CSE 3330 – 004 PROJECT – 1

GROUP Number - 11

DONE BY – VICTOR AROWOSAFE, SIDDHARTH BHAGVAGAR, SHAFIN BARSHAN

Date of Submission – 10th March 2023.

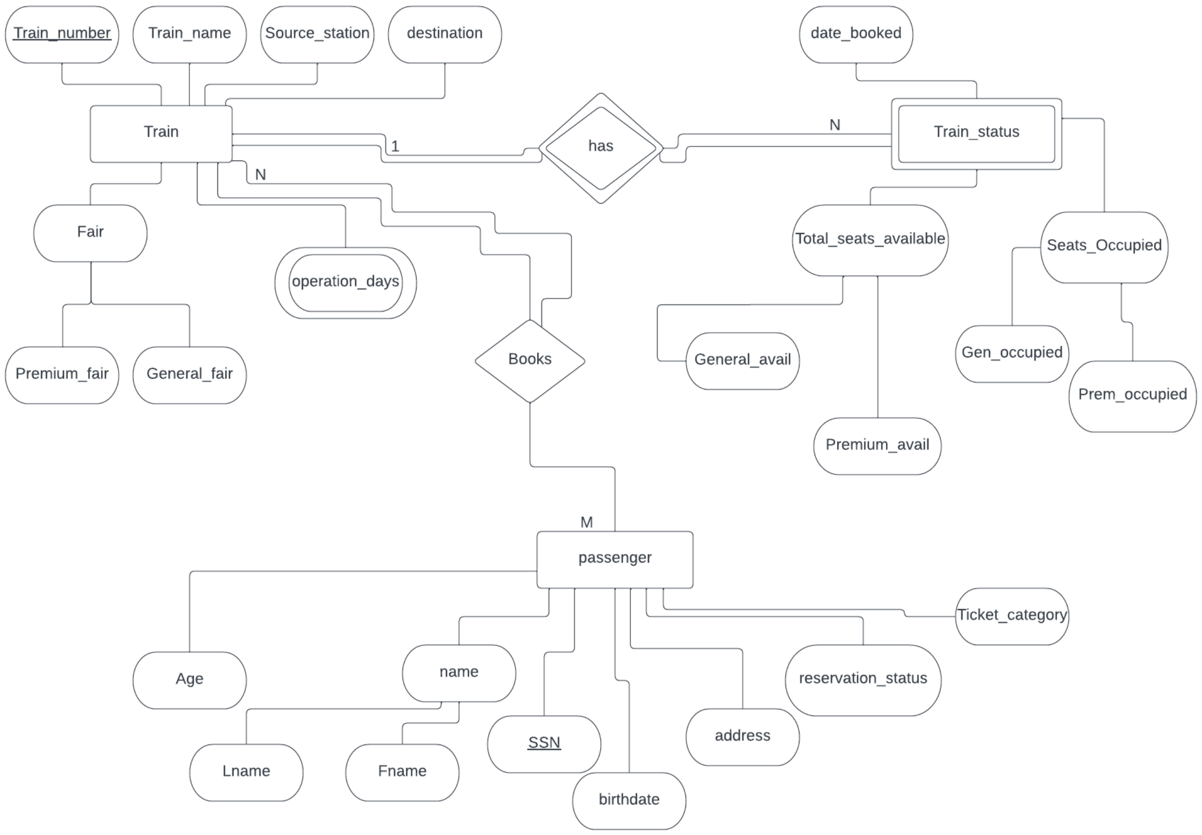
1) TOOLS USED FOR THE PROJECT:

- UTA Omega Server (Used Command Prompt to connect to omega.uta.edu)
- WinSCP/FileZilla (For File transfer from local machine to omega server directory)
- SQLlite3 (for database creation and querying) – accessed using omega server.
- Lucid Chart (for ER diagram)

2) ER Diagram:

Assumptions made to create the ER diagram

- Only two categories of tickets are available: Premium and General Ticket
- The total number of tickets can be booked in each category (Premium and General) is 10
- Number of tickets in waiting list is 2.
- Total Number of trains are 5.
- Any stops made by a train before its destination and their bookings are not considered.
- A train can have multiple train statuses (since it can be booked on multiple dates).
- Passenger has an SSN (which is the primary key), hence Passenger becomes a strong entity.
- Train can be empty (and even if its empty it will depart).
- All trains are able to be booked (i.e have an available ticket), so every train participates in the "Books" relationship. [Reference our ER diagram for clarification on "Books" relationship]
- Not all passengers are going to be able to book a ticket for a given date (if they are waitlisted). Hence not all passengers participate in the "Books" relationship.
- All trains have at least one train status, so all trains participate in the "has" relationship between "Train" and "Train_status" entities.


