基于 SQL Server 的打车系统数据库分析与设计

瞿馨楠 曾 丽
成都锦城学院计算机与软件学院 四川 成都 611731

【摘要】针对现有打车软件所存在的不足，以 SQL Server 数据库技术为基础，分析并实现了基于 C/S 结构的打车系统，旨在指向性地提前对运营车辆进行调度，解决用户打车难的问题，提升乘客的出行体验，同时能够缓解早晚高峰交通拥塞问题。本文从需求分析、数据库设计、数据库实现、数据、功能测试、安全性等方面出发，详细介绍了系统的设计过程。

【关键词】SQL Server；打车系统；出行体验；

1 项目介绍

众所周知，用户使用打车软件的出行体验越好，其今后持续使用打车软件的意愿就会越强烈，但在实际生活中，由于在不同的区域、不同的时段，有车辆需求的乘客数量不同，因此有可能出现处于空载状态的车辆过多以及乘客所在区域车辆供不应求的情况，就会对用户出行的服务体验感造成一定的负面影响以及会使用户使用打车软件的意愿有所降低。

因此，为了解决上述等情况，此打车系统能够准确统计出成都各个区域在不同时间段内的出行量，以便预测出在未来的某一时间段内哪些区域的出行需求量较大，并可以根据人们在打车软件中的用车请求指向性地提前对运营车辆提供一些引导，对车辆进行调度，以求最大程度上缩短乘客的等车时长，解决早晚高峰时期打车困难的问题，进而在更大程度上满足用户的用车需求，同时能够缓解交通压力，提升乘客的整体出行体验，从而加强用户的绩效期望。

2 需求分析

实现一个打车出行系统，其系统设计要求包括乘客信息管理、司机信息管理、打车公司信息管理、车辆信息管理、订单信息管理五部分。面向的用户分为乘客、司机、打车公司、管理员，因此针对不同的用户群体需要满足其特定的功能需求：

1. 乘客。乘客能够注册登录并对个人信息进行更改，发起订单匹配车辆开始行程，并且可以查询自己的订单信息，包含车辆信息、乘车人数、行程时间及地点等具体信息。

2. 司机。司机能够注册入职打车公司并对个人信息进行更改，接单完成行程，并且可以查询自己的订单信息，包含乘客信息、车辆信息、乘车人数、行程时间及地点等具体信息。

3. 打车公司。打车公司能够注册入驻并对公司信息进行更改，可查询本公司的司机详细信息及订单信息，统计司机人数和订单总数，了解公司的业绩情况。

4. 管理员。系统管理员具有对打车出行系统进行管理和操作的最高权限，能够查看后台的程序和数据，并能对不同用户设置和调整权限，保证数据的完整性和安全性。

3 数据库设计

3.1 系统概念模型设计

数据库设计的优劣不但会影响应用系统的运行效率，同时还会影响程序实现的效果。因此，设计合理的数据库模型不仅可以提高数据存储的效率，保证数据的一致性和完整性，也有利于程序功能的实现。

