Rapport sur le Travail Pratique : Ingestion et Traitement des Données avec Apache Hive et Apache Sqoop

Par Jean Marie Fande NDIAYE

Enseignant : Dr. Djibril Mboup

Cours : BigData – ING3 GEIT IPSL 2024

Table des matières

1. Introduction 2
1.1 Contexte du Projet 2
1.2 Objectif du TP 2
2. Prérequis Techniques 2
3. Partie I : Ingestion des Données avec Apache Sqoop 2
3.1 Téléchargement et Préparation de la Base de Données 2
3.2 Création de la Base de Données et des Utilisateurs 2
3.3 Ingestion des Données avec Sqoop 3
4. Partie II : Traitement des Données avec Apache Hive 4
4.1 Création des Tables Hive 4
4.2 Vérification de l'Importation des Données 5
5. Exercices et Requêtes SQL 5
5.1 Nombre Total de Commandes par Client en 2014 5
5.2 Clients sans Commandes, Triés par Nom 6
5.3 Top 5 Clients par Revenu Mensuel 7
5.4 Revenu Total Quotidien par Département 8
5.5 Rank de chaque catégorie par revenue obtenue dans chaque département 9
5.6 Pourcentage de chaque catégorie par revenue dans chaque département 10
5.7 clients qui ont passé une commande d'un montant supérieur à 200 $ 11
5.8 noms customer_fname commence par "Rich” 11
5.9 nombre total de clients dans chaque état (state) dont le prénom commence par « M » 12
5.10 Le produit le plus cher dans chaque catégorie 13
5.11 Les 10 meilleurs produits qui ont généré les revenus les plus élevés. 13
6. Conclusion 14

1. Introduction

1.1 Contexte du Projet

Dans l'ère du Big Data, la capacité à gérer, ingérer et analyser de vastes quantités de données est cruciale pour de nombreuses entreprises. Apache Hive et Apache Sqoop sont des outils essentiels dans l'écosystème Hadoop qui facilitent le traitement et la gestion des données. Apache Hive permet d'exécuter des requêtes SQL sur de grandes bases de données stockées dans Hadoop, tandis que Sqoop est utilisé pour transférer des données entre les bases de données relationnelles et Hadoop.

1.2 Objectif du TP

L'objectif de ce travail pratique est de démontrer les compétences nécessaires pour ingérer des données d'une base de données MySQL dans Apache Hive en utilisant Apache Sqoop, et de traiter ces données pour en extraire des informations utiles. Ce projet comprend deux parties principales : l'ingestion des données avec Sqoop et le traitement des données avec Hive.

2. Prérequis Techniques

- **Apache Sqoop**
- **Apache Hive**
- **MariaDB** ou **MySQL**
- **Vagrant** pour la gestion des machines virtuelles
- **Hadoop** pour le stockage et le traitement des données

3. Partie I : Ingestion des Données avec Apache Sqoop

3.1 Téléchargement et Préparation de la Base de Données

Un fichier SQL contenant la base de données de vente au détail (retail_db.sql) a été téléchargé depuis ce lien.

3.2 Création de la Base de Données et des Utilisateurs

Les étapes suivantes ont été suivies pour créer un utilisateur, une base de données et charger les données dans MariaDB/MySQL.

1. **Connexion à MySQL en tant que root**

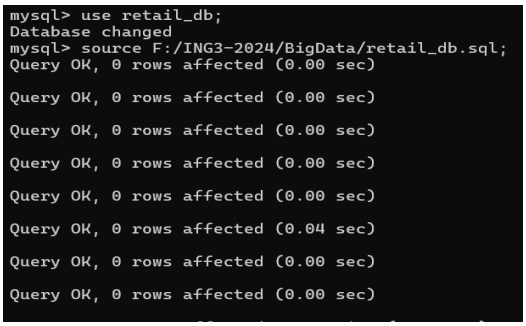
   ```
   mysql -u root -p
   ```

2. **Création d'un utilisateur et d'une base de données**

   ```
   CREATE USER 'retail_user'@'localhost' IDENTIFIED BY 'hadoop';
   CREATE DATABASE retail_db;
   GRANT ALL PRIVILEGES ON retail_db.* TO 'retail_user'@'localhost';
   ```

```
FLUSH PRIVILEGES;
```

