并更新包裹寄存信息。

‘station_pricing‘存储过程与 ‘time_price‘函数：后者可以根据输入的到站时间和取件时间计算寄存费用，从 12 小时开始计费，每两小时一元，最高五元。前者使用后者来实现取件过程。

‘store_count‘触发器：当 Package 的列 layor、shelf 改变时，识别存放与取出并调整 Store 表对应层的 num 值，实现 Store 表实时更新。

7 开发平台及框架

开发平台为 Python，采用 Django 框架。由于其内置的 ORM 框架能够快速地对数据库进行基础的交互操作，且可以通过 Transaction 类进行高效的事务控制，对本项目满足高响应、高并发的需求有非常大的帮助。

数据库平台采用的是 MySQL。

8 系统安装部署说明

8.1 实验环境

环境需求：需要安装 Python 与 MySQL 的最新版本。Python 需要安装 Django,folium,pandas 宏包。数据库搭建：在 MySQL 平台运行数据库文件夹中的 sql 文件，创建数据库‘express’的表结构，通过 csv 导入模拟数据。也可直接将 express 表复制到本地 MySQL 数据文件夹中并重启 MySQL 服务。预设调整：在路径“/物流公司系统/物流公司系统”下的 setting.py 文件中，将 DATABASES 下的’USER’、’PASSWORD’、’HOST’ 及’PORT’ 改为运行主机的 MySQL 实例配置（账户要有 express 表的所有权限）。TIME_ZONE 的属性改为公司当前所在地。在路径“/物流公司系统/app01”下的 views.py 文件中，将视图 depaheatmap 和 destheatmap 下的 file_path 改为运行主机该项目的 templates 路径。 服务器启动：在 Python 平台运行 Django 项目‘物流公司系统’，或在物流公司系统项目下输入指令’python manage.py runserver ‘，便可部署系统。默认的 IP 地址为 127.0.0.1，端口为 8000。

8.2 生产环境

环境准备：以 linux 系统为例，除实验环境要求外，还需装有 nginx，git。通过命令”pip3 install uwsgi” 安装 uwsgi，并在项目文件夹下创建 ini 文件：

```
# myannounce_uwsgi.ini file
[uwsgi]

# Django-related settings

socket = :8000  #服务端口号

# the base directory (full path)
chdir = ~/物流公司系统/ #服务器项目路径

home = ~/物流公司系统/venv/  #服务器虚拟环境路径
PYTHONHOME = ~/物流公司系统/venv/

# Django s wsgi file
module  = announce.wsgi

# process-related settings
# master
master = true

# maximum number of worker processes
processes = 4
buffer-size = 65536

# ... with appropriate permissions - may be needed
# chmod-socket    = 664
# clear environment on exit
vacuum = true
```

并通过命令“uwsgi –ini 该文件名”进行配置。配置 nginx.conf

```
user nginx;
worker_processes auto;
```

```
error_log /var/log/nginx/error.log;
pid /run/nginx.pid;

# Load dynamic modules. See /usr/share/nginx/README.dynamic.
include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile            on;
    tcp_nopush          on;
    tcp_nodelay         on;
    keepalive_timeout   65;
    types_hash_max_size 2048;

    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

server {
    listen 80;
    server_name localhost;
    charset     utf-8;
    access_log      /var/log/nginx/nginx_access.log main;
    error_log       /var/log/nginx/nginx_error.log;
    client_max_body_size 75M;


    location /static {
        alias /djproject/mysite/static/ckeditor/;  #指定django的静态文件
        }

    location / {
        include     /etc/nginx/uwsgi_params;  #加载uwsgi模块
        uwsgi_pass  127.0.0.1:8000;    #所有请求转到8000端口交给uwsgi处理
        }
    }
}
```

启动 nginx 服务。

数据库搭建、预设调整：将 setting.py 中 debug 属性改为 False，其他与实验环境相同。

9 系统使用说明

9.1 普通用户

注册时只需要提供手机和密码，手机需要在三大运营商的号码范围内，密码需要满足至少八位、含大小写字母和数字、无特殊符号。登录后可以在右上角找到修改密码的入口，密码设置的限制是相同的。