该打车出行系统主要面向乘客、司机、打车公司，因此包括乘客实体、司机实体、打车公司实体、车辆实体以及订单实体。

此打车出行系统的 E-R 图如下：

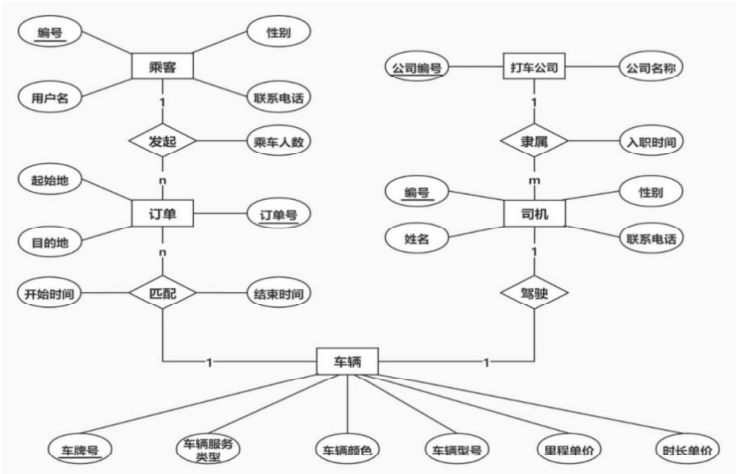

图 1 打车出行系统全局 E-R 图

其各个实体间的关系图如下：

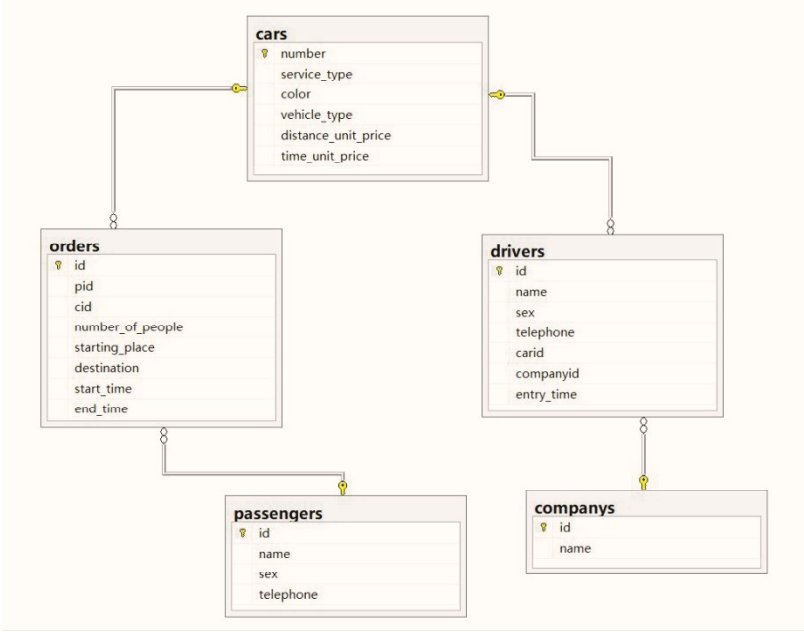

图 2 关系图

3.2 逻辑结构设计

E-R 模型向关系模型的转换如下：

打车公司（公司编号，公司名称）

车辆（车牌号，车辆服务类型，车辆颜色，车辆型号，里程单价，时长单价）

司机（编号，姓名，性别，联系电话，车牌号，公司编号，入职时间）

乘客（编号，姓名，性别，联系电话）

订单（订单号，乘客编号，车牌号，乘车人数，起始地，目的地，开始时间，结束时间）

4 数据库实现

4.1 表

表是数据库中最重要的对象，表的设计和创建是数据库的前提和基础，对整个系统架构的设计尤为重要。现基于分析及需要，本打车出行系统共建立了乘客信息表、打车公司信息表、司机信息表、车辆信息表、订单信息表五张表。下面以乘客信息表、司机信息表和订单信息表为例来具体说明表的设计和创建过程。

乘客信息表应包含下列信息：乘客编号、姓名、性别、联系电话。通过 create table passengers 语句建表，设置乘客编号的数据类型为 int，作为唯一主键，唯一且非空；设置姓名的数据类型为 varchar，非空约束；设置性别的数据类型为 char，默认值约束、检查约束；设置联系电话的数据类型为 bigint，非空约束、唯一约束。

司机信息表应包含下列信息：司机编号、姓名、性别、联系电话、车牌号、公司编号、入职时间。通过 create table drivers 语句建表，设置司机编号的数据类型为 int，作为唯一主键，唯一且非空；设置姓名的数据类型为 varchar，非空约束；设置性别的数据类型为 char，默认值约束、检查约束；设置联系电话的数据类型为 bigint，非空约束、唯一约束；设置入职时间的数据类型为 date，是司机与打车公司之间的隶属联系的属性；车牌号和公司编号均为外键，车牌号引用车辆表中的主键 number，公司编号引用打车公司表中的主键 id。

订单信息表应包含下列信息：订单号、乘客编号、车牌号、乘车人数、起始地、目的地、开始时间、结束时间。其中订单号作为唯一主键，乘客编号和车牌号均为外键。

表 1 订单信息表

orders表				
数据项名	数据类型	长度	完整性约束	备注
订单号	int		主键、唯一、非空	
乘客编号	int		外键	引用passengers表的主键id
车牌号	nchar	7	外键	引用cars表的主键number
乘车人数	int		列取值范围	乘客与订单之间的发起联系的属性
起始地	nvarchar	20	非空	
目的地	nvarchar	20	非空	
开始时间	datetime			订单与车辆之间的匹配联系的属性
结束时间	datetime			订单与车辆之间的匹配联系的属性

4.2 视图

将一些频繁需要使用的查询语句定义为视图不仅可

以为用户集中数据、简化数据操作、提高数据的操作效率和数据库性能，同时还能保证数据库的安全性和数据的逻辑独立性。

1. 为了方便观测早高峰时期的打车出行信息以及交通情况，创建早高峰视图，查询上午时间段 07:00—09:00 的订单信息。

```
create view 早高峰
as
select * from orders
where start_time between '2021-03-15 07:00' and '2021-03-15 09:00'
```

2. 为了方便观测成都市武侯区的打车出行情况，创建武侯区视图，查询由武侯区起始的订单信息。

```
create view 武侯区
as
select * from orders
where starting_place like '武侯区%'
```

4.3 存储过程

存储过程是存储在数据库服务器上的、由 SQL 语句和流程控制语句组成的预编译集合，其独立于程序源代码而单独修改，可以直接在应用程序中调用多次存储过程。

（1）创建存储过程 del_passenger，删除指定名字的乘客信息，同时删除该乘客的订单信息。

```
create proc del_passenger
@pname char(8)
as
delete from orders
where pid in(select id from passengers
where name = @pname)
delete from passengers
where name = @pname
--执行
exec del_passenger '王凯'
```

