New Paltz

STATE UNIVERSITY OF NEW YORK

DEPARTMENT OF COMPUTER SCIENCE

COURSE CODE-CPS 59305

DATABASE DESIGN

TERM- SPRING 2021

INSTRUCTOR-THOMAS MCKEE

PROJECT BASED

ON

RAILWAY MANAGEMENT SYSTEM

SUBMITTED BY

ASHOK KUMAR REDDY YANNAM

N04129149

TABLE OF CONTENTS

ABSTRACT

The Railway Management System help in the maintenance of the railway-related information easily. The project can help in storing train name, railway track, destination, date of departure, date of arrival, departure platform number, etc. of the train easily. It facilitates the passengers to enquire about the trains available based on source and destination, booking of tickets, enquire about the status of the train. The aim of case study is to design and develop a database maintaining the records of different trains, train status, and passengers. This project contains Introduction to the Railway management system.

Then this project contains entity relationship model diagram based on railway reservation system and introduction to relation model. There is also design of the database of the railway management system based on relation model. Example of some SQL queries to retrieves data from rail management database." This Railway management system also saves lot of time for passengers and they can reach the right platform easily.

SOFTWARE USED

ERD SOFTWARE: - LUCID CHART

DBMS: - MYSQL

Reason for choosing this software: -

Lucid Chart Reason:

This is one of top-rated software on internet to draw Entity Relationship Diagrams. Lucid chart is a cloud-based software which is free and most of the features are covered in it. I was planning to use Lucid chart which is very comfortable as it is a cloud-based system.

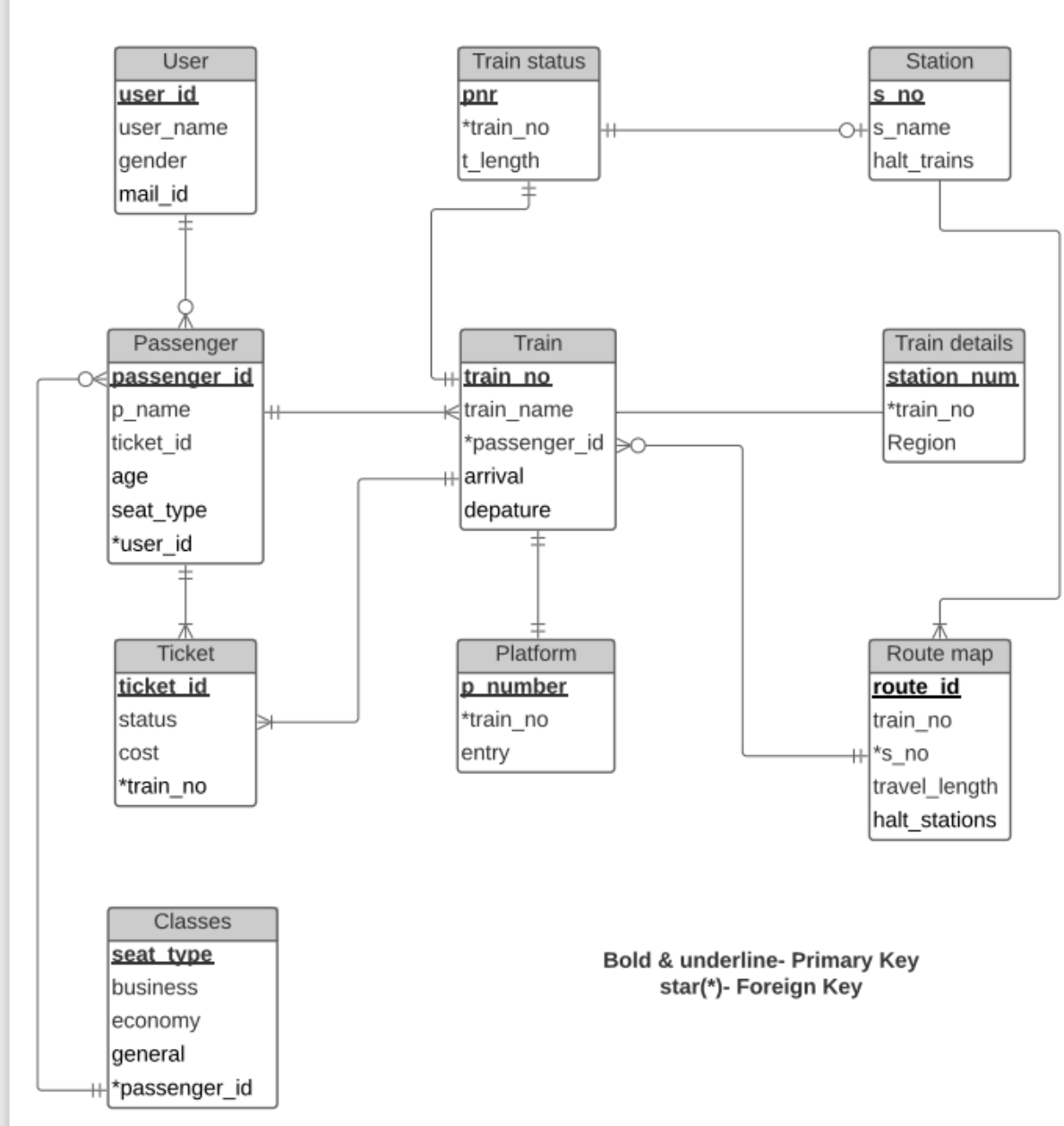
MYSQL Reason:

Being MYSQL an open-source Relational Database Management System. It is one of the most efficient and flexible software which can be used to build any application.