Train_number
Train_name
Source_station
destination
date_booked
Train
has
Train_status
1
N
N
Fair
operation_days
Total_seats_available
Seats_Occupied
Premium_fair
General_fair
Books
General_avail
Gen_occupied
Prem_occupied
Premium_avail
M
passenger
Ticket_category
Age
name
reservation_status
SSN
address
Lname
Fname
birthdate

README:

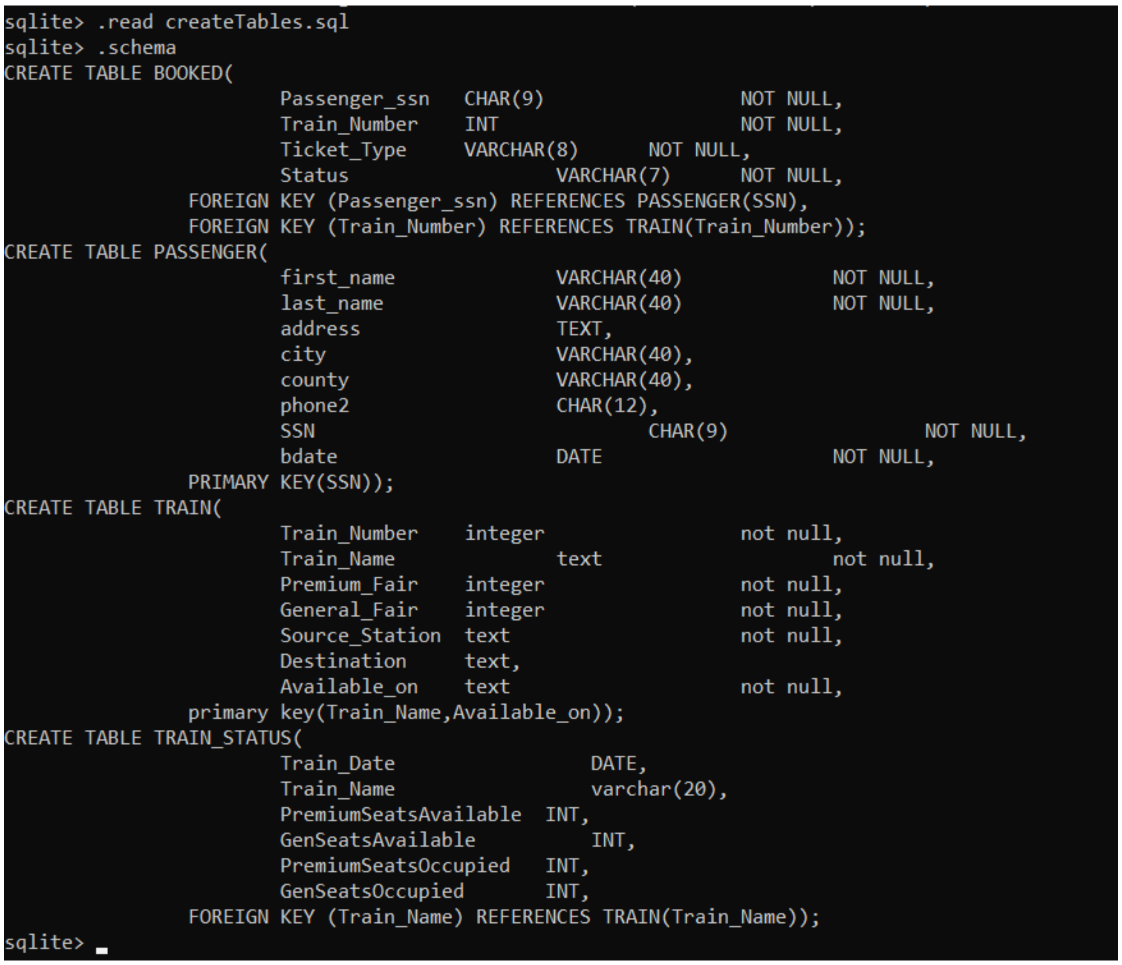
Steps to create the database from the .csv files and .sql files (that we submitted in the zip):

1) **sqlite3 trainDB.db** //create a new database file using sqlite3(using omega server)
2) **.read createTables.sql** //reads the createTables.sql file to create the tables
3) **.read importData.sql** //reads the importData.sql file to load the data from the csv files into the respective tables
4) **.read queries.sql** //reads the queries.sql file which has the queries

3) SOURCE CODE of SQL CREATE statements

See file named "createTables.sql" for the CREATE statements.