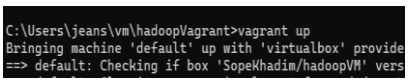
3. **Chargement des données dans la base de donnée**

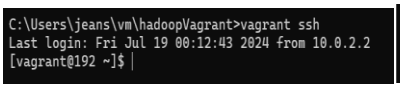


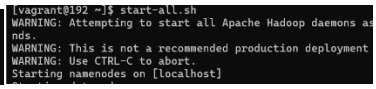
3.3 Ingestion des Données avec Sqoop

Apache Sqoop a été utilisé pour transférer les données depuis la base de données MySQL de la machine locale à Apache Hive de notre cluster hadoop.

1. **Démarrage du cluster cluster et verification de l'appartenance au même réseau**

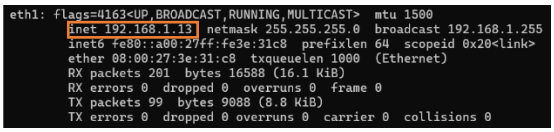






Configuration réseaux Machine virtuelle

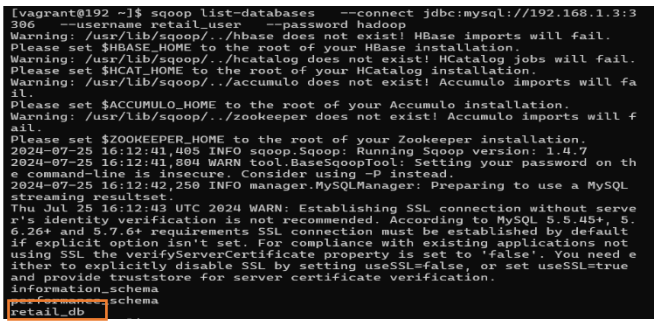
Configuration réseau Machine locale



Nous remarquons bien que nos deux machine partage le même réseaux et donc peuvent communiqué facilemement.

NB : il a fallut desactiver le pare feu de la machine local pour permettre l'envoi de paquet de la VM

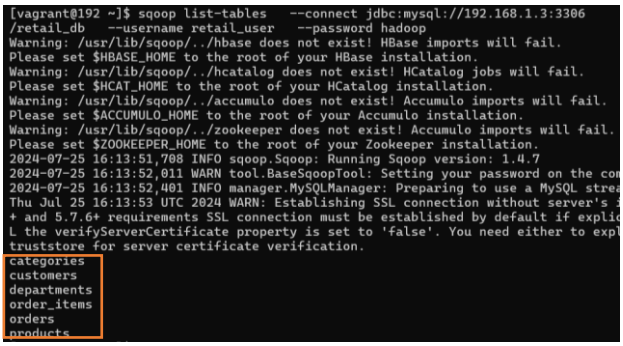
2. **Lister les bases de données disponibles**



La base de données *retail_db* a bien été détecté par la VM à travers la commande :

sqoop list-databases –connect "jdbc:mysql://192.168.1.3:3306" --username retail_user --password hadoop

3. **Lister les tables disponibles dans la base de données**

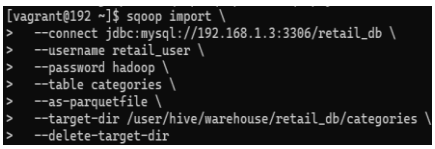


La base de données *retail_db* a bien été détecté par la VM

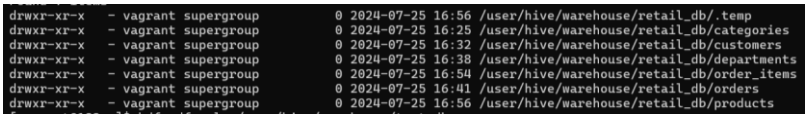
4. **Importer les tables dans Hive**

Pour importer les tables dans Hive la commande suivante a été utilisée :

```
sqoop import --connect jdbc:mysql://192.168.1.3:3306/retail_db --username
retail_user --password hadoop --table tableName --as-parquetfile --target-
dir=/user/hive/warehouse/retail_db.db/tableName --delete-target-dir
```