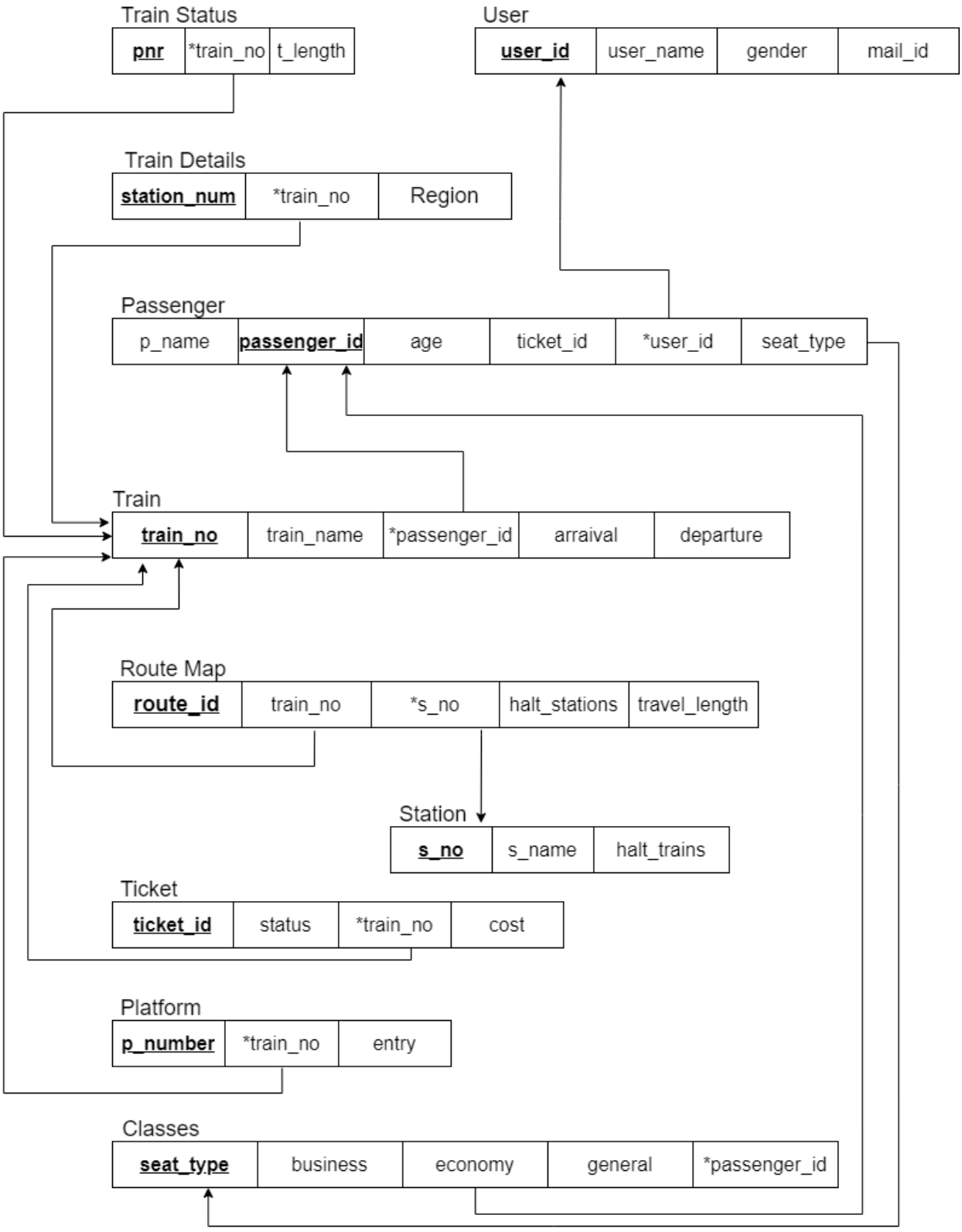
LIST OF ENTITIES AND ATTRIBUTES

ENTITIES	ATTRIBUTES
User	**User_id,** user_name, gender, mail_id
Train	**Train_no,** train_name, passenger_id, arrival, departure
Train status	**Pnr,** train_no, t_length
Station	**s_number**, s_name, halt_trains
Passenger	**Passenger_id**, p_name, ticket_id, age, seat_type, user_id
Train details	**s_num**, train_no, Region
Ticket	**ticket_id**, status, cost, train_no
Platform	**p_number**, train_no, entry
Route map	**route_id**, train_no, s_number, travel_length, halt_stations
Classes	**seat_type**, business, economy, general, passenger_id