在发件前需要先创建身份信息，先设置一个身份标识，再输入称呼、联系方式（此处只要能够联系即可）、地址，选择地区，创建新身份时身份标识不能重复。

发件时，只需要输入包裹内容、发件的身份标识，以及通过联系收件人或接入第三方平台，输入收件人的身份标识和对应的手机即可。运费会在快递公司核验之后自动生成，可以先通过时效查询查询相应的单价和预计时间。

收件时，在包裹状态更新为“已到站”后，会显示包裹的取件码和身份码，以及驿站的地址，也可通过驿站查询查询相应驿站的地址和联系方式。取件码的前两个数是对应包裹存放的货架号和层号，通过取件码找到相应包裹，在驿站系统中输入身份码，即可取出包裹。在取出包裹之后，存放价格会自动更新。

9.2 快递员

快递员可以在“已接单”页面看到自己接的且正在进行中的单，可以查看取件类型，用户电话以及发起时间等信息，方便其执行相关工作。但一单物流结束之后会从“已接单”中移除，来到右上角的历史接单中。同时右上角还提供查看个人信息、修改密码的功能，对于个人信息快递员的权限为“只读”。

除此之外，快递员可以再“继续接单”页面进行继续接单。其中显示出来的物流均根据快递员自身的工作区域筛选得到。一旦进行接单则不能回退该单物流。

9.3 货车司机

货运司机与快递员功能大同小异，主要区别有两点。首先货运司机查看权限不同于快递员，由于货运司机默认来往于两地，因此不需要查看物流的地址信息，查看的时间则为“交付运输时间”而并非“发起时间”。

除此以外，货运司机多了一项功能“修改所在地”。当货运司机到达规定目的地后需要手动输入目的地信息实现目的地与出发地的互换。这个操作也会导致“继续接单”界面的可接单内容发生改变。

9.4 驿站管理员

驿站管理员可以在“待存放”页面看到即将存放于所管理驿站的包裹，在包裹到达驿站后，可点击“存放”，系统会自动显示应该存放的位置，此时系统已经默认包裹存于该位置。

包裹可以通过物流号查询寄存位置、规格等信息。

9.5 人事管理

在人事管理界面，可以对所有快递员、货车司机与驿站管理员的信息进行增删改查。首先在页面上方可以通过指定工号对于特定的员工进行查询。对于每一个员工，其可以执行三个操作：编辑、删除与结算。

首先，在编辑页面可以对一些员工的基本信息进行编辑，比如员工的薪资、性别、工作区域等等。但是诸如员工的工作类型、员工的奖金或补贴是没有权限更改的，仅有查看权限。而对于员工的密码等隐私信息是没有查看权限的。

其次，删除操作会直接删除该员工的信息，并且在登录的表格中也将其账号删去。删除操作不可逆，请谨慎使用。

接着，结算操作仅针对快递员与货车司机。在进行了结算操作之后会直接将其奖金或油价补贴清零，从头开始新一轮的计算。

超级管理员还能够新增人员，需要填写一些员工的基本信息，比如类型、性别、工作区域以及暂定薪资等等，完成后点击添加。该步骤不仅往员工的表格添加了人员，也向员工登录的表格中添加了相关类容。即该员工可以使用超管添加人员中写的手机号以及默认密码（默认为“123456”）进行登录了。

9.6 物流管理

超级管理员可以对所有物流的相关信息进行查询，可以通过目的地、出发地以及包裹状态进行筛选查看。在“待核验包裹”页面显示的是物流状态为“已发货”的包裹。在包裹到达中转站之后，可以对包裹进行核验与称量，若核验通过就输入规格信息接单，否则可以选择“异常”，交由有关部门处理。

页面右上角有热力图查询的入口，系统能够根据发出包裹数量或接收包裹数量实时生成可视化热力图，来显示各地快递分布情况。

10 基于 Ocean Base 构建设想

由于本项目实施的条件有限，无法成功建立功能齐全的 Ocean Base 三副本集群，因此无法有效引入 Ocean Base 分布式数据库。但已经尝试了 Ocean Base 基本的部署实践，做出了基于 Ocean Base 的构建设想。

