Uber Trip Amnlysis using SQL – 40 Questions & Answers

1. Display all records:

```
query="select * from tripdetail"

result=pd.read_sql(query,conn)

display(result)
```

2. Count total trips:

```
query= "select count(*) from tripdetail"

result=pd.read_sql(query,conn)

display(result)
```

3. List unique vehicle types:

```
query= "select distinct vehicle from tripdetail"

result=pd.read_sql(query,conn)

display(result)
```

4. Trips with fare > 100:

```
query= "select * from tripdetail where fare_amount>100"

result=pd.read_sql(query,conn)

display(result)
```

5. Uber Black trips:

```
query= "select * from tripdetail where Vehicle='Uber Black'"

result=pd.read_sql(query,conn)

display(result)
```

6. Total passengers:

```
query= "select sum(passenger_count) as 'Total Passengers' from tripdetail"

result=pd.read_sql(query,conn)

display(result)
```

7. Average trip distance:

```
query= "select avg(trip_distance) as 'Avg Trip Distance' from tripdetail"

result=pd.read_sql(query,conn)
```

```
display(result)
```

8. Total fare per vehicle:

```
query= "select vehicle, sum(fare_amount) as 'Total Fare' from tripdetail group by vehicle "

result=pd.read_sql(query,conn)

display(result)
```

9. Average fare per vehicle:

```
query= "select vehicle, avg(fare_amount) as 'Avg Fare_Amount' from tripdetail group by vehicle"

result=pd.read_sql(query,conn)

display(result)
```

10. Payment methods where avg fare > 200:

```
query= "select Payment_type,avg(fare_amount) as Avg_fare from tripdetail group by payment_type having avg(fare_amount)>200"

result=pd.read_sql(query,conn)

display(result)
```

11. Trip count per payment type:

```
query= "select Payment_type, count(*) as 'Trip Count' from tripdetail group by payment_type "

result=pd.read_sql(query,conn)

display(result)
```

12. Highest fare with vehicle:

```
query= "select vehicle, max(fare_amount) as 'Highest Fare Amount' from tripdetail group by vehicle"

result=pd.read_sql(query,conn)

display(result)
```

13. Trips + avg fare per vehicle:

```
query= "select vehicle, count(*) as 'Trips', avg(fare_amount) from tripdetail group by vehicle"

result=pd.read_sql(query,conn)

display(result)
```

14. Top 5 longest trips:

```
query= "select * from tripdetail order by trip_distance desc limit 5"

result=pd.read_sql(query,conn)
```

```
display(result)
```

15. Bottom 5 shortest trips:

```
query= "select * from tripdetail order by trip_distance limit 5"

result=pd.read_sql(query,conn)

display(result)
```

16. Trips with fare > average fare:

```
query= "select * from tripdetail where fare_amount>(select avg(fare_amount) from tripdetail)"

result=pd.read_sql(query,conn)

display(result)
```

17. Trips longer than avg distance:

```
query= "select * from tripdetail where trip_distance > (select avg(trip_distance) from tripdetail) "

result=pd.read_sql(query,conn)

display(result)
```

18. Vehicles with total fare > overall avg fare:

```
query= "select vehicle, fare_amount from tripdetail group by vehicle having sum(fare_amount) >
(select avg(fare_amount) from tripdetail)"

result=pd.read_sql(query,conn)

display(result)
```

19. Vehicles with above-average trip count:

```
SELECT Vehicle, COUNT(*) FROM TripDetails GROUP BY Vehicle HAVING COUNT(*) > (SELECT
AVG(cnt) FROM (SELECT COUNT(*) cnt FROM TripDetails GROUP BY Vehicle));
```

20. Trips with fare > min UberX fare:

```
query= "select * from tripdetail where fare_amount > (select min(fare_amount) from tripdetail where
vehicle='uberx')"

result=pd.read_sql(query,conn)

display(result)
```

21. Trips in June 2024:

```
query = "select * from tripdetail where strftime('%Y-%m', Pickup_Time) = '2024-06'"

result = pd.read_sql(query, conn)

display(result)
```