ENTITY-RELATIONSHIP DIAGRAM



RELATION SCHEMA SHOWING REFERENTIAL INTEGRITY



SQL STATEMENTS TO CREATE TABLES

Table structure for User

```
CREATE TABLE User(
        userID VARCHAR(8) NOT NULL,
        mail_id VARCHAR(8),
        gender varchar(10),
        user_name varchar(20)
PRIMARY KEY (userID)
);
```

Table structure for Passenger

```
CREATE TABLE Passenger (
        p_name VARCHAR(8) NOT NULL,
        passenger_id VARCHAR(8) NOT NULL,
        ticket_id VARCHAR(9),
        user_id VARCHAR(8) NOT NULL,
        seat_type CHAR(5),
        age int,
        userID varchar(8),
PRIMARY KEY (passenger_id),
FOREIGN KEY (userID) REFERENCES user(userID)
);
```

Table structure for Train

```
CREATE TABLE Train(
        train_no VARCHAR(9) NOT NULL,
        train_name CHAR(8) NOT NULL,
        passenger_id VARCHAR(8) NOT NULL,
        arrival TIME(0),
        departure TIME(0),
PRIMARY KEY (train_no),
FOREIGN KEY (passenger_id) REFERENCES passenger(passenger_id)
);
```

Table structure for Train_Status

```
CREATE TABLE Train_Status (
        Pnr VARCHAR(8) NOT NULL,
        t_length int, train_no VARCHAR(9),
PRIMARY KEY (Pnr),
FOREIGN KEY (train_no) REFERENCES train(train_no)
);
```

Table structure for Station

```
CREATE TABLE Station (
        s_name VARCHAR(8) NOT NULL,
        s_no VARCHAR(8) NOT NULL,
        halt_trains VARCHAR(9),
PRIMARY KEY (s_no)
);
```

Table structure for route_map

```
CREATE TABLE route_Mapp (
        train_no VARCHAR(9) NOT NULL,
        rootID varchar(10) not null primary key,
        s_no CHAR(8) NOT NULL,
        halt_stations CHAR(8),
        route_length int,
FOREIGN KEY (s_no) REFERENCES station(s_no)
);
```

Table structure for ticket

```
CREATE TABLE ticket(
        TicketID VARCHAR(20) NOT NULL,
        Status VARCHAR(20) NOT NULL,
        train_no int,
        Cost DECIMAL(10,2),
PRIMARY KEY(TicketID)
);
```

Table structure for Platform

```
CREATE TABLE Platform(
        PNo int NOT NULL,
        train_no VARCHAR(9),
        PRIMARY KEY(PNo),
FOREIGN KEY(train_no) REFERENCES TRAIN(train_no)
);
```

Table structure for TrainDetails

```
CREATE TABLE TrainDetails(
        StationNum int NOT NULL,
        train_no VARCHAR(8),
        region varchar(10),
PRIMARY KEY(StationNum),
FOREIGN KEY(train_no) REFERENCES Train(train_no)
);
```

Table structure for classes

```
CREATE TABLE classes (
        seat_type varchar(20),
        business varchar(10),
        general varchar(10),
        passenger_id varchar(8),
        economy varchar(10),
FOREIGN KEY (passenger_id) REFERENCES passenger(passenger_id)
);
```

SQL STATEMENTS TO INSERT DATA INTO TABLES

USER:

insert into user values

('ashok1','ash1@gmail.com','m','ashok'),

('sharief2','sha2@gmail.com','m','sharief'),

('sai3','sai3@gmail.com','m','sai'),

('suri4','suri4@gmail.com','m','suri');

PASSENGER:

insert into passenger values

('ashok','1234','456','AC',20,'ashok1'),

('sharief','1235','457','NONAC',25,'sharief2'),

('sai','1236','458','TIREA',26,'sai3'),

('suri','1237','459','TIREB',24,'suri4');

TRAIN:

insert into train values

('TR1234','amar','1234','11:30:00','11:35:12'),

('TR1235','ganga','1235','12:23:23','02:55:02'),

('TR1236','vadodar','1236','13:23:23','23:12:56'),

('TR1237','delhi','1237','01:23:23','06:52:00');

TRAIN_STATUS:

insert into train_status values

('pnr1',28,'TR1234'),

('pnr2',58,'TR1235'),

('pnr3',82,'TR1236'),

('pnr4',77,'TR1237');

STATION:

insert into station values

('ANDHRA','s32','4'),

('HYD','sn2','2'),

('DELHI','sn3','3'),

('GOA','sn4','4');

ROUTE_MAP:

insert into route_mapp values

('TR1234','r1','s32','2',535),

('TR1235','r2','sn2','3',544),

('TR1236','r3','sn3','10',95),

('TR1237','r4','sn4','8',758);

TICKET:

insert into ticket values

('tk1','running','TR1234',23.56),

('tk2','running','TR1235',56.56),

('tk3','stopped','TR1236',12.23),

('tk4','running','TR1237',100.00);