Included below is also the screenshot of how the file "createTables.sql" was used to create the tables for the database using sqlite3. Command ".schema" was then used to verify the tables have been created.



```
sqlite> .read createTables.sql
sqlite> .schema
CREATE TABLE BOOKED(
                        Passenger_ssn   CHAR(9)                 NOT NULL,
                        Train_Number    INT                     NOT NULL,
                        Ticket_Type     VARCHAR(8)      NOT NULL,
                        Status                  VARCHAR(7)      NOT NULL,
                FOREIGN KEY (Passenger_ssn) REFERENCES PASSENGER(SSN),
                FOREIGN KEY (Train_Number) REFERENCES TRAIN(Train_Number));
CREATE TABLE PASSENGER(
                        first_name              VARCHAR(40)             NOT NULL,
                        last_name               VARCHAR(40)             NOT NULL,
                        address                 TEXT,
                        city                    VARCHAR(40),
                        county                  VARCHAR(40),
                        phone2                  CHAR(12),
                        SSN                             CHAR(9)                 NOT NULL,
                        bdate                   DATE                    NOT NULL,
                PRIMARY KEY(SSN));
CREATE TABLE TRAIN(
                        Train_Number    integer                 not null,
                        Train_Name              text                    not null,
                        Premium_Fair    integer                 not null,
                        General_Fair    integer                 not null,
                        Source_Station  text                    not null,
                        Destination     text,
                        Available_on    text                    not null,
                primary key(Train_Name,Available_on));
CREATE TABLE TRAIN_STATUS(
                        Train_Date                 DATE,
                        Train_Name                 varchar(20),
                        PremiumSeatsAvailable  INT,
                        GenSeatsAvailable          INT,
                        PremiumSeatsOccupied   INT,
                        GenSeatsOccupied       INT,
                FOREIGN KEY (Train_Name) REFERENCES TRAIN(Train_Name));
sqlite> 
```

Commands to use: .read createTables.sql

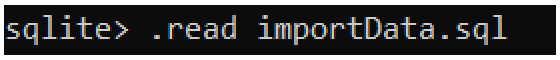
4) Method used to load data into table

See file named “importData.sql” for import statements.

We used the command “.read importData.sql” to read the file “importData.sql” (included in zip) to import each .csv file to its respective tables in the database.

The csv files provided were modified such that the headers (in the first line of each csv file) were removed. Additionally the file “Train.csv” was modified so that the multi-valued attribute “Available_on” was split. We split the attribute so that there was a new row for each attribute (as you can see in the Train.csv that is included in the zip).

Below is the screenshot which shows the command that was used to read the “importData.sql” file.



```
sqlite> .read importData.sql
```

Commands to use:

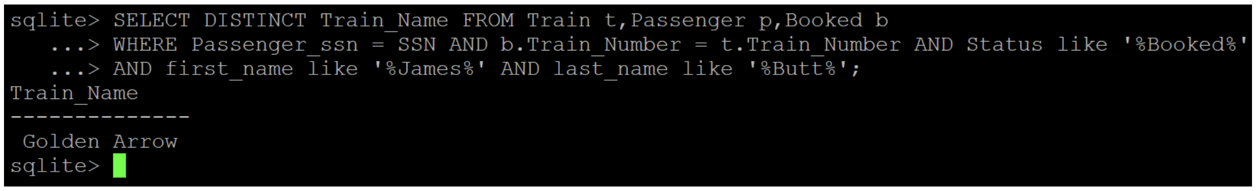
.read importData.sql

5) SOURCE code for SQL SELECT statements AND Query Results

See file named “queries.sql” for the select statements.

