BUSINESS REQUEST

Goodcabs

Presented by: Raj kumar

Topic Outline

1. Introduction
2. Problem statement
3. Datasets
4. Project workflow
5. Query & Output

Introduction

We offer a cab service.

Goodcabs, a two-year-old cab service company, has established a strong presence in India by focusing on tier-2 cities. It empowers local drivers to earn a sustainable livelihood while delivering exceptional service to passengers. Currently operating in 10 cities, Goodcabs aims to achieve ambitious growth and passenger satisfaction goals in 2024.

Problem statement

CITY-LEVEL FARE AND TRIP SUMMARY REPORT

MONTHLY CITY-LEVEL TRIPS TARGET PERFORMANCE REPORT

CITY-LEVEL REPEAT PASSENGER TRIP FREQUENCY REPORT

IDENTIFY CITIES HIGHEST AND LOWEST TOTAL NEW PASSENGERS

IDENTIFY MONTH WITH HIGHEST REVENUE FOR EACH CITY

REPEAT PASSENGER RATE ANALYSIS

DATABASE

trips_db

1. dim_city
2. dim_date
3. fact_passenger_summary (Aggregated Data)
4. dim_repeat_trip_distribution
5. fact_trips

targets_db

1. city_target_passenger_rating
2. monthly_target_new_passengers
3. monthly_target_trips

Project Workflow

1. Data collection & Preparation
2. Creating Table & Inserting data (SQL)
3. Data Analysis(SQL)
4. Conclusion

Data collection & Prep

1. **trips_db**

   Data base:

2. **targets_db**

   Data base

Creating Table

```sql
--CREATE A TABLE DIM_CITY
create table dim_city(
 city_id varchar(5) PRIMARY KEY,
 city_name varchar(50) Default null

)

--IMPORT DIM_CITY DATA
COPY dim_city(city_id, city_name)
FROM 'C:\RPC13_Input_For_Participants\datasets\csv_files\trips_db\dim_city.csv'
DELIMITER ','
CSV HEADER;


--CREATE TABLE dim_date
CREATE TABLE dim_date(
  date date PRIMARY KEY,
  start_of_month date default null,
  month_name varchar(20) default null,
  day_type varchar(10) default null

)
```

```sql
--Insert the data
INSERT INTO dim_date VALUES ('2024-01-01','2024-01-01','January','Weekday'),('2024-01-02','2024-01-01','January

SELECT * FROM dim_date


--CREATE TABLE dim_repeat_trip_distribution
CREATE TABLE dim_repeat_trip_distribution(
  month date NOT NULL,
  city_id varchar(5) not null,
  trip_counts varchar(10) not null,
  trip_count int  DEFAULT NULL,
  repeat_passenger_count int DEFAULT NULL,
  PRIMARY KEY(month, city_id, trip_count)

)

--IMPORT dim_repeat_trip_distribution
COPY dim_repeat_trip_distribution(month, city_id, trip_countS, trip_count, repeat_passenger_count)
```

```sql
--create a table fact_trips
CREATE TABLE fact_trips(
  trip_id varchar(50) NOT NULL,
  date date NOT NULL,
  city_id varchar(5) NOT NULL,
  passenger_type varchar(10) NOT NULL,
  distance_travelled_km int NOT NULL,
  fare_amount int NOT NULL,
  passenger_rating int NOT NULL,
  driver_rating int NOT NULL,
  PRIMARY KEY (trip_id)


  )

ALTER TABLE fact_trips
RENAME distance_travelled_km to distance_travelled_km

--IMPORT CSV file
COPY fact_trips (trip_id, date, city_id, passenger_type, distance_travelled_km, fare_amount, passenger_rating,
FROM 'C:\RPC13_Input_For_Participants\datasets\csv_files\trips_db\fact_trips.csv'
DELIMITER ','
```

```sql
--CREATE TABLE fact_passenger_summary
CREATE TABLE fact_passenger_summary(
  month date NOT NULL,
  city_id varchar(5) not null,
  new_passenger int NOT NULL,
  repeat_passengers int DEFAULT NULL,
  total_passengers int DEFAULT NULL,
  PRIMARY KEY(month, city_id)

)

ALTER TABLE fact_passenger_summary
RENAME column new_passenger to new_passengers;

--IMPORT CSV file
COPY fact_passenger_summary(month, city_id, new_passengers, repeat_passengers, total_passengers)
FROM 'C:\RPC13_Input_For_Participants\datasets\csv_files\trips_db\fact_passenger_summary.csv'
DELIMITER ','
CSV HEADER

SELECT * FROM fact_passenger_summary
```

```sql
month date NOT NULL,
city_id varchar(5) not null,
target_new_passengers int not null
)
--IMPORT CSV file
COPY monthly_target_new_passengers (month, city_id, target_new_passengers)
from 'C:\RPC13_Input_For_Participants\datasets\csv_files\targets_db\monthly_target_new_passengers.csv'
DELIMITER ','
CSV HEADER


--CREATE TABLE monthly_target_trips
CREATE TABLE monthly_target_trips(
month date not null,
city_id varchar(5) not null,
total_target_trips int not null
)
--IMPORT CSV file
COPY monthly_target_trips (month, city_id, total_target_trips)
from 'C:\RPC13_Input_For_Participants\datasets\csv_files\targets_db\monthly_target_trips.csv'
DELIMITER ','
```

```sql
--IMPORT CSV file
COPY monthly_target_trips (month, city_id, total_target_trips)
from 'C:\RPC13_Input_For_Participants\datasets\csv_files\targets_db\monthly_target_trips.csv'
DELIMITER ','
CSV HEADER

SELECT * FROM fact_trips


--CREATE TABLE target_avg_passenger_rating
CREATE TABLE target_avg_passenger_rating (
  city_id varchar(5) primary key,
  target_avg_passenger_rating float NOT NULL
```