PLATFORM:

insert into platform values

(12,'TR1234'),

(23,'TR1235'),

(45,'TR1236'),

(32,'TR1237');

CLASSES:

insert into classes values

('AC','y','n','1234','n'),

('NON-AC','n','y','1235','n'),

('TIRE-1','n','n','1236','y'),

('TIRE-2','y','n','1237','n');

TRAIN DETAILS:

insert into traindetails values

(12,'TR1234','south'),

(1,'TR1235','esat'),

(6,'TR1236','north'),

(11,'TR1237','south');

QUERIES AND RESULTS

INNER JOIN:

select passenger_id,

region,

train_name from traindetails inner join train on traindetails.train_no=train.train_no;

```
+--------------+--------+------------+
| passenger_id | region | train_name |
+--------------+--------+------------+
| 1234         | south  | amar       |
| 1235         | esat   | ganga      |
| 1236         | north  | vadodar    |
| 1237         | south  | delhi      |
+--------------+--------+------------+
4 rows in set (0.01 sec)

mysql>
```

OUTER JOIN:

select stationnum,

region,

train_name from traindetails left outer join train on traindetails.train_no=train.train_no;

```
+------------+--------+------------+
| stationnum | region | train_name |
+------------+--------+------------+
|          1 | esat   | ganga      |
|          6 | north  | vadodar    |
|         11 | south  | delhi      |
|         12 | south  | amar       |
+------------+--------+------------+
4 rows in set (0.00 sec)

mysql>
```

SUB-QUERY:

select * from train where train_no in(select train_no from ticket where ticketid in ('tk1','tk3'));

Result: -

```
+----------+------------+--------------+----------+-----------+
| train_no | train_name | passenger_id | arrival  | departure |
+----------+------------+--------------+----------+-----------+
| TR1234   | amar       | 1234         | 11:30:00 | 11:35:12  |
| TR1236   | vadodar    | 1236         | 13:23:23 | 23:12:56  |
+----------+------------+--------------+----------+-----------+
2 rows in set (0.02 sec)

mysql>
```

CORRELATED SUBQUERY:

SELECT DISTINCT p_name,userID,seat_type from passenger where exists(select*from train where train.passenger_id = passenger.passenger_id);

Result: -

```
+---------+----------+-----------+
| p_name  | userID   | seat_type |
+---------+----------+-----------+
| ashok   | ashok1   | AC        |
| sharief | sharief2 | NONAC     |
| sai     | sai3     | TIER1     |
| suri    | suri4    | TIER1     |
+---------+----------+-----------+
4 rows in set (0.01 sec)

mysql>
```

PROJECT EVOLUTION OVER TIME

Change-1

Moving further I understood that there are many mistakes in my ERD. The mistakes that I figured out in first ERD are

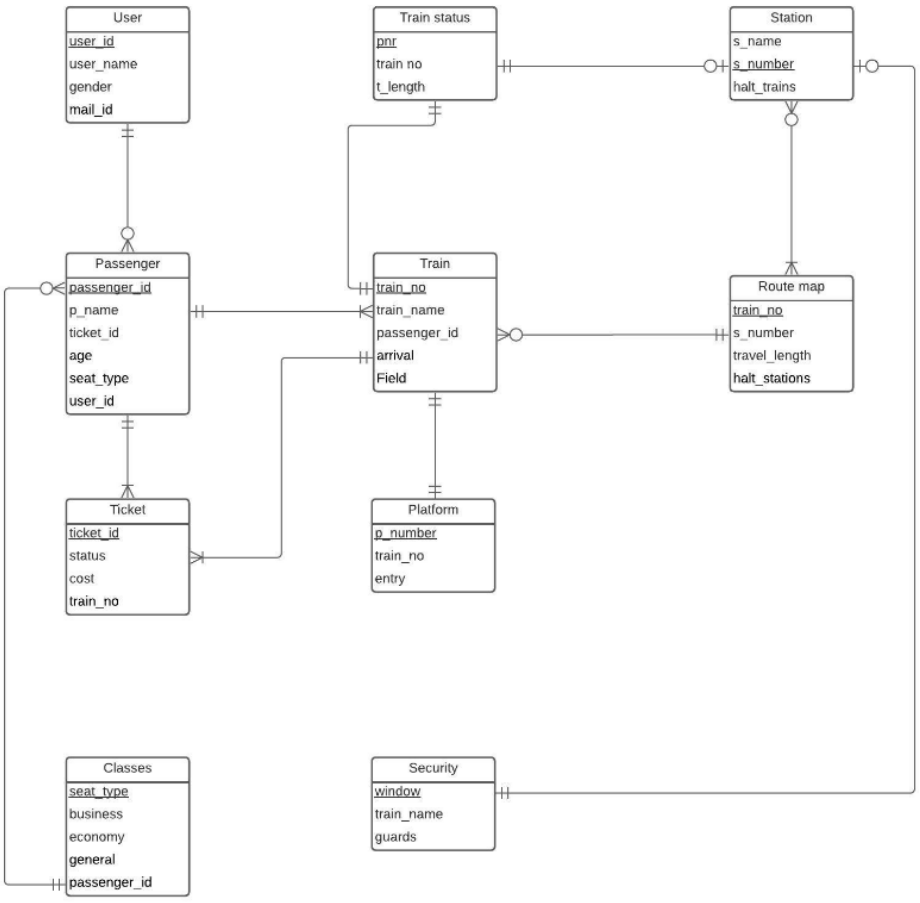
1) Some of the relationships established between are wrong and some are not necessary. (Example:' station 'and 'route map' relationship is one to many not many to many.)