5. **Vérification de l'ingestion correcte des données**

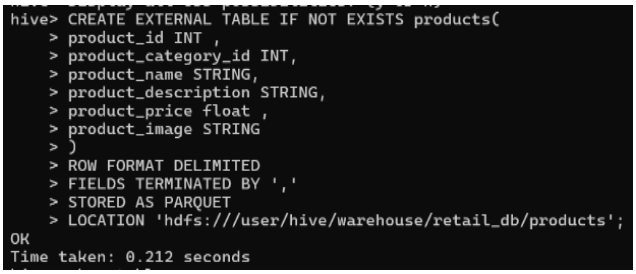


Toutes les tables ont été bien ingérés.

4. Partie II : Traitement des Données avec Apache Hive

4.1 Création des Tables Hive

Les tables externes dans Hive ont été créées pour correspondre aux tables importées de MySQL.



Les tables externes dans Hive ont été créées pour correspondre aux tables importées de MySQL en précisant l'emplacement des données importées avec l'instruction *LOCATION*

4.2 Vérification de l'Importation des Données

Les commandes suivantes ont été exécutées pour vérifier que les données ont été correctement importées dans Hive.

```
hive
SHOW TABLES;

SELECT * FROM products LIMIT 10;
```

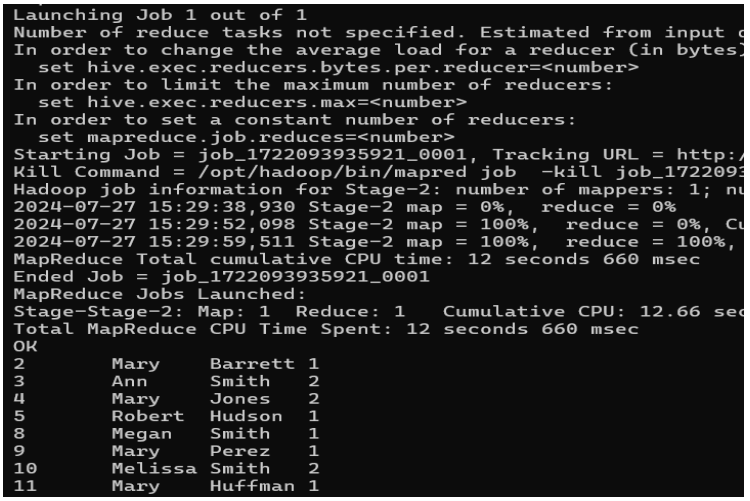
5. Exercices et Requêtes SQL

Des requêtes SQL ont été exécutées pour analyser les données. Voici quelques exemples de requêtes et leurs résultats.

5.1 Nombre Total de Commandes par Client en 2014

```
SELECT C.customer_id,C.customer_fname,C.customer_lname, COUNT(*) AS orders_nb
FROM orders O, customers C
WHERE O.order_customer_id=C.customer_id AND O.order_status='COMPLETE' AND
YEAR(FROM_UNIXTIME(CAST(O.order_date AS BIGINT) DIV 1000))=2014
GROUP BY C.customer_id, C.customer_fname, C.customer_lname,C.customer_email
LIMIT 10;;
```

Résultat :



