Ad-Hoc Request

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 Query:

```
with cte as(
    select
        distinct(city_name) as city_name,
        count(trip_id) as Total_trips,
        sum(fare_amount) as Total_Revenue,
        sum(distance_travelled_km) as Total_KM
     from fact_trips f
     join dim_city c ON f.city_id=c.city_id
     group by city_name
),Total_trips_cte as
(
    select sum(Total_trips) as Grand_total_trips
    From cte
)
 select
     c.city_name,c.total_trips,
     round((c.Total_Revenue/c.Total_KM),2) as Avg_fare_per_KM,
     round((c.Total_Revenue/c.Total_trips),2) as Avg_fare_per_trip,
     round((c.Total_trips/t.Grand_total_trips)*100,2) as Trip_contribution
 From cte c
 cross join Total_trips_cte t;
```

OUTPUT:

city_name	total_trips	Avg_fare_per_K	Avg_fare_p	Trip_contribution
Chandigarh	38981	12.06	283.69	9.15
Coimbatore	21104	11.15	166.98	4.96
Indore	42456	10.9	179.84	9.97
Jaipur	76888	16.12	483.92	18.05
Kochi	50702	13.93	335.25	11.9
Lucknow	64299	11.76	147.18	15.1
Mysore	16238	15.14	249.71	3.81
Surat	54843	10.66	117.27	12.88
Vadodara	32026	10.29	118.57	7.52
Visakhapatnam	28366	12.53	282.67	6.66

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 Query:

```
with cte as(
select
    distinct(city_name) as city_name,
    monthname(date) as Month_name,
    max(total_target_trips) as Target_trips,
    count(f.trip_id) as Actual_trips
from fact_trips f
Join dim_city c ON f.city_id=c.city_id
join targets_db.monthly_target_trips t
ON t.city_id=f.city_id and monthname(t.month)=monthname(f.date)
group by city_name,Month_name
),cte2 as
(
select *,
Case
    When Actual_trips >  Target_trips then 'Above target'
    When Actual_trips < Target_trips then 'Below target'
    Else 'On Target'
end as Performance_status,
round(((Actual_trips-Target_trips)/Target_trips)*100,2) as pct_differnce,
case When Month_name = 'January' Then 1
     When Month_name = 'February' Then 2
     When Month_name ='March' Then 3
     When Month_name = 'April' Then 4
     When Month_name = 'May' Then 5
     When Month_name = 'June' Then 6
     else 7
End as Month_no
From cte
)
Select city_name,Month_name,Target_trips,Actual_trips,Performance_status,pct_differnce
 From cte2
order by City_name,Month_no
```

OUTPUT:

city_name	Month_name	Target_trips	Actual_trips	Performance_status	pct_differnce
Chandigarh	January	7000	6810	Below target	-2.71
Chandigarh	February	7000	7387	Above target	5.53
Chandigarh	March	7000	6569	Below target	-6.16
Chandigarh	April	6000	5566	Below target	-7.23
Chandigarh	May	6000	6620	Above target	10.33
Chandigarh	June	6000	6029	Above target	0.48
Coimbatore	January	3500	3651	Above target	4.31
Coimbatore	February	3500	3404	Below target	-2.74
Coimbatore	March	3500	3680	Above target	5.14
Coimbatore	April	3500	3661	Above target	4.6
Coimbatore	May	3500	3550	Above target	1.43

Coimbatore	June	3500	3158	Below target	-9.77
Indore	January	7000	6737	Below target	-3.76
Indore	February	7000	7210	Above target	3
Indore	March	7000	7019	Above target	0.27
Indore	April	7500	7415	Below target	-1.13
Indore	May	7500	7787	Above target	3.83
Indore	June	7500	6288	Below target	-16.16
Jaipur	January	13000	14976	Above target	15.2
Jaipur	February	13000	15872	Above target	22.09
Jaipur	March	13000	13317	Above target	2.44
Jaipur	April	9500	11406	Above target	20.06
Jaipur	May	9500	11475	Above target	20.79
Jaipur	June	9500	9842	Above target	3.6
Kochi	January	7500	7344	Below target	-2.08
Kochi	February	7500	7688	Above target	2.51
Kochi	March	7500	9495	Above target	26.6
Kochi	April	9000	9762	Above target	8.47
Kochi	May	9000	10014	Above target	11.27
Kochi	June	9000	6399	Below target	-28.9
Lucknow	January	13000	10858	Below target	-16.48
Lucknow	February	13000	12060	Below target	-7.23
Lucknow	March	13000	11224	Below target	-13.66
Lucknow	April	11000	10212	Below target	-7.16
Lucknow	May	11000	9705	Below target	-11.77
Lucknow	June	11000	10240	Below target	-6.91
Mysore	January	2000	2485	Above target	24.25
Mysore	February	2000	2668	Above target	33.4
Mysore	March	2000	2633	Above target	31.65
Mysore	April	2500	2603	Above target	4.12
Mysore	May	2500	3007	Above target	20.28
Mysore	June	2500	2842	Above target	13.68
Surat	January	9000	8358	Below target	-7.13
Surat	February	9000	9069	Above target	0.77
Surat	March	9000	9267	Above target	2.97
Surat	April	10000	9831	Below target	-1.69
Surat	May	10000	9774	Below target	-2.26
Surat	June	10000	8544	Below target	-14.56
Vadodara	January	6000	4775	Below target	-20.42
Vadodara	February	6000	5228	Below target	-12.87
Vadodara	March	6000	5598	Below target	-6.7
Vadodara	April	6500	5941	Below target	-8.6
Vadodara	May	6500	5799	Below target	-10.78
Vadodara	June	6500	4685	Below target	-27.92
Visakhapatnam	January	4500	4468	Below target	-0.71
Visakhapatnam	February	4500	4793	Above target	6.51
Visakhapatnam	March	4500	4877	Above target	8.38
Visakhapatnam	April	5000	4938	Below target	-1.24
Visakhapatnam	May	5000	4812	Below target	-3.76
Visakhapatnam	June	5000	4478	Below target	-10.44

