Desafio CIVITAS

Henrique Borges de Almeida Filho

- **Queries de análise exploratória**

- Maior velocidade já registrada por um veículo. Também são retornados os dados como identificador, latitude e longitude do radar que capturou a foto, a placa e o tipo do veículo. Caso haja mais de um com a mesma velocidade, todos serão retornados.

```
SELECT
velocidade,
camera_numero,
camera_latitude,
camera_longitude,
placa,
tipoveiculo
FROM `rj-cetrio.desafio.readings_2024_06`
WHERE velocidade = (SELECT MAX(velocidade) FROM
`rj-cetrio.desafio.readings_2024_06`);
```

- Qual tipo de veículo é mais frequente. Utilizando a função COUNT(*), serão retornados os tipos de veículo e a quantidade de aparições deles nas fotos. Para descobrirmos o que mais aparece, ordenamos por tipo de veículo em ordem decrescente e limitamos o resultado a 1, recebendo assim o veículo com maior número de aparições.

```
SELECT
tipoveiculo,
COUNT(*) AS Quantidade
FROM `rj-cetrio.desafio.readings_2024_06`
GROUP BY tipoveiculo
ORDER BY tipoveiculo DESC
LIMIT 1;
```

- Quantas vezes cada tipo de veículo aparece. É bem similar à query anterior: basta removermos o ORDER BY e o LIMIT 1, retornando assim todos os tipos de veículo e quantas vezes cada um aparece.

```
SELECT
tipoveiculo,
```

```
COUNT(*) AS Quantidade
FROM `rj-cetrio.desafio.readings_2024_06`
GROUP BY tipoveiculo;
```

- Qual é a média de tempo entre a detecção do radar e o recebimento dos dados.
  Utilizando TIMESTAMP_DIFF(), calcula-se o tempo que cada foto levou para ser recebida. No meu exemplo, utilizei segundos, mas também pode-se receber esse valor em minutos ou horas. Através da função AVG(), calcula-se a média entre os valores.

```
SELECT
AVG(TIMESTAMP_DIFF(datahora_captura, datahora, SECOND)) AS Media
FROM `rj-cetrio.desafio.readings_2024_06`;
```

  Também é possível calcular essa média para um radar específico. Para isso, basta passar qual a latitude e longitude do radar no qual deseja-se receber o valor, através da cláusula WHERE.

```
SELECT
camera_latitude,
camera_longitude,
AVG(TIMESTAMP_DIFF(datahora_captura, datahora, SECOND)) AS Media
FROM `rj-cetrio.desafio.readings_2024_06`
WHERE camera_latitude = -22.854
AND camera_longitude = -43.2476
GROUP BY camera_latitude, camera_longitude;
```

- **Queries de possíveis placas clonadas**

Analisando os dados e colunas disponíveis, podemos descrever alguns cenários para julgarmos se ele é válido ou não para identificar placas clonadas. Por exemplo:

- Uma placa pode ser fotografada mais de uma vez
- Uma placa e tipoveiculo podem ser fotografada mais de uma vez
- Uma placa pode ser fotografada mais de uma vez em lugares diferentes
- Uma placa pode ser fotografada mais de uma vez em lugares diferentes em horários diferentes
- Uma placa e tipoveiculo NÃO podem aparecer com a mesma placa e um tipoveiculo diferente
- Uma placa e tipoveiculo NÃO podem ser fotografados por câmeras diferentes no mesmo horário

Utilizando os dois últimos cenários analisados podem indicar placas clonadas, obtemos então usarei-os como base.

- **Uma placa e tipoveiculo NÃO podem aparecer repetidamente com a mesma placa e um tipoveiculo diferente**

A query abaixo retorna as placas que se repetem e possuem uma placa e tipoveiculo diferentes. São retornados a placa, o tipo do veículo 1 (tomado como parâmetro) e o tipo do veículo 2 (a ser comparado). A query faz uma pesquisa paralela comparando os campos da pesquisa 'a' e pesquisa 'b'. Para isso ela faz 3 comparações:

-se a placa de 'a' é igual a de 'b';

-se o tipoveiculo 'a' é diferente do tipoveiculo 'b';

-se o tipoveiculo de 'a' é menor que o tipoveiculo 'b' (essa verificação serve para não retornar a mesma comparação mais de uma vez, por exemplo: placa1 = placa2 e placa2 = placa1. Dessa forma, apenas placa1 = placa2 será retornada).

Por fim, o "ORDER BY a.placa" ordena os resultados de acordo com as placas alfabeticamente.

```
SELECT DISTINCT
a.placa,
a.tipoveiculo AS tipoveiculo1,
b.tipoveiculo AS tipoveiculo2
FROM `rj-cetrio.desafio.readings_2024_06` a
JOIN `rj-cetrio.desafio.readings_2024_06` b
ON a.placa = b.placa
AND a.tipoveiculo <> b.tipoveiculo
AND a.tipoveiculo < b.tipoveiculo
ORDER BY a.placa;
```

- **Uma placa e tipoveiculo NÃO podem ser fotografados por câmeras diferentes no mesmo horário**

Assim como a query anterior, a query abaixo irá fazer uma pesquisa paralela entre os dados de 'a' e 'b'. Deve-se encontrar a inconsistência de que o mesmo veículo não poder ser fotografado por mais de uma câmera ao mesmo tempo, afinal, ele não pode estar em lugares diferentes ao mesmo tempo. Para isso, são feitas as seguintes comparações:

-se a placa de 'a' é igual à de 'b';

-se o tipoveiculo de 'a' é igual ao de 'b'

-se o datahora de 'a' é igual ao de 'b'

-se o camera_numero de 'a' é diferente do de 'b', o que caracteriza a inconsistência

Por fim, o "ORDER BY a.placa" ordena as placas alfabeticamente.

```
SELECT DISTINCT a.datahora, a.camera_numero, a.placa, a.tipoveiculo AS
tipoveiculo1, b.datahora, b.camera_numero, b.placa, b.tipoveiculo AS
tipoveiculo2
FROM `rj-cetrio.desafio.readings_2024_06` a
JOIN `rj-cetrio.desafio.readings_2024_06` b
```

```
ON a.placa = b.placa
AND a.tipoveiculo = b.tipoveiculo
AND a.tipoveiculo = b.tipoveiculo
AND a.datahora = b.datahora
AND a.camera_numero <> b.camera_numero
ORDER BY a.placa;
```

Obs.: Por motivos de segurança, o arquivo "./desafio-civitas-ba2f48ceea15.json" mencionado no código não foi commitado. Adicionei-o no .gitignore do meu projeto pois nele estão as chaves e credenciais necessárias para conectar o projeto ao Google Cloud Platform.