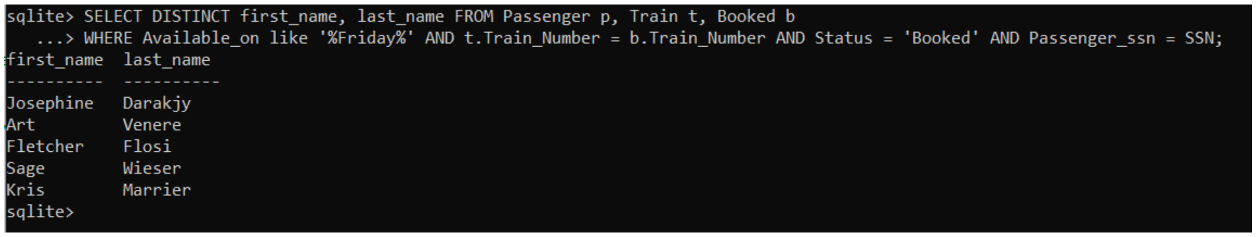
Below are the screenshots of the query result for each query:

2. Input a passenger’s last name and first name and retrieve all trains they are booked on.



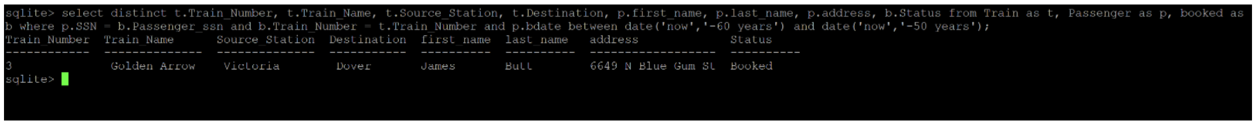
```
sqlite> SELECT DISTINCT Train_Name FROM Train t,Passenger p,Booked b
   ...> WHERE Passenger_ssn = SSN AND b.Train_Number = t.Train_Number AND Status like '%Booked%'
   ...> AND first_name like '%James%' AND last_name like '%Butt%';
Train_Name
--------------
 Golden Arrow
sqlite>
```

3. Input the Day and list the passengers travelling on that day with confirmed tickets.



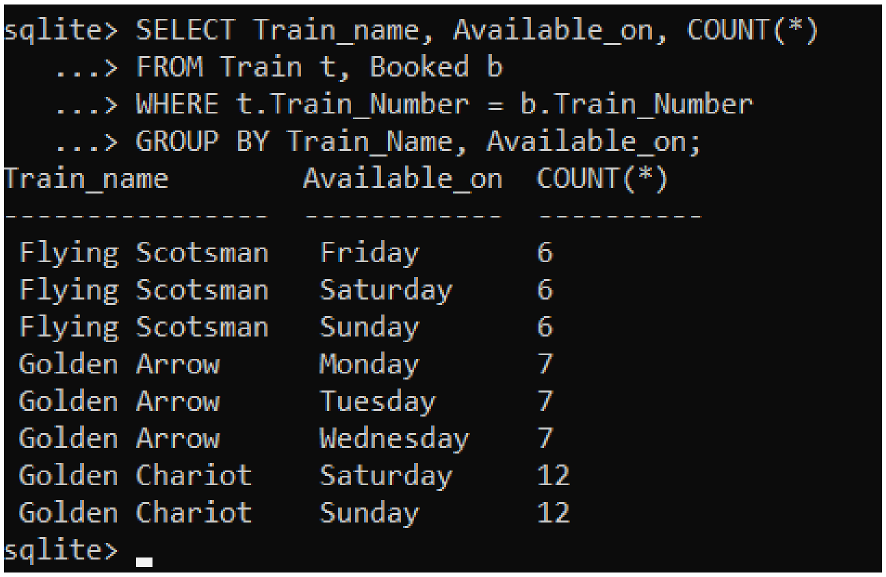
```
sqlite> SELECT DISTINCT first_name, last_name FROM Passenger p, Train t, Booked b
   ...> WHERE Available_on like '%Friday%' AND t.Train_Number = b.Train_Number AND Status = 'Booked' AND Passenger_ssn = SSN;
first_name  last_name
----------  ----------
Josephine   Darakjy
Art         Venere
Fletcher    Flosi
Sage        Wieser
Kris        Marrier
sqlite>
```

4. User input the age of the passenger (50 to 60) and display the train information (Train Number, Train Name, Source and Destination) and passenger information (Name, Address, Category, ticket status) of passengers who are between the ages of 50 to 60.