Query

Business Request - 1: City-Level Fare and Trip Summary Report

Generate a report that displays the total trips, average fare per km, average fare per trip, and the percentage contribution of each city's trips to the overall trips. This report will help in assessing trip volume, pricing efficiency, and each city's contribution to the overall trip count.

```sql
--BUSINESS REQUEST - 1: CITY-LEVEL FARE AND TRIP SUMMARY REPORT

--TOTAL TRIPS
SELECT COUNT(trip_id) as total_trip
from fact_trips

--AVERAGE FARE PER KM

select avg(distance_travelled_km) as avg_distance, avg(fare_amount) as avg_fare_amount
from fact_trips

--AVERAGE FARE PER TRIP
select avg(fare_amount) as avg_fare_per_trip
from fact_trips

--PERCENTAGE CONTRIBUTION OF EACH CITY'S TRIPS TO THE OVERALL TRIPS
SELECT dim_city.city_name, count(trip_id),
(count(trip_id)*100 / sum(count(trip_id)) over()) as percentage_contribution
from fact_trips
Inner join dim_city ON dim_city.city_id = fact_trips.city_id
GROUP BY dim_city.city_name
```

Output

	total_trip bigint
1	425903

Data Output | Messages | Notifications

	avg_distance numeric	avg_fare_amount numeric
1	19.1271721495270050	254.0204952770936105

Data Output | Messages | Noti

	avg_fare_per_trip numeric
1	254.0204952770936105

Data Output | Messages | Notifications

	city_name character varying (50)	count bigint	percentage_contribution numeric
1	Chandigarh	38981	9.1525535157066280
2	Coimbatore	21104	4.9551188885732197
3	Indore	42456	9.9684669983540853
4	Jaipur	76888	18.0529369363446606
5	Kochi	50702	11.9045886035083106
6	Lucknow	64299	15.0970995743162175
7	Mysore	16238	3.8126052176199745
8	Surat	54843	12.8768757205279136
9	Vadodara	32026	7.5195525741776883
10	Visakhapatnam	28366	6.6602019708713017

Query

Business Request - 2: Monthly City-Level Trips Target Performance Report

Generate a report that evaluates the target performance for trips at the monthly and city level. For each city and month, compare the actual total trips with the target trips and categorise the performance as follows:

- If actual trips are greater than target trips, mark it as "Above Target".
- If actual trips are less than or equal to target trips, mark it as "Below Target".

Additionally, calculate the % difference between actual and target trips to quantify the performance gap.

```sql
--BUSINESS REQUEST -2: MONTHLY CITY-LEVEL TRIPS TARGET PERFORMANCE REPORT

--Total Actual Trips
with total_actual_trips as (
select dim_city.city_name, dim_city.city_id, TO_CHAR(fact_trips.date, 'YYYY-MM') AS month,
count(fact_trips.trip_id) as actual_trips
from fact_trips
inner join dim_city ON dim_city.city_id = fact_trips.city_id
GROUP BY dim_city.city_name, month, dim_city.city_id
)

--Compare both Actual trips and target trips and additonaly % difference between actual and target trips
select a.city_name, t.city_id, t.month, a.actual_trips, t.total_target_trips,
case
    WHEN a.actual_trips > t.total_target_trips THEN 'Above Target'
    WHEN a.actual_trips < t.total_target_trips OR a.actual_trips = t.total_target_trips THEN 'Below Target'
END AS Performance_category,
ROUND(((a.actual_trips - t.total_target_trips) * 100.0) / t.total_target_trips, 2) as Percentange_difference

FROM total_actual_trips as a
Inner join monthly_target_trips as t ON t.city_id = a.city_id AND t.month = TO_DATE(a.month, 'YYYY-MM')
ORDER BY a.city_name, t.month