5.2 Clients sans Commandes, Triés par Nom

```
SELECT C.customer_fname, C.customer_lname
FROM customers C
LEFT JOIN orders O ON C.customer_id = O.order_customer_id
WHERE O.order_id IS NULL
ORDER BY C.customer_lname, C.customer_fname;
Resultat:
Hadoop job information for Stage-2: number of mappers: 1; number of reducers:
1
2024-07-27 15:37:19,040 Stage-2 map = 0%,  reduce = 0%
2024-07-27 15:37:29,678 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 6.57
sec
2024-07-27 15:37:37,096 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU
9.32 sec
MapReduce Total cumulative CPU time: 9 seconds 320 msec
Ended Job = job_1722093935921_0002
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 9.32 sec   HDFS Read:
142937 HDFS Write: 334 SUCCESS
Total MapReduce CPU Time Spent: 9 seconds 320 msec
OK
Mary    Bolton
Albert  Ellison
Carolyn Green
Mary    Greene
Mary    Harrell
Mary    Lewis
Mary    Mueller
Matthew Patel
Mary    Shaw
Amanda  Smith
```

5.3 Top 5 Clients par Revenu Mensuel

```sql
WITH monthly_revenue AS (
    SELECT
        C.customer_id,
        C.customer_fname,
        C.customer_lname,
        C.customer_email,
        C.customer_street,
        C.customer_city,
        C.customer_state,
        C.customer_zipcode,
        FROM_UNIXTIME(CAST(O.order_date AS BIGINT) DIV 1000, 'yyyy-MM') AS order_month,
        SUM(Oi.order_item_product_price * Oi.order_item_quantity) AS monthly_revenue
    FROM
        customers C
    JOIN
        orders O ON C.customer_id = O.order_customer_id
    JOIN
        order_items Oi ON O.order_id = Oi.order_item_order_id
    WHERE
        O.order_status IN ('COMPLETE', 'CLOSED')
    GROUP BY
        C.customer_id,
        C.customer_fname,
        C.customer_lname,
        C.customer_email,
        C.customer_street,
        C.customer_city,
        C.customer_state,
        C.customer_zipcode,
        FROM_UNIXTIME(CAST(O.order_date AS BIGINT) DIV 1000, 'yyyy-MM')
),
ranked_customers AS (
    SELECT
        *,
        ROW_NUMBER() OVER (PARTITION BY order_month ORDER BY monthly_revenue DESC) AS customer_rank
    FROM
        monthly_revenue
)
SELECT customer_id, customer_fname, customer_lname,customer_email, customer_street,customer_city,customer_state,
    customer_zipcode,order_month,monthly_revenue,customer_rank
FROM   ranked_customers
WHERE
    customer_rank <= 5
ORDER BY
    order_month,
    monthly_revenue DESC;
```

Résultats(limité a 16)

customer_id	customer_fname	customer_lname	customer_email	customer_street	customer_city
1478	Anna	Sanchez	XXXXXXXXX	3186 Heather Rise Terrace	Rome
11941	Jeffrey	Pugh	XXXXXXXXX	3233 Sleepy View Link	Cayey
1498	Peter	Bray	XXXXXXXXX	4490 Amber Rise	Houston
5570	Mary	Smith	XXXXXXXXX	852 Hazy Gate Circuit	Mission Viejo
2419	Rachel	Horn	XXXXXXXXX	3838 Bright Creek Vista	Caguas
9515	Victoria	Smith	XXXXXXXXX	870 High View Close	Caguas
9494	Robert	Gilmore	XXXXXXXXX	8248 Clear Glade	Del Rio
6088	Mary	Brooks	XXXXXXXXX	8711 Middle Mountain Maze	Caguas
7186	Mary	Meadows	XXXXXXXXX	9257 Round Field	Caguas
3295	Maria	Joseph	XXXXXXXXX	8808 Gentle Crossing	Los Angeles
1148	Mary	Anderson	XXXXXXXXX	7213 Cozy Quay	Augusta
12066	Kevin	Smith	XXXXXXXXX	6234 Honey Grove Expressway	Union City
8090	Mary	Martinez	XXXXXXXXX	6077 Heather Zephyr Estates	Clarksville
5147	Mary	Hurst	XXXXXXXXX	9276 Jagged Towers	Caguas
7607	Scott	Smith	XXXXXXXXX	9471 Middle Grove Place	Caguas
11284	Sarah	Sherman	XXXXXXXXX	7668 Fallen Orchard	Manati

