Documentation du nettoyage ou de la manipulation des données

- **Quels outils choisissez-vous et pourquoi ?** J'ai utilisé BigQuery pour la manipulation et le nettoyage de la base de données du 1er et 4e trimestre 2019 ainsi que le 1er trimestre 2020.
- **Avez-vous assuré l'intégrité de vos données ?** Cela sera fait en vérifiant les erreurs et en documentant le processus de nettoyage.

Je vous partage un fichier avec les codes SQL que j'ai utilisés pour nettoyer les fichiers.

- **Nettoyage :**

1- Suppression des doublons
2- Renommage des colonnes du 1er trimestre 2020 pour faciliter l'union des tables
3- Formater le trip_id (integer en string)
4- Unifier les fichiers trimestriels
5- Supprimer les colonnes non pertinentes à notre problématique
6- Changer les noms de deux observations dans la base
7- Identifier les lignes vides : il y en une ligne avec 2 cellules vides
8- Compléter les espaces vides avec les données de la base
9- Calculer la durée de trajet en secondes en mettant une condition, si end_time est inférieur à start_time, le résultat doit être null
10- Faire un filtre sur les durées qui sont nulles, il y en a 130 observations nulles, qui sont des erreurs techniques
11- Supprimer ces 130 observations nulles
12- Faire un filtre croissant, il y a des durées de 0 secondes, ce sont des valeurs aberrantes
13- On va faire un filtre et supprimer les données inférieures à 120 secondes, car je suppose un utilisateur prend un vélo pour un trajet environ 120 secondes/ 2 minutes pour 420 secondes/7 minutes à pied. Il y a 22934 observations
14- Filtre décroissant sur la durée, la durée maximum est de 10632022 secondes de trajet, de manière logique c'est impossible, en plus comme c'est un service vélo en libre-service les utilisateurs ne peuvent passer plus d'une journée avec les vélos
15- Supprimer les durées de trajet supérieures à 82800 secondes /23 heures. Il y a 908 trajets qui ont une durée supérieure à 82800 secondes
16- Calculer la variable ride_length sur le format H :M :S
17- Calculer le jour de la semaine où chaque trajet à commencer
18- Calculer les mois de chaque trajet
19- Calculer la saison de chaque trajet
20- Calcul statistique descriptive sur la durée de trajet des utilisateurs
21- Statistiques descriptives par type d'utilisateurs
22- Mode du jour de la semaine
23- Statistiques descriptives par jour de la semaine et par type d'utilisateurs
24- Statistiques descriptives par saison et type d'utilisateur

Avec une population de 1472038, les membres occasionnels représentent 12% et 88% les membres, il y a un fort écart entre les deux groupes. En catégorisant les durées de trajet, il semblerait que les clients occasionnels font en moyenne plus de temps avec les vélos que les membres soient 2194 secondes contre 696 secondes. La médiane nous montre que 50% des

clients occasionnels a fait une durée de 1339 secondes contre 50% des membres avec 518 secondes. Cette tendance montre que les clients occasionnels font plus de temps avec les vélos que les membres. Puisqu'il y a un écart important entre les deux groupes, le mieux c'est d'avancer les calculs statistiques.

Pour affiner les résultats et analyser cette base, on va continuer sur le logiciel **R**.

```
 --Déterminer s'il y a de doublon aux fichier 1er et 4e trimestre 2019 et celui de 2020: --1er trimestre 2019
SELECT
 trip_id,
 COUNT(*)
FROM
 `caduran2025.Projet_certif_google.Trajet_Q1`
GROUP BY
 trip_id
HAVING
 COUNT (*) > 1;
--Il n'ya pas de doublon dans la base du 1er trimestre 2019

--4e trimestre 2019
SELECT
 trip_id,
 COUNT(*)
FROM
 `caduran2025.Projet_certif_google.Trajet_Q4`
GROUP BY
 trip_id
HAVING
 COUNT (*) > 1;
--Il n'ya pas de doublon dans la base du 4e trimestre 2019
```

```
--1er trimestre 2020
SELECT
 ride_id,
 COUNT(*)
FROM
 `caduran2025.Projet_certif_google.Trajet_Q1_2020`
GROUP BY
 ride_id
HAVING
 COUNT (*) > 1;
--Il n'ya pas de doublon dans la base du 1er trimestre 2020
```