2) The Names of entities and attributes are updated for easy understanding. The modification relationship between Train and Train details was added as I felt it is necessary and I added the new entity train details.

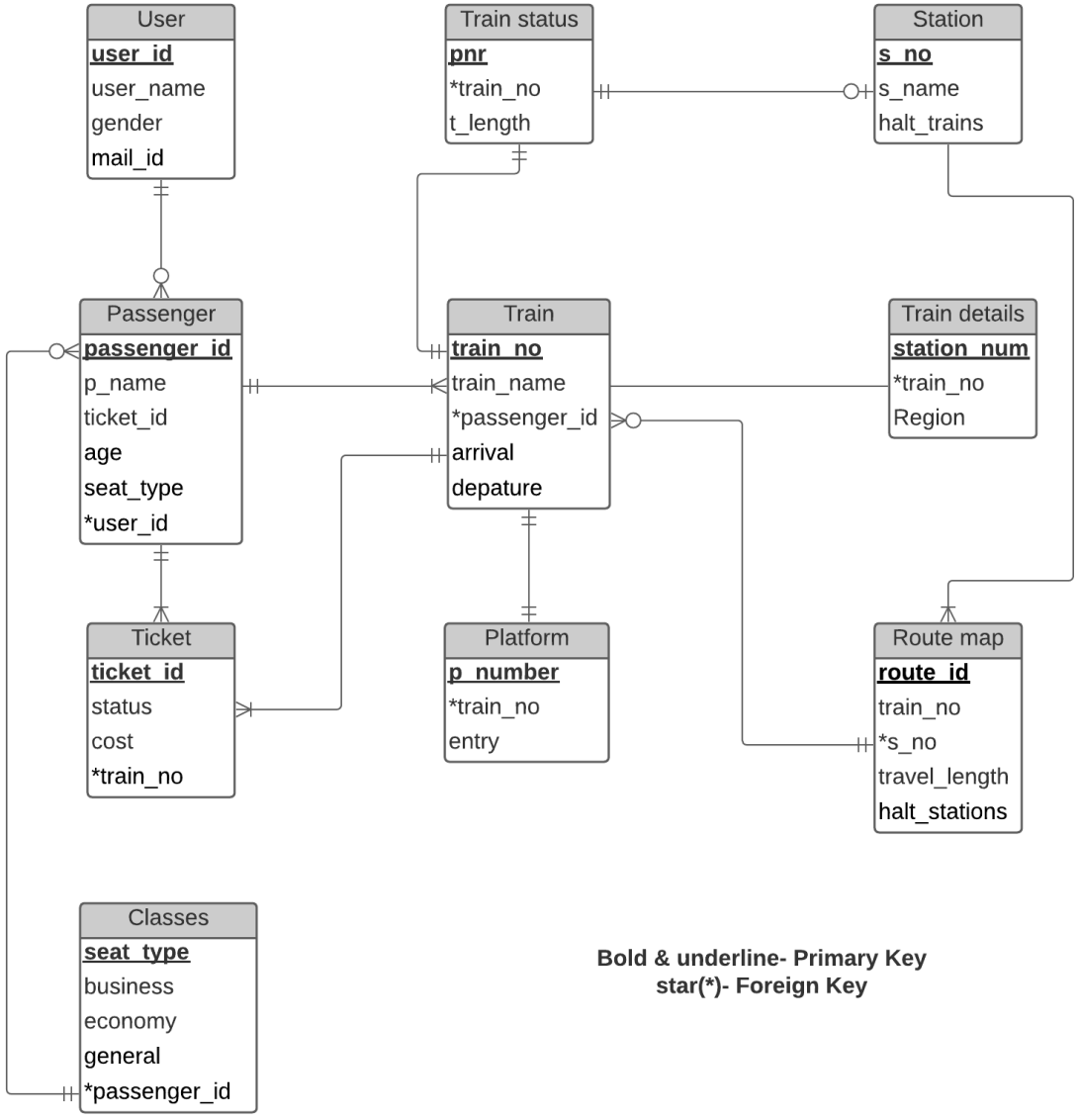
3) In final, the security entity was completely removed from ERD because it belongs to different department unrelated to my project.

4) In train table I took train name as primary key which does not satisfy primary key rule as train name is not unique, so I updated my primary key to train number. And also I added rootID as primary key in route map so it satisfied the referential integrity with the relational schema diagram.