customer_city	customer_state	customer_zipcode	order_month	monthly_revenue	customer_rank
Rome	NY	13440	2013-07	1784.7600002288818	1
Cayey	PR	00736	2013-07	1649.8000183105469	2
Houston	TX	77084	2013-07	1619.8800468444824	3
Mission Viejo	CA	92691	2013-07	1549.8699989318848	4
Caguas	PR	00725	2013-07	1499.900032043457	5
Caguas	PR	00725	2013-08	3449.909999847412	1
Del Rio	TX	78840	2013-08	2869.730037689209	2
Caguas	PR	00725	2013-08	2639.7700424194336	3
Caguas	PR	00725	2013-08	2465.8100452423096	4
Los Angeles	CA	90024	2013-08	2279.900047302246	5
Augusta	GA	30907	2013-09	2859.889995574951	1
Union City	NJ	07087	2013-09	2659.840072631836	2
Clarksville	TN	37042	2013-09	2519.740074157715	3
Caguas	PR	00725	2013-09	2259.740062713623	4
Caguas	PR	00725	2013-09	2224.800043106079	5
Manati	PR	00674	2013-10	2854.7300567626953	1

5.4 Revenu Total Quotidien par Département

```
SELECT  TO_DATE(FROM_UNIXTIME(CAST(order_date AS BIGINT) DIV 1000)), D.department_name,
SUM(Oi.order_item_product_price*Oi.order_item_quantity) AS order_revenue
FROM orders O, order_items Oi, products P, categories C, departments D
WHERE O.order_id=Oi.order_item_order_id AND Oi.order_item_product_id=P.product_id AND
P.product_category_id=C.category_id AND C.category_department_id=D.department_id
AND (O.order_status='COMPLETE' OR O.order_status='CLOSED')
GROUP BY TO_DATE(FROM_UNIXTIME(CAST(order_date AS BIGINT) DIV 1000)), D.department_name;
Resultats :
MapReduce Jobs Launched:
Stage-Stage-13: Map: 1   Cumulative CPU: 4.22 sec   HDFS Read: 28172 HDFS Write: 27455 SUCCESS
Stage-Stage-14: Map: 1   Cumulative CPU: 6.05 sec   HDFS Read: 846656 HDFS Write: 3235814 SUCCESS
Stage-Stage-11: Map: 1   Cumulative CPU: 4.44 sec   HDFS Read: 3242071 HDFS Write: 3092424 SUCCESS
Stage-Stage-4: Map: 1  Reduce: 1   Cumulative CPU: 10.34 sec   HDFS Read: 3113472 HDFS Write: 581
SUCCESS
Total MapReduce CPU Time Spent: 25 seconds 50 msec
OK
2013-07-25      Apparel 5309.290176391602
2013-07-25      Fan Shop        15898.150375366211
2013-07-25      Fitness 494.9299907684326
2013-07-25      Footwear        5699.479881286621
```

```
2013-07-25      Golf    3049.690055847168
2013-07-25      Outdoors        1095.6899967193604
2013-07-26      Apparel 11228.420360565186
2013-07-26      Fan Shop        26646.87062072754
2013-07-26      Fitness 344.9700050354004
2013-07-26      Footwear        7R59.20986366272
```