10.1 Package 表分区

在实际的生产场景下，随着时间的推移和各地订单的创建，Package 表将存储海量数据。本项目初步的处理方法是定期清空距离结束超过两个月的订单数据，这就无法满足订单历史查询的用户需求。在引入分布式数据库后，可以对 Package 表进行二级分区：基于目的地进行 Hash 分区，基于时间进行 Range 分区。具体代码实现：

```
CREATE TABLE `package` (
  `pid` int NOT NULL AUTO_INCREMENT COMMENT '物流号',
  `weight` int DEFAULT NULL COMMENT '包裹重量',
  `size` int DEFAULT NULL COMMENT '包裹体积',
  `content` varchar(16) NOT NULL COMMENT '包裹内容',
  `sender_id` int NOT NULL COMMENT '发件人ID',
  `receiver_id` int NOT NULL COMMENT '收件人ID',
  `pdeparture` varchar(16) NOT NULL COMMENT '包裹出发地',
  `pdestination` varchar(16) NOT NULL COMMENT '包裹目的地',
  `start_time` datetime NOT NULL COMMENT '创建时间',
  `courier_a_id` int DEFAULT NULL COMMENT '上门取件快递员工号',
  `send_time` datetime DEFAULT NULL COMMENT '发货时间',
  `driver_id` int DEFAULT NULL COMMENT '货车司机工号',
  `courier_b_id` int DEFAULT NULL COMMENT '配送快递员工号',
  `station_id` int DEFAULT NULL COMMENT '寄存驿站编号',
  `arrival_time` datetime DEFAULT NULL COMMENT '到站时间',
  `shelf` int DEFAULT NULL COMMENT '存放货架号',
  `layor` int DEFAULT NULL COMMENT '存放层号',
  `pick_id` varchar(10) DEFAULT NULL COMMENT '取件码',
  `picker_id` char(11) DEFAULT NULL COMMENT '身份码',
  `pick_time` datetime DEFAULT NULL COMMENT '出站时间',
  `status` enum('未发货','已发货','已接单','运输中','配送中','已到站','已收货') NOT NULL DEFAULT '
    未发货' COMMENT '物流状态',
  `station_price` float DEFAULT NULL COMMENT'寄存价格',
  `express_price` float DEFAULT NULL COMMENT'运费',
  `sender_iid` int NOT NULL COMMENT'发件人身份ID',
  `receiver_iid` int NOT NULL COMMENT'收件人身份ID',
  `expected_time` datetime DEFAULT NULL COMMENT'预计时间',
  PRIMARY KEY (`pid`),
  KEY `sender_index` (`sender_id`),
  KEY `receiver_index` (`receiver_id`),
  KEY `departure_index` (`pdeparture`),
  KEY `destination_index` (`pdestination`),
  KEY `courier_a_index` (`courier_a_id`),
  KEY `driver_index` (`driver_id`),
  KEY `station_index` (`station_id`),
  KEY `iid` (`sender_iid`) /*!80000 INVISIBLE */,
  KEY `iiid` (`receiver_iid`),
  CONSTRAINT `package_chk_1` CHECK ((`weight` < 100000)),
  CONSTRAINT `package_chk_2` CHECK ((`size` < 100000))
) PARTITION BY HASH(destination_index)
  SUBPARTITION BY RANGE COLUMNS(start_time)
  SUBPARTITION TEMPLATE (
    SUBPARTITION p0 VALUES LESS THAN ('2021-06-01'),
    SUBPARTITION p1 VALUES LESS THAN ('2021-07-01'),
    SUBPARTITION p2 VALUES LESS THAN ('2021-08-01'),
    SUBPARTITION p3 VALUES LESS THAN ('2021-09-01'),
    SUBPARTITION p4 VALUES LESS THAN ('2021-10-01'),
    SUBPARTITION p5 VALUES LESS THAN ('2021-11-01'),
    SUBPARTITION p6 VALUES LESS THAN ('2021-12-01'),
    SUBPARTITION p7 VALUES LESS THAN ('2022-01-01'),
    SUBPARTITION p8 VALUES LESS THAN ('2022-02-01'),
    SUBPARTITION p9 VALUES LESS THAN ('2022-03-01'),
    SUBPARTITION p10 VALUES LESS THAN ('2022-04-01'),
```

```
    SUBPARTITION p11 VALUES LESS THAN ('2012-05-01'),
    SUBPARTITION p12 VALUES LESS THAN (MAXVALUE)
  )
 PARTITIONS 10;
```