Before changes: -



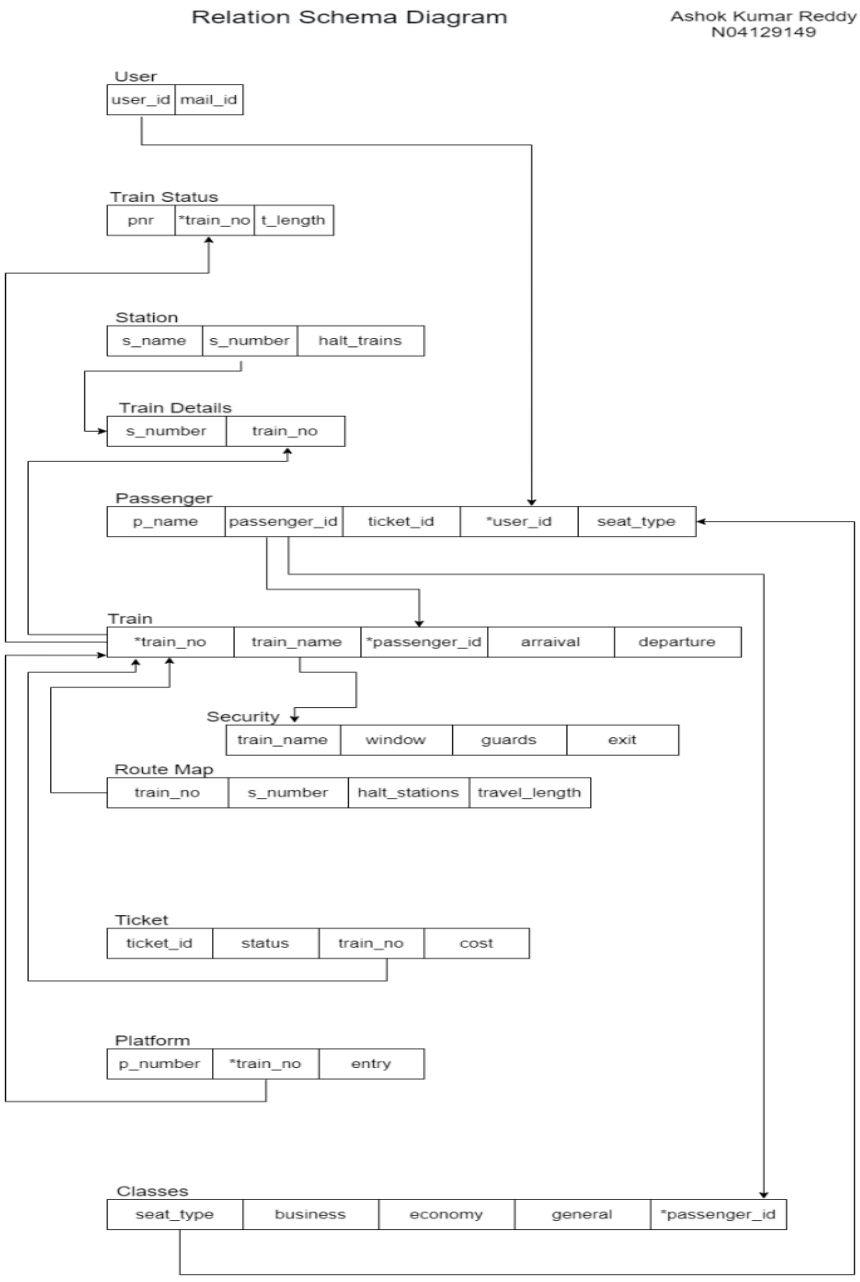
After changes: -



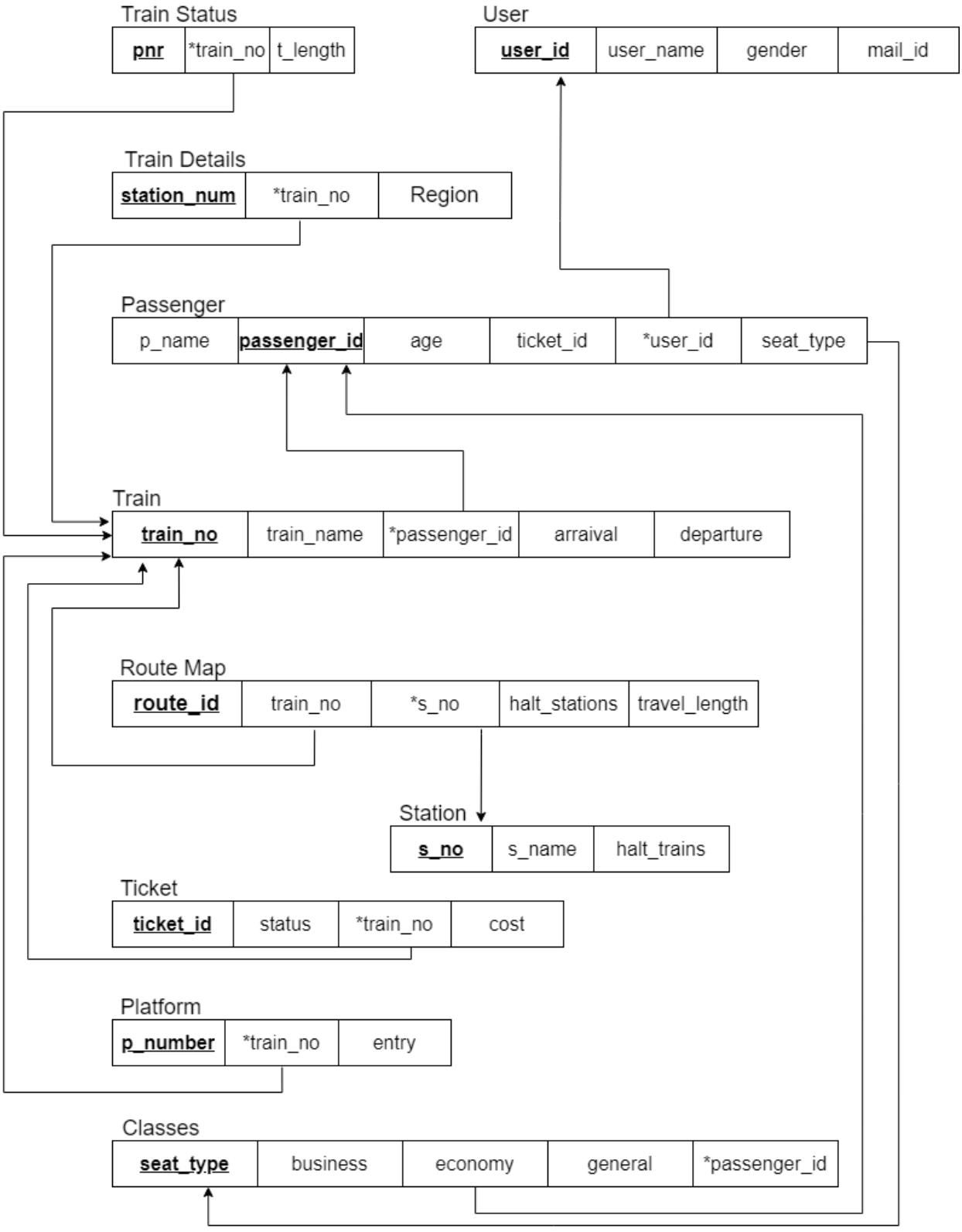
Change-2

I updated some connections in my relational schema diagram (For Example there is no connection between station and route map) and satisfying referential integrity rule which states that any foreign key value (on the relation of the many side) MUST match a primary key value in the relation of the one side. (Or the foreign key can be null.)