-- Les noms des colonnes pour le premier trimestre 2020 diffèrent de ceux de 2019. Pour simplifier les opérations, il est nécessaire d'uniformiser les noms en attributant ceux de 2019 à 2020, tout en créant une nouvelle TABLE dédiée au premier trimestre 2020.

```
CREATE TABLE
 `caduran2025.Projet_certif_google.Trajet_Q1_2020_v2` AS
SELECT
 ride_id AS trip_id,
 started_at AS start_time,
 ended_at AS end_time,
 start_station_id AS from_station_id,
 start_station_name AS from_station_name,
 end_station_id AS to_station_id,
 end_station_name AS to_station_name,
 member_casual AS usertype,
 rideable_type,
 start_lat,
 start_lng,
 end_lat,
 end_lng
FROM
```

```
  `caduran2025.Projet_certif_google.Trajet_Q1_2020`;

-- Créer une nouvelle table en ajoutant les données Q1 et Q4 2019)
CREATE TABLE
  `caduran2025.Projet_certif_google.Trajet_2019_Q1_Q4` AS
SELECT
  trip_id,
  start_time,
  end_time,
  from_station_id,
  from_station_name,
  to_station_id,
  to_station_name,
  usertype
FROM
  `caduran2025.Projet_certif_google.Trajet_Q1`

UNION ALL

SELECT
  trip_id,
  start_time,
  end_time,
  from_station_id,
  from_station_name,
  to_station_id,
  to_station_name,
  usertype
FROM
  `caduran2025.Projet_certif_google.Trajet_Q4`;
```

```
-- Changer les noms d'une colonne
--ALTER TABLE
-- `caduran2025.Projet_certif_google.Trajet_Q1_2020_v2`
--RENAME
--  COLUMN from_startion_name TO from_station_name;

--Convertir le format trip_id (integer) en string
SELECT
  CAST(trip_id AS string) AS trip_id,
  *
FROM
  `caduran2025.Projet_certif_google.Trajet_2019_Q1_Q4`

----Ajouter les données du 1er et 4e trimestres 2019 à celles du 1er trimestre de 2020
CREATE TABLE
  `caduran2025.Projet_certif_google.trajet_data` AS
SELECT
  trip_id,
  start_time,
  end_time,
  from_station_id,
  from_station_name,
  to_station_id,
  to_station_name,
  usertype
FROM
  `caduran2025.Projet_certif_google.Trajet_2019_Q1_Q4_v1`

UNION ALL
```

```
SELECT
 trip_id,
 start_time,
 end_time,
 from_station_id,
 from_station_name,
 to_station_id,
 to_station_name,
 usertype
FROM
 `caduran2025.Projet_certif_google.Trajet_Q1_2020_v1`;


-- Supprimer les colonnes vides
ALTER TABLE
 `caduran2025.Projet_certif_google.Trajet_Q1_Q4`
DROP
 COLUMN trip_id_1


--Changer les noms de deux observations pour avoir les mêmes noms d'observations
UPDATE
 `caduran2025.Projet_certif_google.trajet_data`
SET
 usertype =
 CASE
  WHEN usertype = 'Subscriber' THEN 'member'
  WHEN usertype = 'Customer' THEN 'casual'
  ELSE usertype
END
WHERE
```

```
 usertype IN ('Subscriber', 'Customer');


--Identifier les cellules vides de la base
SELECT
 *
FROM
 `caduran2025.Projet_certif_google.Trajet_Q4_v1`
WHERE
 trip_id IS NULL
 OR start_time IS NULL
 OR end_time IS NULL
 OR from_station_name IS NULL
 OR from_station_id IS NULL
 OR to_station_id IS NULL
 OR to_station_name IS NULL
 OR usertype IS NULL
--Il y a 2 cellules vides dont l'une dans la colonne 'to_station_id' et l'autre 'to_station_name'


-- Compléter les espaces vides par analogie
UPDATE
 `caduran2025.Projet_certif_google.trajet_data`
SET
 to_station_name = "HQ QR",
 to_station_id = 675
WHERE
 trip_id = '157EAA4C4A3C8D36'


--Calculer la durée du trajet en seconde
SELECT
 trip_id,
 start_time,
```

```
 end_time,
 from_station_id,
 from_station_name,
 to_station_id,
 to_station_name,
 usertype,
 CASE
  WHEN end_time < start_time THEN NULL
  ELSE TIMESTAMP_DIFF(end_time, start_time, SECOND)
END
 AS Trip_duration
FROM
 `caduran2025.Projet_certif_google.trajet_data`

-- Filtrer les résultats NULLS, il y en a 130
SELECT
 *
FROM
 `caduran2025.Projet_certif_google.trajet_data_v1`
WHERE
 Trip_duration IS NULL

--Il y a 130 cellules nulles

