ExpressRailway Manual

In our program, you will be able to add, edit, and view client data as well as plenty of search options to find the right train for you. There are also admin privileges to allow database alterations.

To run our program: Simply use the makefile provided or follow the steps below:

Make sure you have all the files in the same folder and the appropriate postgresql JAR file (in the example below, .

You can compile the program with this line:

```
javac -cp postgresql-42.2.5.jar ExpressRailway.java
```

IF you already have the program executable or just compiled, you can run the program with this line on windows:

```
java -cp postgresql-42.2.5.jar;. ExpressRailway
```

OR on linux:

```
java -cp postgresql-42.2.5.jar;. ExpressRailway
```

The program is self-explanatory in that it will prompt you will all the choices visually to choose from as described above. You have to type in the number represented by the statement you want to choose. There are integers 1-17 that represent the options to choose from.

The following options are presented below:

Customers

1: add a Customer (option 1.1.1)

- Enter all customer information when prompted and the ID of the customer added will be shown upon success.

2: edit a Customer based on ID number (option 1.1.2)

- After entering a cutstomer_id, it will be verified and if it does not exist in the database, an error will be prompted to the user and the program will exit.

3: = view all the Customers (option 1.1.3)

- This option will view all the customers in the customer table.

Route Trips

4: find a single route trip (option 1.2.1)

- Enter a start station, end station, and time of day and all the routes available will be shown.

5: find a combination route trip (option 1.2.2)

- Enter a start station, end station, and time of day and all the route combinations available will be shown.

In addition to these two route options, the following sorting options are available:

1.2.4.1. Fewest stops
1.2.4.2. Run through most stations
1.2.4.3. Lowest price
1.2.4.4. Highest price 1
1.2.4.5. Least total time
1.2.4.6. Most total time
1.2.4.7. Least total distance
1.2.4.8. Most total distance

Type the digits (1-8) after option 1.2.1 or option 1.2.2 to sort the options by 10 paginated results at a time.

After the routes have been found with options 1.2.1 and options 1.2.2, you can book the routes with option 0, which requires specified route and given day.

Advanced Searches

6: find all the trains that pass through a specific station at a specific day/time

- Enter a specific station, day, and time when prompted and the trains passed through will be shown.

combination (option 1.3.1)

7: find the routes that travel more than one rail line (option 1.3.2)

- List of routes will be shown, no user input taken

8: find the routes that pass through the same stations but don't have the same stops (option 1.3.3)

- List of routes will be shown, no user input taken

9: find any stations through which all trains pass through (option 1.3.4)

- Stations, if any, will be shown, no user input taken

10: find all trains that do not stop at a specific station (option 1.3.5)

- Enter a specified station and the trains will be shown

11: find routes that stop at at least XX% of the Stations they visit (option 1.3.6)

- Enter a percentage number XX and the list of routes will be shown

12: find the schedule of a route (option 1.3.7)

- Enter a routeID and the schedule will be displayed.

13: find availability of a route at every stop on a specific day and time (option 1.3.8)

- Enter a specific routeID, day, and week and the number of seats of available will be shown

Database Admin Options

14: Choose a delimiter ; text file to import into the database (option 2.1)

15: Choose a table and a file path to export data from the database (option 2.2)

16: Delete ALL the rows from ALL the tables (option 2.3)

17: Type 17 to exit.

Simply type the integer in the prompt after the options are displayed. If you type an invalid option, the program will prompt you with an error.

Design Constraints, Limitations, and Improvements

Additional Tables:

Route_Stations(RouteID, StationID)

Rail_Line_Stations(RailID, StationID, distance_between_stations)

Route_Stops(RouteID, StationID)

To store the order of the stations/stops that compose rail lines AND routes, we added two additional tables so that foreign keys may be maintained. Though this does repeat the rail line and route data which is a tradeoff storage cost, it makes the order for the routes and the distance_between_stations for the rail_line_stations easily maintained and the foreign key constraints easily represented to refer to the station_id of the station table. With an array in postgresql, arrays of elements that each are a foreign key is not possible. Thus, the stations in the rail_line and route tables would have not have been updated if operations like deletion or updating occured.

In terms of indices, since the user can display the schedule of a route with option 1.3.7, it made since to make an primary index on the primary key of a schedule, (schedule_route_id, day_of_week, time_of_day). Similarly since the user can view all the customers with the option 1.1.3, it made sense to put a primary index on the customerID when displaying the list of customers in the database.