

Q1:

```
select count (*) FROM `dtc-de-410519.green_taxi_2022.green_taxi_external`
```

Q2:

```
SELECT count (distinct PULocationID ) FROM `dtc-de-410519.green_taxi_2022.green_taxi_external`
```

Q3:

```
CREATE OR REPLACE TABLE `dtc-de-410519.green_taxi_2022.green_taxi_internal_nonpartitioned_tripdata`
AS SELECT * FROM `dtc-de-410519.green_taxi_2022.green_taxi_external`;
----
SELECT count (*) FROM `dtc-de-410519.green_taxi_2022.green_taxi_internal_nonpartitioned_tripdata` where
fare_amount =0;
```

Q4:

the most optimized strategy in BigQuery would be to **cluster on PULocationID and partition by lpep\_pickup\_datetime**.

**Cluster on PULocationID:** Clustering the data based on PULocationID ensures that rows with similar PULocationID values are stored together in the same storage blocks. When you order the results by PULocationID, BigQuery can efficiently retrieve the data because it's already clustered based on that column. This can significantly reduce the amount of data scanned and improve query performance.

**Partition by lpep\_pickup\_datetime:** Partitioning the data based on lpep\_pickup\_datetime allows for partition pruning when filtering data by this column. Partition pruning helps eliminate unnecessary partitions from being scanned, reducing query costs and improving performance.

```
CREATE TABLE `dtc-de-410519.green_taxi_2022_2022.green_taxi_internal_partitioned_tripdata`
PARTITION BY lpep_pickup_datetime
CLUSTER BY PULocationID AS (
  SELECT * FROM `dtc-de-410519.green_taxi_2022.green_taxi_internal_nonpartitioned_tripdata`
);
```

Q5:

1

2

3

4

5

-- lpep\_pickup\_datetime 06/01/2022 and 06/30/2022  
SELECT count(\*) FROM `dtc-de-410519.green\_taxi\_2022.green\_taxi\_internal\_partitioned\_tripdata`  
WHERE DATE(lpep\_pickup\_datetime) BETWEEN '2022-06-01' AND '2022-06-30';

Appuyez sur Alt+F1 pour afficher les options d'accessibilité

Résultats de la requête

ENREGISTRER LES RÉSULTATS

EXPLORER LES DONNÉES

INFORMATIONS SUR LE JOB

RÉSULTATS

GRAPHIQUE

BÉTA

JSON

DÉTAILS DE L'EXÉCUTION

Date et heure de création

11 févr. 2024, 02:05:26 UTC+1

Heure de début

11 févr. 2024, 02:05:26 UTC+1

Heure de fin

11 févr. 2024, 02:05:27 UTC+1

Durée

0 s

Octets traités

575,85 Ko

Octets facturés

10 Mo

Millisecondes d'emplacement

2202

Priorité de la tâche

INTERACTIVE

Utiliser l'ancien SQL

false

Table de destination


Table temporaire


Étiquettes


```
1
2
3 -- lpep_pickup_datetime 06/01/2022 and 06/30/2022
4 SELECT count(*) FROM `dtc-de-410519.green_taxi_2022.green_taxi_internal_nonpartitioned_tripdata`
5 WHERE DATE(lpep_pickup_datetime) BETWEEN '2022-06-01' AND '2022-06-30';
```

Appuyez sur Alt+F1 pour afficher les options d'accessibilité

Résultats de la requête

 ENREGISTRER LES RÉSULTATS

 EXPLORER LES DONNÉES



<

INFORMATIONS SUR LE JOB

RÉSULTATS

GRAPHIQUE

BÉTA

JSON

DÉTAILS DE L'EXÉCUTION

>

Date et heure de création	11 févr. 2024, 02:04:47 UTC+1
Heure de début	11 févr. 2024, 02:04:47 UTC+1
Heure de fin	11 févr. 2024, 02:04:47 UTC+1
Durée	0 s
Octets traités	6.41 Mo
Octets facturés	10 Mo
ms	201