22. Trips per day:

```
query = "select strftime('%Y-%m-%d', Pickup_Time) as Day, count(*) from tripdetail group by day"

result = pd.read_sql(query, conn)

display(result)
```

23. Extract hour from pickup time:

```
query = "select strftime('%H', Pickup_Time) as Hour, count(*) from tripdetail group by Hour"

result = pd.read_sql(query, conn)

display(result)
```

24. Average trip duration (minutes):

```
query = "select avg((julianday (drop_off_time) - julianday(Pickup_Time))*1440) from tripdetail"

result = pd.read_sql(query, conn)

display(result)
```

25. Earliest & latest trip times:

```
query = "select min(pickup_time),max(pickup_time) from tripdetail"

result = pd.read_sql(query, conn)

display(result)
```

26. Weekend trip count:

```
query = "select count(*) from tripdetail where strftime('%w', Pickup_Time) in ('0','6') "

result = pd.read_sql(query, conn)

display(result)
```

27. Monthly average fare:

```
query = "select avg(fare_amount),strftime('%m', Pickup_Time) as Month from tripdetail group by month "

result = pd.read_sql(query, conn)

display(result)
```

28. Rank trips by fare:

```
query = "select trip_ID, fare_amount, rank() over (order by fare_amount desc) FROM TripDetail"

result = pd.read_sql(query, conn)

display(result)
```

29. Top 10 longest trips:

```
query = "select * from tripdetail order by trip_distance desc limit 10"

result = pd.read_sql(query, conn)

display(result)
```

30. Fare percentage share per vehicle:

```
query = "select vehicle,sum(fare_amount), round(sum(fare_amount)*100.0/(select
sum(fare_amount) from tripdetail),2) as Percentage from tripdetail group by vehicle"

result = pd.read_sql(query, conn)

display(result)
```

31. Compare weekday vs weekend avg fare:

```
query = "select case when strftime('%w', pickup_time) in ('0','6') then 'weekend' else 'weekday' end
as daytype, avg(fare_amount) from tripdetail group by daytype"

result = pd.read_sql(query, conn)

display(result)
```

32. Pickup location with highest avg distance:

```
query = "select pulocationid, avg(trip_distance) from tripdetail group by pulocationid order by
avg(trip_distance) desc limit 1;"

result = pd.read_sql(query, conn)

display(result)
```

33. Revenue per vehicle per month:

```
query = "select vehicle, strftime('%m',pickup_time) as month, sum(fare_amount) from tripdetail group
by vehicle,month"

result = pd.read_sql(query, conn)

display(result)
```

34. Vehicles with long trips across multiple months:

```
query = "select vehicle, trip_distance from tripdetail where trip_distance > (select avg(trip_distance)
from tripdetail) group by vehicle having count(distinct strftime('%y-%m', pickup_time)) > 1;"

result = pd.read_sql(query, conn)

display(result)
```

35. Count trips where tip_amount > fare_amount*0.20:

```
SELECT COUNT(*) FROM TripDetails WHERE tip_amount > fare_amount*0.20;
```

36. Highest Fare trip:

```
query = "select vehicle,max(fare_amount) from tripdetail order by fare_amount desc limit 1"

result = pd.read_sql(query, conn)

display(result)
```

37. Count trips with passenger_count > 3:

```
query = "select count(*) from tripdetail where passenger_count>3"

result = pd.read_sql(query, conn)

display(result)
```

38. Average distance per passenger count:

```
query = "select avg(trip_distance), passenger_count from tripdetail group by passenger_count"

result = pd.read_sql(query, conn)

display(result
```

39. Total revenue of entire dataset:

```
query = "select sum(fare_amount) as 'Total Revenue' from tripdetail"

result = pd.read_sql(query, conn)

display(result)
```

40. Trips where fare is between 50 and 150:

```
query = "select * from tripdetail where fare_amount between 50 and 150"

result = pd.read_sql(query, conn)

display(result)
```