表分区能增强系统的可扩展性、可管理性以及提高性能。包裹信息的有效期延长至一年，并且可以实现快捷的定期清理。各地租户可以通过 OceanBase 分区裁剪的功能高效访问相应的数据分区。同时，也可对 Store 表基于驿站进行 Hash 分区。

10.2 表分组

表分组作为 OceanBase 特有功能，可以将需要进行频繁联合查询的表进行组合，让其同时存储在同一节点，以提高查询效率。本系统中最需频繁联合查询的是 Map 表和 Package 表（每次查询相应地区都需要联合查询以将地区编号转为地区名称），但是 Package 表已经经过二级分区，再进行分组会使运维变得异常复杂。因此初步设想将 Service 表和 Map 表分组，将 UserLogin 表和 Identity 表分组。以前者为例，MySQL 租户示例：

```
create tablegroup service_map partition by hash partitions 10;
alter tablegroup add service,map;
```

10.3 异地多活

当物流业务范围扩大之后，可采用 OceanBase 的“三地五中心异地多活”部署模式，有效解决城市级容灾问题。如 Figure9，将 Identity 表和 UserLogin 表根据 uid 拆分成 100 个表，并且进行分组，分布在五个租户中，每个租户的主副本分布在不同的 Zone。Zone1 和 Zone2 是同城双机房，Zone3 和 Zone4 是同城双机房，两个城市靠的比较近，Zone5 实际很远，所以一般不提供写入。为节省空间，Zone5 里的副本放的是日志副本。在实际运行过程中，通过负载均衡调整用户流量到对应的 Zone，能基本实现本地访问，增大响应速率。在单个城市故障的情况下，可以保证任何数据不丢失。

11 演示数据说明

数据说明

由于隐私保护等原因，现实中的快递数据难以通过网络搜索或爬取获得，因此我们选择通过随机生成的方式模拟出符合需求的数据集。

数据生成主要使用了 python，以 NumPy、Pandas 库的数据处理功能为核心，通过 random 库的随机数和随机选取功能创建。（代码随提交文件附上）

最初我创建的数据集包含了中国所有区级单位，共 2856 个，后由于数据量过大会造成用户界面设计工作量过大，因此最后精简了数据集：

数据集	说明	数据集大小
市、区二级地址[1]	采用中华人民共和国行政区划代码，从中国区级单位中筛选了北京、上海、深圳、成都等大型城市的下属单位，共 81 项。 同时增加了区级经纬度[2]，以便后续可视化操作	81
线路信息	在所选的区级地址直接两两生成线路（考虑顺序，也包括本区到本区），通过 random 库随机生成时效和单价，id 从 0 开始自增升序排列。	81*81
快递员	为每个区匹配一名快递员，姓、名各自从我设置的库中通过 random 随机选取并配对，手机号和薪酬也在规定范围内随机选取，id 从 0 开始自增升序排列。	81
货车司机	为每条线路匹配一名货车司机，姓、名各自从我设置的库中通过 random 随机选取并配对，手机号和薪酬也在规定范围内随机选取，id 从 0 开始自增升序排列。	81*81
驿站信息	为每个区匹配一个驿站，具体地址从[1]的街道信息中随机选取并与随机门牌号连接，id 从 0 开始自增升序排列。	81
驿站管理员	为每个驿站匹配一名管理员，姓、名各自从我设置的库中通过 random 随机选取并配对，手机号和薪酬也在规定范围内随机选取，id 从 0 开始自增升序排列。	81
用户注册信息	手机号随机选取，密码在 8-20 位数字中随机选取并进行哈希加密，id 从 0 开始自增升序排列。（id 为 0 的用户将手机号设为 11451419198，密码设定为 123456 的加密结果，便于实	1000