3. 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.

SQL Query:

```
with cte1 as(
select
    city_name,
    trip_count,
    sum(repeat_passenger_count) as Repeat_Passenger,
    sum(sum(repeat_passenger_count)) over(partition by city_name) as Total_Passenger
 from dim_repeat_trip_distribution r
 join dim_city c
 ON c.city_id=r.city_id
 group by city_name,trip_count
 order by city_name,trip_count
), cte2 as(
select
    city_name,
    case
        when trip_count='2-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        when trip_count='3-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        when trip_count='4-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        when trip_count='5-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        when trip_count='6-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        when trip_count='7-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        when trip_count='8-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        when trip_count='9-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        when trip_count='10-Trips' then round((Repeat_Passenger/Total_Passenger)*100,0)
        end as Percentage,
        trip_count
From cte1
)
select
    city_name,
    max(CASE WHEN trip_count = '2-Trips' THEN percentage END) AS "2-Trips",
    max(CASE WHEN trip_count = '3-Trips' THEN percentage END) AS "3-Trips",
    max(CASE WHEN trip_count = '4-Trips' THEN percentage END) AS "4-Trips",
    max(CASE WHEN trip_count = '5-Trips' THEN percentage END) AS "5-Trips",
    max(CASE WHEN trip_count = '6-Trips' THEN percentage END) AS "6-Trips",
    max(CASE WHEN trip_count = '7-Trips' THEN percentage END) AS "7-Trips",
    max(CASE WHEN trip_count = '8-Trips' THEN percentage END) AS "8-Trips",
    max(CASE WHEN trip_count = '9-Trips' THEN percentage END) AS "9-Trips",
    max(CASE WHEN trip_count = '10-Trips' THEN percentage END) AS "10-Trips"
From cte2
group by city_name
order by city_name;
```

OUTPUT:

city_name	2-Trips	3-Trips	4-Trips	5-Trips	6-Trips	7-Trips	8-Trips	9-Trips	10-Trips
Chandigarh	32	19	16	12	7	5	3	2	2
Coimbatore	11	15	16	21	18	10	6	2	1
Indore	34	23	13	10	7	5	3	2	2
Jaipur	50	21	12	6	4	3	2	1	1
Kochi	48	24	12	6	4	2	2	1	1
Lucknow	10	15	16	18	20	11	6	2	1
Mysore	49	24	13	6	4	2	1	1	0
Surat	10	14	17	20	18	12	6	2	1
Vadodara	10	14	17	18	19	13	6	2	2
Visakhapatnam	51	25	10	5	3	2	1	1	1

4. 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.

SQL Query:

```
(SELECT
        city_name,
        sum(new_passengers) as total_New_Passenger,
        concat('Top',ROW_NUMBER() OVER (ORDER BY SUM(new_passengers) DESC)) AS city_category
FROM fact_passenger_summary p
join dim_city c
ON c.city_id=p.city_id
group by city_name
order by total_New_Passenger desc
Limit 3)
union all
(SELECT
        city_name,
        sum(new_passengers) as New_Passenger,
        concat('Bottom',ROW_NUMBER() OVER (ORDER BY SUM(new_passengers) asc)) AS city_category
FROM fact_passenger_summary p
join dim_city c
ON c.city_id=p.city_id
group by city_name
order by new_passenger asc
Limit 3)
```

OUTPUT:

city_name	total_New_Passenger	city_category
Jaipur	45856	Top1
Kochi	26416	Top2
Chandigarh	18908	Top3
Coimbatore	8514	Bottom1
Vadodara	10127	Bottom2
Surat	11626	Bottom3

5. 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.