-- Supprimer ces 130 lignes nulles
DELETE
FROM
 `caduran2025.Projet_certif_google.trajet_data_v1`
WHERE
 Trip_duration IS NULL
```

```
-- Mettre en ordre croissant
SELECT
 *
FROM
 `caduran2025.Projet_certif_google.trajet_data_v1`
ORDER BY
 Trip_duration ASC;



-- Calculer le nombre de durée inférieure à 120 secondes
SELECT
 *
FROM
 `caduran2025.Projet_certif_google.trajet_data_v1`
WHERE
 Trip_duration < 120

--Il y a 22934 trajets qui sont inférieurs à 120 secondes

--Supprimer les trajets inférieurs à 120 secondes
DELETE
FROM
 `caduran2025.Projet_certif_google.trajet_data_v1`
WHERE
 Trip_duration < 120



--la durée maximum est de 10632022 secondes, il y a une erreur humaine, selon le projet les vélos sont en libre service, donc une personne ne peut rentrer avec le vélo chez elle, donc on va filtrer sur les durées supérieures à 82800 secondes
SELECT
 *
```

```
FROM
  `caduran2025.Projet_certif_google.trajet_data_v1`
ORDER BY
  Trip_duration DESC;


--Filtrer les durées supérieures à 23 h ou 82800s
SELECT
  *
FROM
  `caduran2025.Projet_certif_google.trajet_data_v1`
WHERE
  Trip_duration >=82800


-- Supprimer les 908 lignes qui ont une durée supérieure ou égale à 82800 secondes
DELETE
FROM
  `caduran2025.Projet_certif_google.trajet_data_v1`
WHERE
  Trip_duration >=82800


-- Calculer ride_length sur le format h:m:s
SELECT
  *,
  CASE
    WHEN end_time < start_time THEN NULL
    ELSE FORMAT('%02d:%02d:%02d', DIV(TIMESTAMP_DIFF(end_time, start_time, SECOND),
3600), MOD(DIV(TIMESTAMP_DIFF(end_time, start_time, SECOND), 60), 60),
MOD(TIMESTAMP_DIFF(end_time, start_time, SECOND), 60))
END
  AS ride_length
FROM
  `caduran2025.Projet_certif_google.trajet_data_v1`;
```

```
--Calculer jour de la semaine où chaque trajet a commencé
SELECT
 *,
 EXTRACT(DAYOFWEEK
 FROM
  start_time) AS day_of_week
FROM
 `caduran2025.Projet_certif_google.trajet_data_v2`;



--Calculer les mois de chaque début de trajet
ALTER TABLE
 `caduran2025.Projet_certif_google.trajet_data_v3`
ADD COLUMN IF NOT EXISTS month STRING;
UPDATE
 `caduran2025.Projet_certif_google.trajet_data_v3`
SET
 month = FORMAT_DATE("%B", start_time)
WHERE
 start_time IS NOT NULL;


-- Création de la colonne season
ALTER TABLE
 `caduran2025.Projet_certif_google.trajet_data_v3` ADD COLUMN season3 STRING;
UPDATE
 `caduran2025.Projet_certif_google.trajet_data_v3`
SET
 season3 =
 CASE
```

```
  WHEN EXTRACT(MONTH FROM SAFE_CAST(start_time AS DATETIME)) IN (12, 1, 2) THEN 'Hiver'
  WHEN EXTRACT(MONTH FROM SAFE_CAST(start_time AS DATETIME)) IN (3, 4, 5) THEN 'Printemps'
  WHEN EXTRACT(MONTH FROM SAFE_CAST(start_time AS DATETIME)) IN (6, 7, 8) THEN 'Été'
  WHEN EXTRACT(MONTH FROM SAFE_CAST(start_time AS DATETIME)) IN (9, 10, 11) THEN 'Automne'
  ELSE NULL
END
WHERE TRUE;


-- Calcul statistique descriptive sur la durée du trajet des utilisateurs
SELECT
 AVG(Trip_duration) AS average_trip_duration,
 STDDEV(Trip_duration) AS ecart_type_trip_duration,
 VAR_POP(Trip_duration) AS variance_trip_duration,
 count(*) as n
FROM
 `caduran2025.Projet_certif_google.trajet_data_v3`;
-- en moyenne les utilisateurs ont fait une durée de 872 secondes c'est-à-dire moins d'une heure de trajet, avec un total de 1472038 trajets distincts.



-- Statistiques descriptives par type d'utilisateur
SELECT
 usertype,
 AVG(Trip_duration) AS average_trip_duration,
 APPROX_QUANTILES(Trip_duration, 2)[OFFSET (1)] AS median_trip_duration,
 MAX(Trip_duration) AS max_trip_duration,
 MIN(Trip_duration) AS min_trip_duration,
 STDDEV(Trip_duration) AS stddev_trip_duration,
 VAR_POP(Trip_duration) AS variance_trip_duration,
```

```
 COUNT(*) AS n
FROM
 `caduran2025.Projet_certif_google.trajet_Q4_v4`
GROUP BY
 usertype;
```