数据集	说明	数据集大小
	验）。	
员工注册信息	将快递员、货车司机、驿站管理员的手机号、id 信息归入表中，再全部加上默认密码 123456 哈希加密后的结果。 此外，还特别加入了一名‘超管’信息，以对应超管功能。	81+81*81+81+1
用户身份信息	每个用户可以创建若干个身份信息用于收发快递，我先调用之前生成的用户注册信息中的用户 id，在为这些用户各自随机生成若干自己的身份信息，地址采用和驿站相似的生成方法，汇总起来的身份 id 也从 0 开始自增。	1000
包裹信息	汇总了之前生成的各项信息，随机生成了 10000 件包裹，关联起发货、运输、收货各环节的 id 与信息； 主要是将包裹分为未发货、已接单、已发货、运输中、配送中、已到站、已取货几个状态，随机选取每个包裹的状态，不同状态包裹所包含的信息不同。	10000
存储信息	记录每个驿站、每个货架（1-20）、每一层（1-6）当前的暂存货物数（处于已到站状态）。	81*20*6

【1】：数据来源：uiwjs/province-city-china: CN最全最新中国【省、市、区县、乡镇街道】json,csv,sql 数据 (github.com)

【2】：经纬度来源：pfinal/city: 中华人民共和国行政区划数据：省份、城市、区县。中国省市区镇三级联动地址数据。城市经纬度数据。 (github.com)

13 项目总结

经过本次项目实践，我们充分运用了课堂所学知识，项目为我们提供了一个很好的机会去应用实践课堂将手的数据库知识，有助于巩固相关内容，为未来进一步的工作做铺垫。我们对数据库的潜能有了新的认识。诸多实际生产生活中的业务如订单管理、人事管理、信息检索、仓储管理都可以基于数据库技术安全高效地运行。在应用 OceanBase 为代表的分布式技术后，更能进一步安全高效地管理海量数据。

我们在探索解决方案如前端设计、部署开发中收获了丰富的实践经验，在课堂传授内容之外，还学习了许多诸如前端开发、前后端交互等对实际开发不可或缺的课外知识。尽管我们的系统距离实际生产环境所需还有一定距离，但这些经验对我们未来的工作学习生活来说必定弥足珍贵。

我们开发主要使用 Django 下的 ORM 框架，我们仔细考察了 Django 有关事务 ACID、防止 SQL 注入等底层逻辑，在调试过程中也会频繁地使用底层的 SQL 语句，例如在某些视图显示不如人意时，我们会进入底层数据库查询相关记录；在我们的业务流程进行到一半时出错，我们也会进入底层数据库恢复相关记录。因此，我们本次实践同时也巩固了理论课程中学到的数据库知识。