```

Output

	city_name character varying (50)	city_id character varying (5)	month date	actual_trips bigint	total_target_trips integer	performance_category text	percentange_difference numeric
1	Chandigarh	CH01	2024-01-01	6810	7000	Below Target	-2.71
2	Chandigarh	CH01	2024-01-01	6810	7000	Below Target	-2.71
3	Chandigarh	CH01	2024-01-01	6810	7000	Below Target	-2.71
4	Chandigarh	CH01	2024-01-01	6810	7000	Below Target	-2.71
5	Chandigarh	CH01	2024-02-01	7387	7000	Above Target	5.53
6	Chandigarh	CH01	2024-02-01	7387	7000	Above Target	5.53
7	Chandigarh	CH01	2024-02-01	7387	7000	Above Target	5.53
8	Chandigarh	CH01	2024-02-01	7387	7000	Above Target	5.53
9	Chandigarh	CH01	2024-03-01	6569	7000	Below Target	-6.16
10	Chandigarh	CH01	2024-03-01	6569	7000	Below Target	-6.16
11	Chandigarh	CH01	2024-03-01	6569	7000	Below Target	-6.16
12	Chandigarh	CH01	2024-03-01	6569	7000	Below Target	-6.16
13	Chandigarh	CH01	2024-04-01	5566	6000	Below Target	-7.23
14	Chandigarh	CH01	2024-04-01	5566	6000	Below Target	-7.23
15	Chandigarh	CH01	2024-04-01	5566	6000	Below Target	-7.23

Output Link

Query

Business Request - 3: City-Level Repeat Passenger Trip Frequency Report

Generate a report that shows the percentage distribution of repeat passengers by the number of trips they have taken in each city. Calculate the percentage of repeat passengers who took 2 trips, 3 trips, and so on, up to 10 trips.

Each column should represent a trip count category, displaying the percentage of repeat passengers who fall into that category out of the total repeat passengers for that city.

This report will help identify cities with high repeat trip frequency, which can indicate strong customer loyalty or frequent usage patterns.

- Fields: city_name, 2-Trips, 3-Trips, 4-Trips, 5-Trips, 6-Trips, 7-Trips, 8-Trips, 9-Trips, 10-Trips

```sql
--BUSINESS REQUEST 3: CITY-LEVEL REPEAT PASSENGER TRIP FREQUENCY REPORT

--Total Passenger Count Per Month each City
with total_customer as(
select month, city_id, sum(repeat_passenger_count) as total_customer_counts
from dim_repeat_trip_distribution
Group by month, city_id
),

--Percentage of each trip_counts
Percentage_distribution as (
select dm.month, dm.city_id, dm.trip_counts, dm.repeat_passenger_count, tc.total_customer_counts,
ROUND((dm.repeat_passenger_count::Numeric/ tc.total_customer_counts) * 100, 2) as percentage_distribution
FROM dim_repeat_trip_distribution as dm
INNER JOIN total_customer AS tc ON tc.city_id = dm.city_id and tc.month = dm.month
)

--Final Update
SELECT month, city_id, trip_counts, repeat_passenger_count, total_customer_counts,percentage_distribution
FROM Percentage_distribution
ORDER BY month, city_id, trip_counts;
```

Output

Data Output | Messages | Notifications

	month date	city_id character varying (5)	trip_counts character varying (10)	repeat_passenger_count integer	total_customer_counts bigint	percentage_distribution numeric
1	2024-01-01	AP01	10-Trips	7	650	1.08
2	2024-01-01	AP01	2-Trips	352	650	54.15
3	2024-01-01	AP01	3-Trips	158	650	24.31
4	2024-01-01	AP01	4-Trips	53	650	8.15
5	2024-01-01	AP01	5-Trips	38	650	5.85
6	2024-01-01	AP01	6-Trips	14	650	2.15
7	2024-01-01	AP01	7-Trips	10	650	1.54
8	2024-01-01	AP01	8-Trips	11	650	1.69
9	2024-01-01	AP01	9-Trips	7	650	1.08
10	2024-01-01	CH01	10-Trips	14	720	1.94
11	2024-01-01	CH01	2-Trips	196	720	27.22
12	2024-01-01	CH01	3-Trips	182	720	25.28
13	2024-01-01	CH01	4-Trips	87	720	12.08
14	2024-01-01	CH01	5-Trips	119	720	16.53
15	2024-01-01	CH01	6-Trips	44	720	6.11
16	2024-01-01	CH01	7-Trips	33	720	4.58

output_link

query

Business Request - 4: Identify Cities with Highest and Lowest Total New Passengers