5.5 Rank de chaque catégorie par revenue obtenue dans chaque département

```sql
WITH category_revenue AS (
    SELECT
        D.department_name,
        C.category_name,
        SUM(Oi.order_item_product_price * Oi.order_item_quantity) AS category_revenue
    FROM
        orders O
    JOIN order_items Oi ON O.order_id = Oi.order_item_order_id
    JOIN products P ON Oi.order_item_product_id = P.product_id
    JOIN categories C ON P.product_category_id = C.category_id
    JOIN departments D ON C.category_department_id = D.department_id
    WHERE
        O.order_status IN ('COMPLETE', 'CLOSED')
    GROUP BY
        D.department_name,
        C.category_name
)


SELECT department_name, category_name, category_revenue,
    RANK() OVER (PARTITION BY department_name ORDER BY category_revenue DESC) AS
category_rank
FROM
    category_revenue
ORDER BY
    department_name ASC,
    category_rank ASC;
```

resultats :

department_name	category_name	category_revenue	category_rank
Apparel	Cleats	1934378.2438316345	1
Apparel	Men's Footwear	1278971.6640472412	2
Fan Shop	Fishing	3022248.9630126953	1
Fan Shop	Camping & Hiking	1802879.866027832	2
Fan Shop	Water Sports	1342783.0065917969	3
Fan Shop	Indoor/Outdoor Games	1257646.7284812927	4
Fan Shop	Hunting & Shooting	24352.049995422363	5
Fitness	Baseball & Softball	40757.16142272949	1
Fitness	Hockey	19557.929931640625	2
Fitness	Tennis & Racquet	18490.890689849854	3
Fitness	Lacrosse	16045.619770050049	4
Fitness	Soccer	12438.579696655273	5
Fitness	Basketball	9299.759765625	6
Footwear	Cardio Equipment	1639187.2152328491	1
Footwear	Electronics	49014.70033836365	2

5.6 Pourcentage de chaque catégorie par revenue dans chaque département

```sql
WITH category_revenue AS (
    SELECT
        D.department_name,
        C.category_name,
        SUM(Oi.order_item_product_price * Oi.order_item_quantity) AS category_revenue
    FROM
        orders O
    JOIN order_items Oi ON O.order_id = Oi.order_item_order_id
    JOIN products P ON Oi.order_item_product_id = P.product_id
    JOIN categories C ON P.product_category_id = C.category_id
    JOIN departments D ON C.category_department_id = D.department_id
    WHERE
        O.order_status IN ('COMPLETE', 'CLOSED')
    GROUP BY
        D.department_name,
        C.category_name),
department_revenue AS (
    SELECT
        department_name,
        SUM(category_revenue) AS total_revenue
    FROM
        category_revenue
    GROUP BY
        department_name
)
SELECT cr.department_name, cr.category_name, cr.category_revenue,
    (cr.category_revenue / dr.total_revenue) * 100 AS category_percentage
FROM category_revenue cr
JOIN  department_revenue dr ON cr.department_name = dr.department_name
ORDER BY cr.department_name ASC, category_percentage DESC;

résultats(limité à 3 dept)
| department_name | category_name        | category_revenue   | category_percentage |
+-----------------+----------------------+--------------------+---------------------+
| Apparel         | Cleats               | 1934378.2438316345 |   60.19818255984804 |
| Apparel         | Men's Footwear       | 1278971.6640472412 |   39.80181744015196 |
| Fan Shop        | Fishing              | 3022248.9630126953 |  40.567586908881815 |
| Fan Shop        | Camping & Hiking     |  1802879.866027832 |   24.20002009975048 |
| Fan Shop        | Water Sports         | 1342783.0065917969 |  18.024149230042607 |
| Fan Shop        | Indoor/Outdoor Games | 1257646.7284812927 |  16.881366685118262 |
| Fan Shop        | Hunting & Shooting   | 24352.049995422363 |  0.3268770762068359 |
| Fitness         | Baseball & Softball  |  40757.16142272949 |   34.95769958924147 |
| Fitness         | Hockey               | 19557.929931640625 |    16.7749719379731 |
| Fitness         | Tennis & Racquet     | 18490.890689849854 |   15.85976499120934 |
| Fitness         | Lacrosse             | 16045.619770050049 |  13.762439190178494 |
| Fitness         | Soccer               | 12438.579696655273 |  10.668655941039608 |
| Fitness         | Basketball           |     9299.759765625 |    7.97646835035799 |
```