--Avec une population de 1472038, les membres occasionnels représentent 12% et 88% les membres, il y a un fort ecart entre les deux groupes. En catégorisant les durées de trajet, il semblerait que les clients occasionnels font en moyenne plus de temps avec les vélos que les membres soit 2194 secondes contre 696 secondes. La médiane nous montre que 50% des clients occasionnels a fait une durée de 1339 secondes contre 50% des membres avec 518 secondes. Cette tendance montre que les clients occasionnels font plus de temps avec le vélos que les membres. mais puisque il y a un écart important entre les deux groupes, le mieux c'est d'avancer les calculs statistiques.

```
-- Mode de day_of_week
SELECT
 day_of_week,
 COUNT(*) AS count
FROM
 `caduran2025.Projet_certif_google.trajet_Q4_v4`
GROUP BY
 day_of_week
ORDER BY
 count DESC
```

```
-- Mode de day_of_week par usertype
SELECT
 day_of_week,usertype,
 COUNT(*) AS n
FROM
 `caduran2025.Projet_certif_google.trajet_data_v3`
GROUP BY
 day_of_week, usertype
```

```
ORDER BY
 n DESC

--Statistiques descriptives par jour de la semaine et type d'utilisateur
SELECT
 day_of_week,
 usertype,
 AVG(Trip_duration) AS trip_duration,
 COUNT(*) AS n
FROM
 `caduran2025.Projet_certif_google.trajet_data_v3`
GROUP BY
 day_of_week, usertype
ORDER BY
 day_of_week, usertype;

--Statistiques descriptives par saison et type d'utilisateur
SELECT
 season3,
 usertype,
 AVG(Trip_duration) AS moy_trip_duration,
 COUNT(*) AS n
FROM
 `caduran2025.Projet_certif_google.trajet_data_v3`
GROUP BY
 season3, usertype
ORDER BY
 season3, usertype;

--Pour affiner les résultats et analyser cette base, on va continuer sur le logiciel R.
```