Generate a report that calculates the total new passengers for each city and ranks them based on this value. Identify the top 3 cities with the highest number of new passengers as well as the bottom 3 cities with the lowest number of new passengers, categorising them as "Top 3" or "Bottom 3" accordingly.

Fields

- city_name
- total_new_passengers
- city_category ("Top 3" or "Bottom 3")

```
225
226  --BUSINESS REQUIRMENTS 4: IDENTIFY CITIES HIGHEST AND LOWEST TOTAL NEW PASSENGERS
227
228  With rank_city as (
229  SELECT dim_city.city_name, fact_passenger_summary.month, fact_passenger_summary.new_passengers,
230  RANK()OVER (ORDER BY fact_passenger_summary.new_passengers Desc) as rank_desc,
231  RANK()OVER (ORDER BY fact_passenger_summary.new_passengers ASC) as rank_asc
232  From fact_passenger_summary
233  INNER JOIN dim_city ON  dim_city.city_id = fact_passenger_summary.city_id
234  )
235
236  select rank_city.city_name, rank_city.month, rank_city.new_passengers, 'TOP_3' as City_category
237  from rank_city
238  Where rank_city.rank_desc <=3
239
240  union all
241
242  select rank_city.city_name, rank_city.month, rank_city.new_passengers, 'bottom_3' as City_category
243  from rank_city
244  Where rank_city.rank_asc <=3
245
```

Output

Data Output | Messages | Notifications

	city_name character varying (50)	month date	new_passengers integer	city_category text
1	Jaipur	2024-03-01	7417	TOP_3
2	Jaipur	2024-01-01	10423	TOP_3
3	Jaipur	2024-02-01	10789	TOP_3
4	Coimbatore	2024-05-01	1039	bottom_3
5	Vadodara	2024-06-01	1104	bottom_3
6	Coimbatore	2024-06-01	1226	bottom_3

Query

```sql
--BUSINESS REQUEST 5 (IDENTIFY MONTH WITH HIGHEST REVENUE FOR EACH CITY)

--City_Month_Wise_Revenue
with city_revenue as (
select dim_city.city_name, fact_trips.city_id, TO_CHAR(fact_trips.date, 'YYYY-MM')AS month,
sum(fact_trips.fare_amount) as month_revenue
From fact_trips
INNER JOIN dim_city ON dim_city.city_id = fact_trips.city_id
Group by month,dim_city.city_name,fact_trips.city_id
),

--City_total_month_revenue
City_total_monthly_revenue as (
select sum(cr.month_revenue) as total_revenue,cr.month
from city_revenue as cr
Group by month
),

--City_monthly_revenue_percentage_distribution
city_percentage_dist as (
select cr.month, cr.month_revenue, cr.city_name,
ROUND((cr.month_revenue::Numeric/City_total_monthly_revenue.total_revenue ) * 100, 2) as percentage_distributio
from city_revenue as cr
INNER JOIN City_total_monthly_revenue ON City_total_monthly_revenue.month = cr.month
),

--City_Max_Revenue_Each_Month
city_max_revenue as(
select  month, month_revenue, city_name,
RANK() OVER (PARTITION BY cr.month ORDER BY cr.month_revenue DESC) AS rank
from city_revenue as cr
)

--Final_Result
SELECT cmr.month, cmr.city_name, cmr.month_revenue AS highest_revenue, cpd.percentage_distribution
FROM city_max_revenue AS cmr
INNER JOIN city_percentage_dist  AS cpd ON cmr.city_name = cpd.city_name AND cmr.month = cpd.month
WHERE cmr.rank = 1
ORDER BY cmr.month;
```

Output

Business Request - 5: Identify Month with Highest Revenue for Each City

Generate a report that identifies the month with the highest revenue for each city. For each city, display the month_name, the revenue amount for that month, and the percentage contribution of that month's revenue to the city's total revenue.