5.7 clients qui ont passé une commande d'un montant supérieur à 200 $

```
WITH order_totals AS (
    SELECT
        O.order_customer_id,
        SUM(Oi.order_item_product_price * Oi.order_item_quantity) AS total_amount
    FROM
        orders O
    JOIN
        order_items Oi ON O.order_id = Oi.order_item_order_id
    WHERE
        O.order_status IN ('COMPLETE', 'CLOSED')
    GROUP BY
        O.order_id, O.order_customer_id
    HAVING
        SUM(Oi.order_item_product_price * Oi.order_item_quantity) > 200
) /*suite page suivantes */
SELECT DISTINCT C.customer_id, C.customer_fname, C.customer_lname, C.customer_email
FROM order_totals OT
JOIN customers C ON OT.order_customer_id = C.customer_id
ORDER BY
    C.customer_lname, C.customer_fname;

Resultats(limité à 10):
+-------------+----------------+----------------+----------------+
| customer_id | customer_fname | customer_lname | customer_email |
+-------------+----------------+----------------+----------------+
|       10925 | Marie          | Abbott         | XXXXXXXXX      |
|        9880 | Donna          | Acevedo        | XXXXXXXXX      |
|        2528 | Mary           | Acevedo        | XXXXXXXXX      |
|        7802 | Mary           | Acevedo        | XXXXXXXXX      |
|       11735 | Mary           | Acevedo        | XXXXXXXXX      |
|        5496 | Kevin          | Acosta         | XXXXXXXXX      |
|        1800 | Lori           | Acosta         | XXXXXXXXX      |
|       10787 | Mary           | Acosta         | XXXXXXXXX      |
|       11654 | Michelle       | Acosta         | XXXXXXXXX      |
|        6030 | Alexander      | Adams          | XXXXXXXXX      |
```

5.8 noms customer_fname commence par "Rich”

```
SELECT customer_id, customer_fname, customer_lname, customer_email
FROM customers
WHERE
    customer_fname LIKE 'Rich%'
ORDER BY
    customer_lname, customer_fname;
```

```
résultats(limité à 10):
+-------------+----------------+----------------+----------------+
| customer_id | customer_fname | customer_lname | customer_email |
+-------------+----------------+----------------+----------------+
|        8853 | Richard        | Ali            | XXXXXXXXX      |
|       11576 | Richard        | Andrade        | XXXXXXXXX      |
|        7385 | Richard        | Arellano       | XXXXXXXXX      |
|       12100 | Richard        | Bolton         | XXXXXXXXX      |
|        5556 | Richard        | Burns          | XXXXXXXXX      |
|        3301 | Richard        | Davila         | XXXXXXXXX      |
|       10703 | Richard        | Dickson        | XXXXXXXXX      |
|        6779 | Richard        | Durham         | XXXXXXXXX      |
|        2221 | Richard        | Edwards        | XXXXXXXXX      |
|       12403 | Richard        | Ferguson       | XXXXXXXXX      |
```

5.9 nombre total de clients dans chaque état (state) dont le prénom commence par « M »

```sql
SELECT
    customer_state,
    COUNT(*) AS total_customers
FROM
    customers
WHERE
    customer_fname LIKE 'M%'
GROUP BY
    customer_state
ORDER BY
    total_customers DESC;
```

```
resultats(limité à 10)
+----------------+-----------------+
| customer_state | total_customers |
+----------------+-----------------+
| PR             |            2063 |
| CA             |             850 |
| NY             |             331 |
| TX             |             267 |
| IL             |             222 |
| FL             |             162 |
| OH             |             130 |
| PA             |             120 |
| MI             |             114 |
| AZ             |              98 |
```

5.10 Le produit le plus cher dans chaque catégorie

```
WITH max_price_per_category AS (
    SELECT
        product_category_id,
        MAX(product_price) AS max_price
    FROM  products
    GROUP BY  product_category_id
)
SELECT DISTINCT P.product_id, P.product_name, P.product_price, P.product_category_id,
    C.category_name
FROM products P
JOIN  max_price_per_category MPC ON P.product_category_id = MPC.product_category_id AND
P.product_price = MPC.max_price
JOIN categories C ON P.product_category_id = C.category_id
ORDER BY
    P.product_category_id;
```