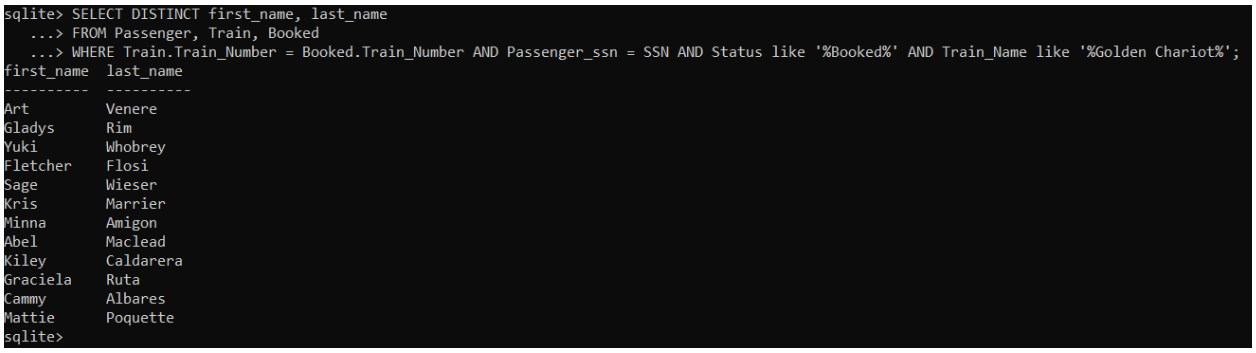
```
sqlite> select distinct t.Train_Number, t.Train_Name, t.Source_Station, t.Destination, p.first_name, p.last_name, p.address, b.Status from Train as t, Passenger as p, booked as
b where p.SSN = b.Passenger_ssn and b.Train_Number = t.Train_Number and p.bdate between date('now','-60 years') and date('now','-50 years');
Train_Number  Train_Name      Source_Station  Destination  first_name  last_name   address             Status
------------  --------------  --------------  -----------  ----------  ----------  ------------------  ----------
3              Golden Arrow    Victoria        Dover        James       Bull        6649 N Blue Gum St  Booked
sqlite>
```

5. List train name, day and number of passenger on that train.



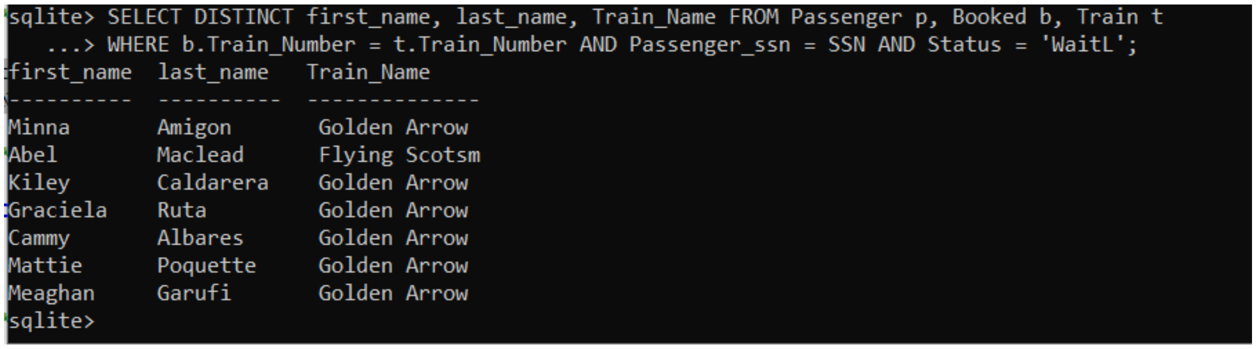
```
sqlite> SELECT Train_name, Available_on, COUNT(*)
   ...> FROM Train t, Booked b
   ...> WHERE t.Train_Number = b.Train_Number
   ...> GROUP BY Train_Name, Available_on;
Train_name        Available_on  COUNT(*)
----------------  ------------  ----------
 Flying Scotsman   Friday       6
 Flying Scotsman   Saturday     6
 Flying Scotsman   Sunday       6
 Golden Arrow      Monday       7
 Golden Arrow      Tuesday      7
 Golden Arrow      Wednesday    7
 Golden Chariot    Saturday     12
 Golden Chariot    Sunday       12
sqlite>
```