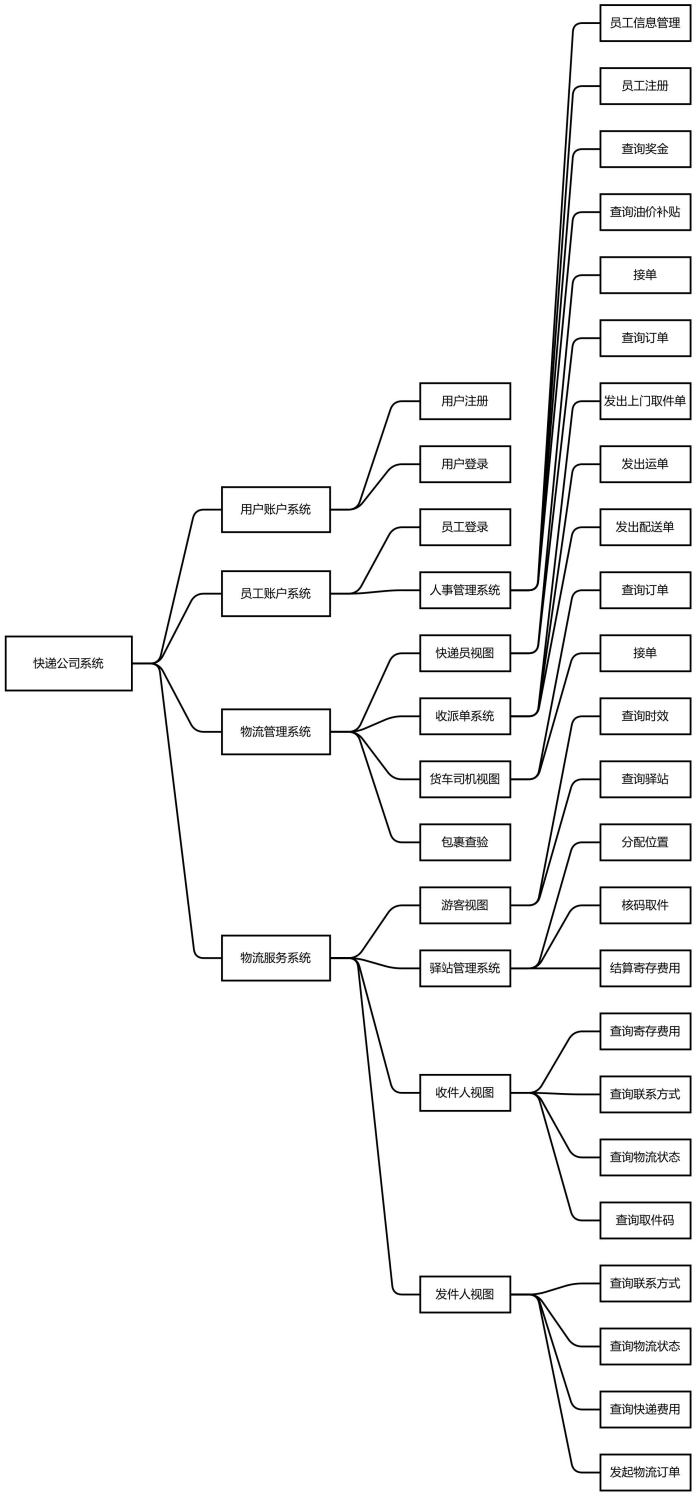


Figure 7: 功能设计

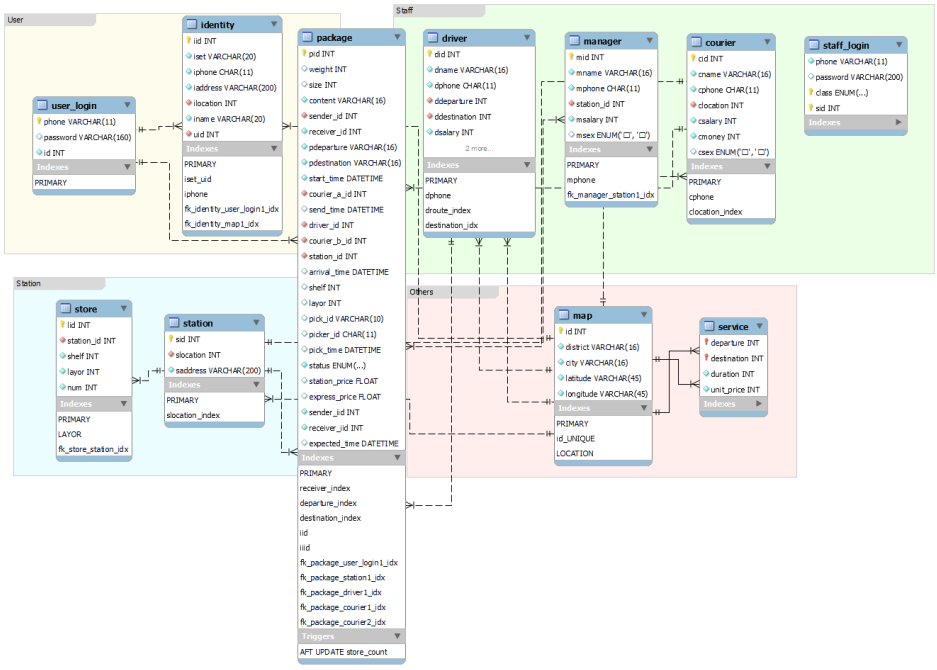