Resultats (limit 10)

product_name	product_price	product_category_id	category_name
Riddell Youth 360 Custom Football Helmet	299.99	2	Soccer
Quik Shade Summit SX170 10 FT. x 10 FT. Canop	199.99	3	Baseball & Softball
SOLE F85 Treadmill	1799.99	4	Basketball
Goaliath 54" In-Ground Basketball Hoop with P	499.99	5	Lacrosse
YETI Tundra 65 Chest Cooler	399.99	6	Tennis & Racquet
Stiga Master Series ST3100 Competition Indoor	329.99	7	Hockey
YETI Tundra 65 Chest Cooler	399.99	8	More Sports
Goaliath 54" In-Ground Basketball Hoop with P	499.99	9	Cardio Equipment
SOLE E35 Elliptical	1999.99	10	Strength Training
Marcy Diamond 9010 Smith Cage	799.99	11	Fitness Accessories

5.11 Les 10 meilleurs produits qui ont généré les revenus les plus élevés.

```
SELECT P.product_id, P.product_name,
    SUM(Oi.order_item_product_price * Oi.order_item_quantity) AS total_revenue
FROM order_items Oi
JOINproducts P ON Oi.order_item_product_id = P.product_id

GROUP BY P.product_id, P.product_name
ORDER BY total_revenue DESC
LIMIT 10;
```

customer_id	customer_fname	customer_lname	customer_email	customer_street	customer_city
9586	Mary	Olson	XXXXXXXXX	1399 Dewy Expressway	Caguas
9515	Victoria	Smith	XXXXXXXXX	870 High View Close	Caguas
9494	Robert	Gilmore	XXXXXXXXX	8248 Clear Glade	Del Rio
1148	Mary	Anderson	XXXXXXXXX	7213 Cozy Quay	Augusta
11284	Sarah	Sherman	XXXXXXXXX	7668 Fallen Orchard	Manati
10351	Teresa	Gray	XXXXXXXXX	8021 Misty Plaza	Caguas
2564	Phillip	Smith	XXXXXXXXX	6068 Quaking Heath	Amarillo
9337	Mary	Smith	XXXXXXXXX	5687 Lazy Parade	Chicago
8777	Mary	Campos	XXXXXXXXX	8842 Crystal Horse Green	Kent
2555	Mary	Long	XXXXXXXXX	1022 Dusty Glen	Costa Mesa

customer_city	customer_state	customer_zipcode	order_month	monthly_revenue	customer_rank
Caguas	PR	00725	2607-12	4029.6100883483887	1
Caguas	PR	00725	2607-12	3449.909999847412	2
Del Rio	TX	78840	2607-12	2869.730037689209	3
Augusta	GA	30907	2607-12	2859.889995574951	4
Manati	PR	00674	2607-12	2854.7300567626953	5
Caguas	PR	00725	2608-03	4489.650043487549	1
Amarillo	TX	79109	2608-03	4069.840030670166	2
Chicago	IL	60643	2608-03	3603.6500968933105	3
Kent	WA	98031	2608-03	3029.720039367676	4
Costa Mesa	CA	92626	2608-03	2954.6300678253174	5

6. Conclusion

Ce TP a permis de démontrer la capacité à ingérer des données d'une base de données relationnelle dans Hive en utilisant Sqoop et à exécuter des requêtes SQL pour analyser ces données. Les principaux résultats montrent des tendances intéressantes sur les comportements d'achat des clients et les performances des produits.

- La configuration correcte de l'environnement de travail est essentielle pour le bon déroulement du projet.
- La maîtrise des commandes Sqoop et Hive est cruciale pour l'ingestion et le traitement efficaces des données.
- L'analyse des données nécessite une compréhension approfondie des structures de données et des requêtes SQL.