6. Enter a train name and retrieve all the passengers with confirmed status traveling in that train.



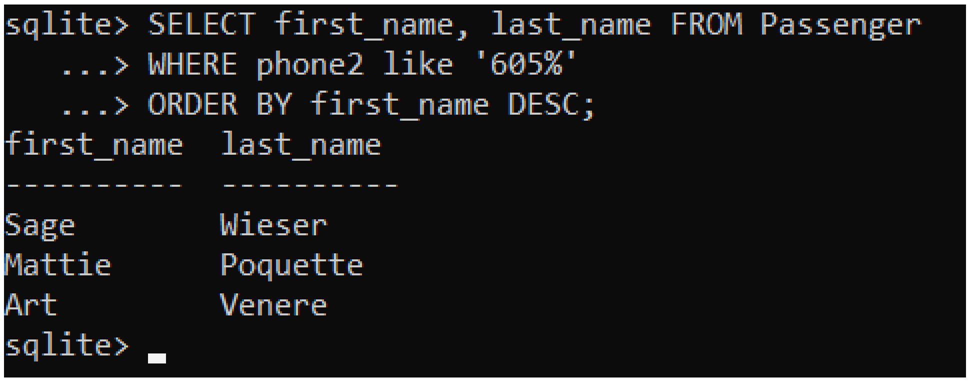
```
sqlite> SELECT DISTINCT first_name, last_name
   ...> FROM Passenger, Train, Booked
   ...> WHERE Train.Train_Number = Booked.Train_Number AND Passenger_ssn = SSN AND Status like '%Booked%' AND Train_Name like '%Golden Chariot%';
first_name  last_name
----------  ----------
Art         Venere
Gladys      Rim
Yuki        Whobrey
Fletcher    Flosi
Sage        Wieser
Kris        Marrier
Minna       Amigon
Abel        Maclead
Kiley       Caldarera
Graciela    Ruta
Cammy       Albares
Mattie      Poquette
sqlite>
```

7. List passengers that are waitlisted including the name of the train.



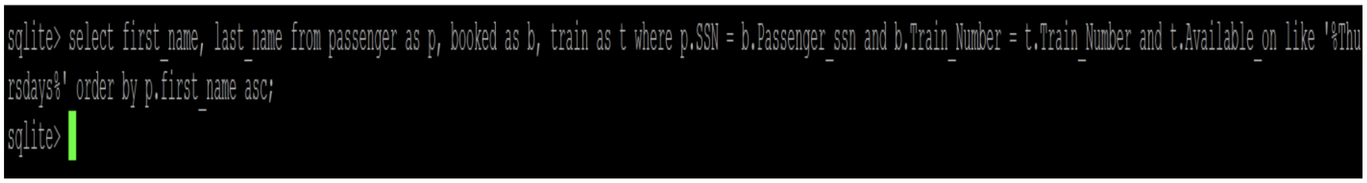
```
sqlite> SELECT DISTINCT first_name, last_name, Train_Name FROM Passenger p, Booked b, Train t
   ...> WHERE b.Train_Number = t.Train_Number AND Passenger_ssn = SSN AND Status = 'WaitL';
first_name  last_name   Train_Name
----------  ----------  -------------
Minna       Amigon       Golden Arrow
Abel        Maclead      Flying Scotsm
Kiley       Caldarera    Golden Arrow
Graciela    Ruta         Golden Arrow
Cammy       Albares      Golden Arrow
Mattie      Poquette     Golden Arrow
Meaghan     Garufi       Golden Arrow
sqlite>
```

8. List passengers that have '605' phone area code in descending order.



```
sqlite> SELECT first_name, last_name FROM Passenger
   ...> WHERE phone2 like '605%'
   ...> ORDER BY first_name DESC;
first_name  last_name
----------  ----------
Sage        Wieser
Mattie      Poquette
Art         Venere
sqlite>
```

9. List name of passengers that are traveling on Thursdays in ascending order.



```
sqlite> select first_name, last_name from passenger as p, booked as b, train as t where p.SSN = b.Passenger_ssn and b.Train_Number = t.Train_Number and t.Available_on like '%Thu
rsdays%' order by p.first_name asc;
sqlite> 
```

NO OUTPUT for this query.

6) CONTRIBUTIONS:

We all worked on the ER diagram together.

Worked together on the 4th and 5th query, and importData.sql.

Siddharth Bhagvagar:

- Worked on the CREATE statements for the BOOKED and TRAIN_STATUS tables.
- Worked on the 1st and 2nd query.
- Contributed to writing this report.

Victor Arowosafe:

- Worked on the CREATE statements for the TRAIN table
- Worked on 3rd and 8th query.
- Contributed to writing this report.

Shafin Barshan:

- Worked on the CREATE statements for the PASSENGER table
- Worked on 6th and 7th query.
- Contributed to writing this report.

HONOR CODE

I pledge, on my honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the pursuit of academic excellence.

I promise that I will submit only work that I personally create or that I contribute to group collaborations, and I will appropriately reference any work from other sources. I will follow the highest standards of integrity and uphold the spirit of the Honor Code.