Fields

- city_name
- highest_revenue_month
- revenue
- percentage_contribution (%)

Data Output | Messages | Notifications

	month text	city_name character varying (50)	highest_revenue bigint	percentage_distribution numeric
1	2024-01	Jaipur	7223310	39.14
2	2024-02	Jaipur	7747202	39.01
3	2024-03	Jaipur	6462092	34.31
4	2024-04	Jaipur	5490146	31.03
5	2024-05	Jaipur	5495976	30.56
6	2024-06	Jaipur	4788771	31.18

Query

Business Request - 6: Repeat Passenger Rate Analysis

Generate a report that calculates two metrics:

1. Monthly Repeat Passenger Rate: Calculate the repeat passenger rate for each city and month by comparing the number of repeat passengers to the total passengers.
2. City-wide Repeat Passenger Rate: Calculate the overall repeat passenger rate for each city, considering all passengers across months.

These metrics will provide insights into monthly repeat trends as well as the overall repeat behaviour for each city.

Fields:

- city_name
- month
- total_passengers
- repeat_passengers
- monthly_repeat_passenger_rate (%): Repeat passenger rate at the city and month level
- city_repeat_passenger_rate (%): Overall repeat passenger rate for each city, aggregated across months

---

```
292  --BUSINESS REQUEST 6(REPEAT PASSENGER RATE ANALYSIS)
293
294  --Case -1
295  select fact_passenger_summary.month, fact_passenger_summary.city_id,dim_city.city_name,
296  fact_passenger_summary.total_passengers,fact_passenger_summary.repeat_passengers,
297  ROUND((fact_passenger_summary.repeat_passengers::Numeric/fact_passenger_summary.total_passengers) * 100, 2) as
298  from fact_passenger_summary
299  INNER JOIN dim_city ON dim_city.city_id = fact_passenger_summary.city_id
300
301  --Case-2
302  select fact_passenger_summary.city_id, dim_city.city_name,
303  sum(fact_passenger_summary.repeat_passengers) as total_repeat_passengers,
304  sum(fact_passenger_summary.total_passengers) as total_passengers_month,
305  ROUND ((sum(repeat_passengers)::numeric/sum(total_passengers))*100, 2) as percentage_distribution
306  from fact_passenger_summary
307  INNER JOIN dim_city ON dim_city.city_id = fact_passenger_summary.city_id
308  group by fact_passenger_summary.city_id,dim_city.city_name
309
```

Output

Data Output | Messages | Notifications

	month date	city_id character varying (5)	city_name character varying (50)	total_passengers integer	repeat_passengers integer	percentage_distribution numeric
1	2024-01-01	AP01	Visakhapatnam	3163	650	20.55
2	2024-01-01	CH01	Chandigarh	4640	720	15.52
3	2024-01-01	GJ01	Surat	3616	1184	32.74
4	2024-01-01	GJ02	Vadodara	2633	544	20.66
5	2024-01-01	KA01	Mysore	2129	172	8.08
6	2024-01-01	KL01	Kochi	5660	795	14.05
7	2024-01-01	MP01	Indore	3876	1033	26.65
8	2024-01-01	RJ01	Jaipur	11845	1422	12.01
9	2024-01-01	TN01	Coimbatore	2214	392	17.71
10	2024-01-01	UP01	Lucknow	4896	1431	29.23
11	2024-02-01	AP01	Visakhapatnam	3170	790	24.92
12	2024-02-01	CH01	Chandigarh	4957	853	17.21
13	2024-02-01	GJ01	Surat	3567	1313	36.81
14	2024-02-01	GJ02	Vadodara	2756	610	22.13
15	2024-02-01	KA01	Mysore	2290	183	7.99
16	2024-02-01	KL01	Kochi	5372	1005	18.71

Data Output | Messages | Notifications

	city_id character varying (5)	city_name character varying (50)	total_repeat_passengers bigint	total_passengers_month bigint	percentage_distribution numeric
1	GJ02	Vadodara	4346	14473	30.03
2	KA01	Mysore	1477	13158	11.23
3	GJ01	Surat	8638	20264	42.63
4	UP01	Lucknow	9597	25857	37.12
5	KL01	Kochi	7626	34042	22.40
6	MP01	Indore	7216	22079	32.68
7	TN01	Coimbatore	2551	11065	23.05
8	RJ01	Jaipur	9682	55538	17.43
9	AP01	Visakhapatnam	5108	17855	28.61
10	CH01	Chandigarh	5070	23978	21.14

output link

Thank you