User Manual for Costa_Project

Introduction:

Costa Express provide great domestic travel experiences through the USA. Through the database system, customers can easily check the detailed travel information including stations, trains, routes, and delay, etc. Also, customers can simply perform operations concerning their travel through the platform in a non-technical way.

Instruction:

- *Welcome*
- *Input your username & password*
    0) Exit <u>*(note: Exit the system and Re-enter the username &password)*</u>
    1) Data admin
        0) Exit
        1) Import data
            0) Back 1) Stations 2) Rail lines 3) Trains 4) Routes 5) route schedule 6) Customers 7) Build legs
        2) Export data
            0) Back 1) Stations 2) Rail lines 3) Trains 4) Routes 5) route schedule 6) customers
        3) Delete database
    2) Passage service Operator
        0) Exit
        1) Customer Form
            0) Back
            1) Add
                → *(Customer Fname? Lname? Street? Town? Postal Code?);*
            2) Edit
                → *(id? ànew Fname, Lname, Street, Town, Postal Code)*
            3) View
                → *(id?)*
        2) Trip search
            → *(what day? Leaving from? Going?)*
            0) Back
            1) Single route
                - Order by?
                    → 1)Price 2)Total Time 3)Stops 4)Stations
                - Search again?
            2) Combination of Routes <u>(WARNING DO NOT RUN THIS IT WORKS BUY IT MAY TAKE HOURS)</u>
                → Max number of stops (<u>WARNING KEEP THIS VERY LOW. OR IT WILL RUN FOR A LONG TIME)</u>
                - Order by?
                    → 1)Price 2)Total Time 3)Stops 4)Stations
                - Search again?

3) Trip search

→(id of customer being booked for:)

(note: id of customers is supposed to be exist in passenger table)

→(id of start station:)

→ (id of end station:)

→ (do you want your ticket to be auto adjusted:))

→ (id of leg being booked for:)

4) Get ticket

5) Advanced search

1) Find all trains that pass through a specific station at a specific day/time

→(what station? What day? What hour 0-23?)

2) Find the routes that travel more than one rail line

3) Rank the trains that are scheduled for more than one route.

4) Find routes that pass through the same stations but do not have the same. stops (WARNING TAKES A LONG TIME)

5) Find any stations through which all trains pass through

6) Find all the trains that do not stop at a specific station

→(What Station?)

7) Find routes that stop at least at XX% of the Stations they visit:

→(What percent (in form 12 = 12%))

8) Display the schedule of a route

→(What the route_id?)

9) Find the availability of a route at every stop on a specific day and time

→(What the route_id? What is the day? What is the time(0-23))

6) triggers

1) will choose the first alternative it can find

2) automatically deletes unpayed for tickets.

Various Difficulty:

- Having difficulty with trigger 1. Also had difficulty with Time variable and knowing when trains arrive.
- Also, some difficulty with various advanced searches requiring me to change the table multiple times.
  - Did you know explicit cursors are not local variables and don't work with recursion. I now do .

System Constrain & Improvement:

- Since I decided to not limit the multi_route to 2 routes it takes a long time to calculate since its traveling salesman which as we know is a long-time complexity.
- The other thing that takes a long time is finding routes with matching stations but different stops.
- Trigger 1 just chooses the first possible route not optimized in any way.