SQL Query:

```
with cte as(SELECT
    city_name,
    MONTHNAME(date) AS Month_Name,
    ROUND((SUM(fare_amount) / 1000000),2) AS Revenue
FROM fact_trips t
JOIN dim_city c ON t.city_id = c.city_id
GROUP BY city_name, MONTHNAME(date)
), cte2 as(
select *,
sum(Revenue) OVER (PARTITION BY city_name) AS Total_Revenue,
RANK() OVER (PARTITION BY city_name ORDER BY revenue DESC) as Rank_no
From  cte

)
select
        City_name,
        Month_Name as Higest_Revenue_month,
        Revenue,
        ROUND((revenue / total_revenue) * 100, 2) AS percentage_contribution
From cte2
where Rank_no = 1
```

OUTPUT:

City_name	Higest_Revenue_month	Revenue	percentage_contribution
Chandigarh	February	2.11	19.1
Coimbatore	April	0.61	17.33
Coimbatore	January	0.61	17.33
Coimbatore	March	0.61	17.33
Indore	May	1.38	18.04
Jaipur	February	7.75	20.83
Kochi	May	3.33	19.59
Lucknow	February	1.78	18.8
Mysore	May	0.75	18.47
Surat	April	1.15	17.88
Vadodara	April	0.71	18.73
Visakhapatnam	April	1.39	17.35
Visakhapatnam	March	1.39	17.35

6. 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.

SQL Query:

```
with cte as(
select
        city_name,
        monthname(month) as Month_name,
        sum(total_passengers) as Total_Passengers,
        sum(repeat_passengers) as Repeat_Passengers
from fact_passenger_summary p
join dim_city c
ON c.city_id=p.city_id
group by city_name,Month_name
),cte2 as(
select city_name,
        sum(total_passengers) as Total_city_Passengers,
        sum(Repeat_Passengers) as Total_city_Repeat_Passengers
From cte
Group by city_name
)
select cte.*,
      round((Repeat_Passengers/Total_Passengers)*100,0) as Monthly_repeat_Passenger_rate,
      round((Total_city_Repeat_Passengers/Total_city_Passengers)*100,0) as city_repeat_Passenger_rate
From cte
join cte2 ON cte.city_name=cte2.city_name
```

OUTPUTS:

city name	Month name	Total Passengers	Repeat Passengers	Monthly repeat Passenger rate	City repeat Passenger rate
Visakhapatnam	January	3163	650	21	29
Visakhapatnam	February	3170	790	25	29
Visakhapatnam	March	3093	923	30	29
Visakhapatnam	April	2837	992	35	29
Visakhapatnam	May	2890	951	33	29
Visakhapatnam	June	2702	802	30	29
Chandigarh	January	4640	720	16	21
Chandigarh	February	4957	853	17	21
Chandigarh	March	4100	872	21	21
Chandigarh	April	3285	789	24	21
Chandigarh	May	3699	969	26	21
Chandigarh	June	3297	867	26	21
Surat	January	3616	1184	33	43
Surat	February	3567	1313	37	43
Surat	March	3440	1494	43	43
Surat	April	3394	1551	46	43
Surat	May	3217	1606	50	43
Surat	June	3030	1490	49	43
Vadodara	January	2633	544	21	30
Vadodara	February	2756	610	22	30
Vadodara	March	2522	759	30	30

Vadodara	April	2499	862	34	30
Vadodara	May	2256	868	38	30
Vadodara	June	1807	703	39	30
Mysore	January	2129	172	8	11
Mysore	February	2290	183	8	11
Mysore	March	2194	208	9	11
Mysore	April	2072	236	11	11
Mysore	May	2270	349	15	11
Mysore	June	2203	329	15	11
Kochi	January	5660	795	14	22
Kochi	February	5372	1005	19	22
Kochi	March	6213	1348	22	22
Kochi	April	6515	1576	24	22
Kochi	May	6222	1853	30	22
Kochi	June	4060	1049	26	22
Indore	January	3876	1033	27	33
Indore	February	3981	1103	28	33
Indore	March	3833	1091	28	33
Indore	April	3646	1295	36	33
Indore	May	3591	1563	44	33
Indore	June	3152	1131	36	33
Jaipur	January	11845	1422	12	17
Jaipur	February	12450	1661	13	17
Jaipur	March	9257	1840	20	17
Jaipur	April	7856	1736	22	17
Jaipur	May	7174	1842	26	17
Jaipur	June	6956	1181	17	17
Coimbatore	January	2214	392	18	23
Coimbatore	February	1993	346	17	23
Coimbatore	March	1965	427	22	23
Coimbatore	April	1722	480	28	23
Coimbatore	May	1543	504	33	23
Coimbatore	June	1628	402	25	23
Lucknow	January	4896	1431	29	37
Lucknow	February	5188	1659	32	37
Lucknow	March	4781	1622	34	37
Lucknow	April	3807	1496	39	37
Lucknow	May	3487	1662	48	37
Lucknow	June	3698	1727	47	37