（2）创建存储过程 area，根据给定区域，返回由该区域起始的所有订单信息并按时间先后顺序排序。

```
create proc area
@area varchar(8)
as
select * from orders
where starting_place like @area+'%'
order by end_time
--执行
exec area '锦江区'
```

4.4 触发器

触发器是一段由对数据进行 INSERT、UPDATE 或 DELETE 更改操作而引起的自动执行的代码，常用于实现复杂的处理逻辑和业务规则，以增强数据完整性约束的功能。

（1）使用 FOR/AFTER（后）选项定义的触发器叫做后触发性触发器，也就是只有在引发触发器执行的语句中指定的操作都已经成功执行后，才能执行该触发器。

用后触发器实现插入新的司机信息时，提示：欢迎 ** 司机入职！

```
create trigger insert_newdriver
on drivers
after insert
as
declare @dname char(8)
select @dname = name from inserted
print '欢迎' + rtrim(@dname) + '司机入职！'
```

（2）使用 INSTEAD OF（前）选项定义的触发器叫做前触发型触发器，也就是在前触发型触发器中，指定执行触发器而不是执行引发触发器执行的具体操作，从而替代引发触发器语句的执行，因此常用于阻止不符合规则的操作发生。

用前触发器实现插入新的订单信息时，如果乘车人数大于 4，提示：超过限载人数！

```
create trigger insert_neworder
on orders
instead of insert
as
declare @number int
select @number = number_of_people from inserted
if @number > 4
print '超过限载人数！'
```

5 测试

5.1 数据测试

（1）向车辆表中插入新的车辆信息：

```
insert into cars(number, service_type, color, vehicle_type, distance_unit_price, time_unit_price)
values('川 A753ST','舒适型快车','黑色','大众途观','2.42','0.46')
```

（2）将订单号为 16 的订单信息删除：

```
delete from orders
where pid = 16
```

（3）将司机姓名为马洋的个人信息中的性别修改为“女”：

```
update drivers
set sex = '女'
where name = '马洋'
```

5.2 功能测试

（1）查询订单最多的车辆，显示车牌号、司机姓名、订单数：

```
select top 1 with ties cid 车牌号, drivers.name 司机姓名, count(cid) 订单数 from orders
join cars on orders.cid = cars.number
join drivers on cars.number = drivers.carid
group by cid,drivers.name
order by count(cid) desc
```

（2）查询各个打车公司的订单数，显示打车公司名称、订单数：

```
select companys.id 编号, companys.name 公司名称, count(*) 订单数 from companys
```

```
join drivers on companys.id = drivers.companyid
join orders on drivers.carid = orders.cid
group by companys.id,companys.name
order by count(*) desc
```

6 安全性

由于用户以及司机在注册登录出行系统时，需要填写一系列必要的个人信息，并且在出行过程中也会生成相应的订单信息，而该出行系统采用的是线上交易，乘客需要通过微信、支付宝等网络平台进行费用的支付，网络的开放性则会对电子支付造成信息泄露、网贷诈骗等安全方面的漏洞和隐患。因此，必须重视电子支付安全性的保障，负责数据库系统的全部安全的数据库管理员 (DBA) 需要能够识别出一系列会造成数据被意外或故意的丢失、破坏或滥用等威胁行为，并实施安全措施和采取适当的安全控制策略，从而防止数据库被滥用和误用，除了采用安全措施外，还可以对用户私人信息（用户密码、电话号码等）这些高度敏感的数据采用数据加密技术，尽量最大程度上降低系统数据风险，增强安全性，为用户提供好隐私保障。

此外，在数据库的运行阶段，数据库管理员 (DBA) 还需要对数据库进行经常性的维护工作，包括数据库的备份和恢复、安全性和完整性控制、监视、分析、调整数据库的性能等工作，以保证数据库始终能够保持良好的性能。

7 结语

本文从乘客、司机、交通情况等实际需求出发，分析并设计了基于 SQL Server 数据库的打车出行系统，此打车系统会根据人们在打车软件中的用车请求对车辆进行调度，进而在更大程度上满足用户的用车需求，力求最大程度上提升乘客的出行体验，加强用户的绩效期望。

在当前以服务型经济为主导的社会经济体制下，对于打车出行系统来说，需要不断完善和优化软件功能，力求为用户提供更高质量的服务，最大程度上提升用户的出行体验，从而加强用户的绩效期望和使用意愿。

【参考文献】

[1] 张凤静，汪磊．基于 Oracle 数据库的车型数据管理系统设计与实现 [J]. 电子技术与软件工程，2018,17:199-199.

[2] 贾立伟，李琨，贾福运．基于 MySQL 的医院导航系统数据库设计与实现 [J]. 甘肃科技，2017,33(17):14-16.