Before change: -



After Change: -

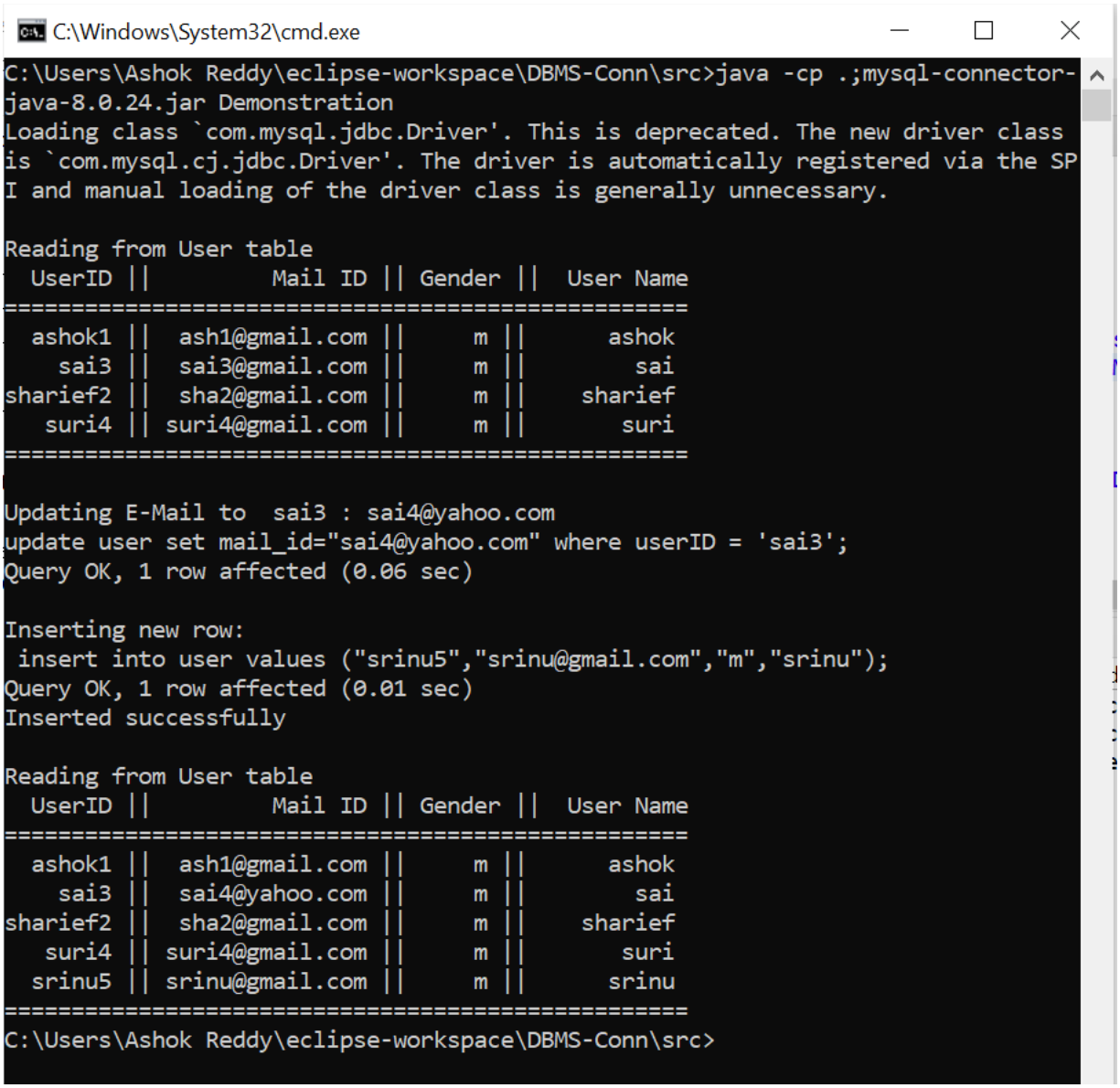


PROGRAMMING CODE DEMONSTRATING CONNECTION TO DATABASE

```
import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

public class Demonstration {

public static void main(String[] args) {

// TODO Auto-generated method stub

try {

Class.forName("com.mysql.jdbc.Driver");

//com.mysql.cj.jdbc.Driver

Connectionconn=DriverManager.getConnection("jdbc.mysql://localhost:3306/railway_management","root","rootpassword");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("select * from user");

System.out.printf("%8s || %50s || %10s || %20s \n","UserID","Mail ID","Gender","User Name");

System.out.println();

for (int i =0;i<=100;i++)

        System.out.print("=");

while(rs.next()) {

        String uid = rs.getString("userID");

        String mail = rs.getString("mail_id");

        String gen = rs.getString("gender");

        String usr = rs.getString("user_name");

        System.out.printf("%8s || %50s || %10s || %20s \n",uid,mail,gen,usr);

}
```

```
System.out.println();
for (int i =0;i<=100;i++)
        System.out.print("=");
System.out.println();
System.out.println("Updating E-Mail to  sai3 : sai4@yahoo.com\r\n");
rs = stmt.executeQuery("update user set mail_id=\"sai4@yahoo.com\" where userID = 'sai3'");
System.out.println("Inserting new row: ");
rs=stmt.executeQuery("insert into user values  (\"srinu5\", \"srinu@gmail.com\", \"m\",\"srinu\");
\r\ n");
System.out.printf("%8s || %50s || %10s || %20s \n","UserID","Mail ID","Gender","User Name");
System.out.println("Reading from user table");
for (int i =0;i<=100;i++)
        System.out.print("=");
System.out.println();
while(rs.next()) {
        String uid = rs.getString("userID");
        String mail = rs.getString("mail_id");
        String gen = rs.getString("gender");
        String usr = rs.getString("user_name");
        System.out.printf("%8s || %50s || %10s || %20s \n",uid,mail,gen,usr);
}
System.out.println();
for (int i =0;i<=100;i++)
        System.out.print("=");
}
catch(Exception e){
        System.out.println(e);
        }}}
```

Output: -


C:\Windows\System32\cmd.exe
C:\Users\Ashok Reddy\eclipse-workspace\DBMS-Conn\src>java -cp .;mysql-connector-java-8.0.24.jar Demonstration
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manual loading of the driver class is generally unnecessary.
Reading from User table
UserID || Mail ID || Gender || User Name
ashok1 || ash1@gmail.com || m || ashok
sai3 || sai3@gmail.com || m || sai
sharief2 || sha2@gmail.com || m || sharief
suri4 || suri4@gmail.com || m || suri
Updating E-Mail to sai3 : sai4@yahoo.com
update user set mail_id="sai4@yahoo.com" where userID = 'sai3';
Query OK, 1 row affected (0.06 sec)
Inserting new row:
insert into user values ("srinu5","srinu@gmail.com","m","srinu");
Query OK, 1 row affected (0.01 sec)
Inserted successfully
Reading from User table
UserID || Mail ID || Gender || User Name
ashok1 || ash1@gmail.com || m || ashok
sai3 || sai4@yahoo.com || m || sai
sharief2 || sha2@gmail.com || m || sharief
suri4 || suri4@gmail.com || m || suri
srinu5 || srinu@gmail.com || m || srinu
C:\Users\Ashok Reddy\eclipse-workspace\DBMS-Conn\src>