Documentation of solution(code in java) to backend task(ide eclipse)

|  |  |
| --- | --- |
| Author | Initial version |
| Abhishek Pandey  ASE (java developer at cavisson systems) | Contact info: 7909007263  email:abhisheckvarun@gmail.com |

## Datasets provided:

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 id 43431 non-null int64

1 user\_id 43431 non-null int64

2 vehicle\_model\_id 43431 non-null int64

3 package\_id 7550 non-null float64

4 travel\_type\_id 43431 non-null int64

5 from\_area\_id 43343 non-null float64

6 to\_area\_id 34293 non-null float64

7 from\_city\_id 16345 non-null float64

8 to\_city\_id 1588 non-null float64

9 from\_date 43431 non-null object

10 to\_date 25541 non-null object

11 online\_booking 43431 non-null int64

12 mobile\_site\_booking 43431 non-null int64

13 booking\_created 43431 non-null object

14 from\_lat 43338 non-null float64

15 from\_long 43338 non-null float64

16 to\_lat 34293 non-null float64

17 to\_long 34293 non-null float64

18 Car\_Cancellation 43431 non-null int64

dtypes: float64(9), int64(7), object(3)

memory usage: 6.3+ MB

## MY APPROACH

* Above data set has to be saved in database(Postgres sql) using rest api’s fetching data of individual rides with above details.
* After saving it into database the user gets an acknowledgement that your ride information is saved.
* Further if there are any errors while saving it into the database appropriate message to be delivered to the user so that he can track the error and it will be easy to resolve.

## My Implementation

Basically i am implementing the rest API in eclipse and i will be sharing the code with aditi ma’am with this doc.

# MY query optimisation for Brownie points

* As above dataset is in raw format for performing analytical query operation and performiing data analytics this database format has to be changed .
* Of all the parameter saved in data base i would first of all segregate the data based on peak timings and non peak timings.(i.e booking created)
* After which i will find out what part of area most of the booking are done in those peak times(also the type of booking)
* After this any individual city to city route will be stored based on its frequency within those peak time.
* Based on above point i will optimised only those ride which are much frequent and thus needed more attention in terms of number cab driver, availability and intensive to cab driver willing to work for more hrs (trip) in those areas.
* Also there should be a check on to check any specific area where cabs are booked but cancelled more frequently.
* The area precision at any point can be easily fetched by using latitude and longitude of that ride details.
* **I feel there has to be synchronisation with the data science team during backend implementation, although whatever i feel could make it easy for them to perform analytical operation i did.**

## Eclipse Coding related to api’s Using java

Although there could be couples of ways to implement the Rest services related to this task bu i am using core java(with get/post wherever i feel it will be required i will change the doc accordingly)

* I created a restservices which will fetch the above parameters from the gui side and it will save the data of each individual ride in postres sql.
* Upon saving the data this function will return the json format for related result having query optimised and ready to be used by the data science team as directed above.
* Some external jar need to added to the ide as i have used json/gson libraries wherever required.

## Rest services and function/methods in my code implementations attached to this documentation in java.

1. Savedetails
   1. Saving details of each and every ride in the DayRideDetails table in the database.I have used some customised jbdc code(only somewhere which is easily understandable as there are lots of ways for implementing jbdc).
   2. Program in savedetails:
      1. Localinsertdata:Program to store data by passing query parameters same as given in the dataset provided.
      2. This program has INSERTDATA() method which inserts the entries of the dataset in the database using postgre sql code.
2. getPeaktimeRideDetails
   1. This rest service basically calls a program PeaktimeRides which stores the ride details during Peak hours of the day(i have taken 6 to 10 and 15 to 20(24 hrs));
   2. PeaktimeRides ()Program
      1. has functionality of extracting the rides from the database based on the timestamp which has the time of 6 am to 10 am and 15 to 20 pm) and return the details in string format.
3. RideCancellationDetails
   1. This rest service will call CancellationData progrma which stores those ride details which have been cancelled(0) and cancelled ride detail is saved to a jsonArray and returned .
   2. CancellationData ()queries the database for all those ride details which has car\_cancellation set as 1 which means they are cancelled.
4. PackageDetails
   1. This rest service will call PackageDetails function which basically stores details of the package based on the package type provided in the dataset..
   2. PackageDetails() basically segregates all the different type of the package and provide area ,lat long, details of each of the package..

# Implementations Regarding Brownie Points

* In the PeakRides() the String returning in the program will give the data of all those rides which are booked during peak hours.
* The format of the data returned is in json format(JsonArray which can be used by data engineers for visualisation and mapping to respective areas depending upon the lat/long and can decide better (more cab availability) cab facility in those areas.
* Also the details will be segregated based on the travel type (hourly rental, point to point , long distance) which will help Xriders in deciding which type of travel needs to be kept in focus during peak hours for better delivery to the riders.
* With the help of packageDetails we can easily visualise which type of package is having more frequency so that type needs focused and relevant action to be taken accordingly.
* The format of all the rest calls are easy and can be used in data visualisation and mapping comfortably.
* CarCancellation() will give details of the cancelled ride , its return format is again easily used to decide which areas are more frequent to cab cancellation so those area has to be provided more cabs.
* I have implemented the full code keeping some assumptions which can vary from developer to developer keeping in mind what parameter he/she would like to put in main focus so depending upon the business requirement code implementation could be altered/changed for different analytical operations on the provided dataset depending upon the other assumptions.

## Code is attached with the email containing this doc.

# Thanks

Abhishek Pandey,

7909007263.

abhisheckvarun@gmail.com