Figure 8: 数据库全局 ER 图

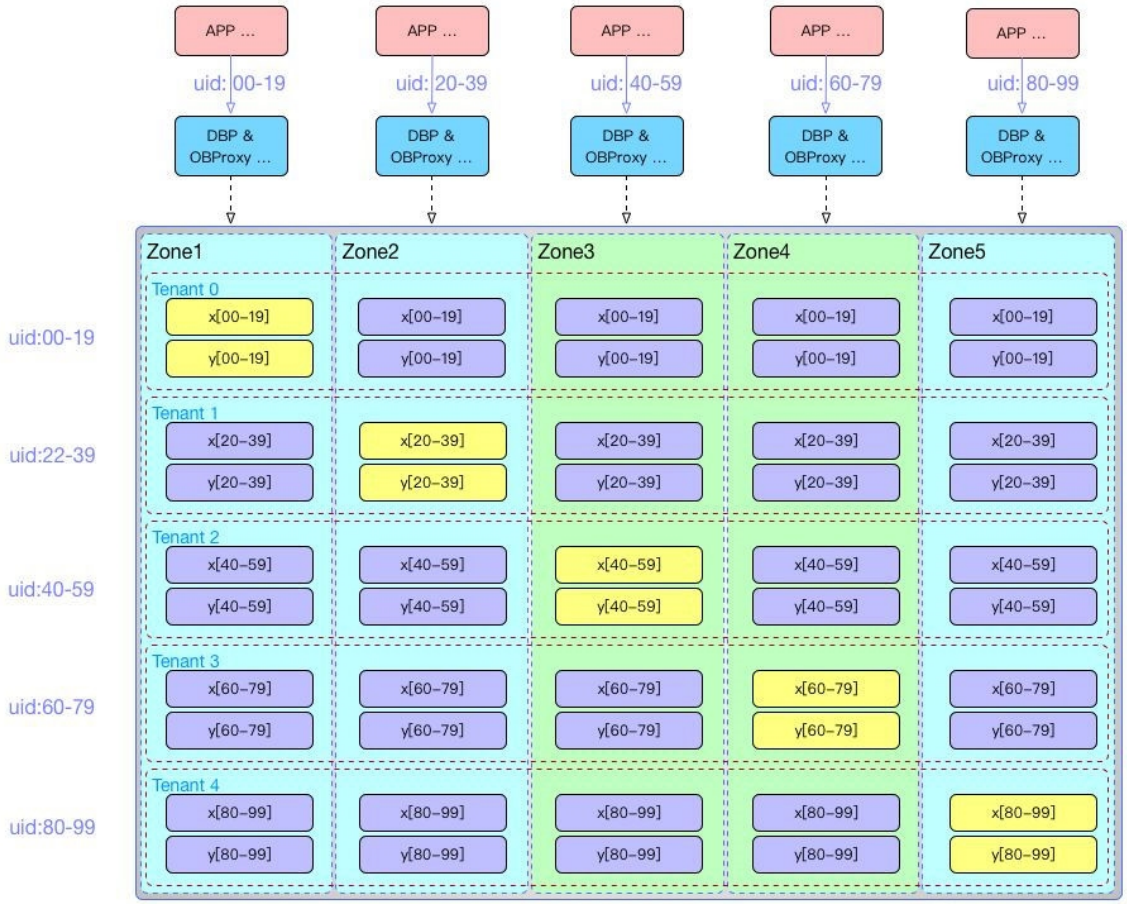


Figure 9: 三地五中心部署模式