**Data cleaning process**

**Non-uniform values from film table**

--looking for non-uniform values from film table

SELECT film\_id, title, description, release\_year, language\_id, rental\_duration, rental\_rate, replacement\_cost, rating, last\_update

FROM film

**There is no non-uniform data in film table. If I found that the rating PG was entered as P'G, P,G, or P.G. I will correct it with the following query;**

--Update P'G, P,G, and P.G as PG in the film table

UPDATE film SET rating ='G' WHERE rating IN ('P'G', 'P.G ', 'P.G ')

**Duplicate data from film table**

--looking for duplicate data from film table

SELECT film\_id, title, description, release\_year, language\_id, rental\_duration, rental\_rate, replacement\_cost, rating, last\_update,

COUNT(\*) FROM film

GROUP BY film\_id, title, description, release\_year, language\_id, rental\_duration, rental\_rate, replacement\_cost, rating, last\_update

HAVING COUNT(\*) >1;

**No duplicate was found on the film table. To fix the duplicate records in film table, we can either Create a virtual table, known as a “view,” where you select only unique records or delete the duplicate record from the table or view. Deleting records is not a good option, the best option is to create the unique record. The following query will create a unique record;**

--Shows only unique records from the film table

SELECT DISTINCT film\_id, title, description, release\_year, language\_id,

rental\_duration, rental\_rate, replacement\_cost, rating, last\_update

FROM film

**Missing values from film table**

--looking for missing values from film table

SELECT film\_id, title, description, release\_year, language\_id, rental\_duration, rental\_rate, replacement\_cost, rating, last\_update

FROM film

**There are no missing values in the film table. If there are high percentage of missing values, its best to ignore the column in a query. If there are few missing values in a column, then, we can impute values such as estimates to fill in the missing values. The following query can be used;**

**--imputing missing values in language\_id with the MODE value in film table**

UPDATE film SET = MODE(language\_id ) WHERE language\_id IS NULL

**Non-uniform values from customer table**

**--looking for non-uniform values data from customer table**

SELECT customer\_id, store\_id, first\_name, last\_name, email, address\_id, activebool, create\_date, last\_update

FROM customer

**There is no non-uniform data in customer table. If I found that the customer\_id 2 was entered as .2, /2, and ;2. I will correct it with the following query;**

--Update 2, /2, and ;2 as 2 in the customer table

UPDATE customer SET rating ='2' WHERE customer\_id IN ('.2', '/2 ', ';2 ')

**Duplicate data from customer table**

--looking for duplicate data from customer table

SELECT customer\_id, store\_id, first\_name, last\_name, email, address\_id, activebool, create\_date, last\_update,

COUNT(\*) FROM customer

GROUP BY customer\_id, store\_id, first\_name, last\_name, email, address\_id, activebool, create\_date, last\_update HAVING COUNT(\*) >1;

**No duplicate was found on the customer table. To fix the duplicate records in customer table, we can either Create a virtual table, known as a “view,” where you select only unique records or delete the duplicate record from the table or view. Deleting records is not a good option, the best option is to create the unique record. The following query will create a unique record;**

**CREATE VIEW customer\_table2**

AS SELECT customer\_id, store\_id, first\_name, last\_name, email, address\_id, activebool, create\_date, last\_update FROM customer

GROUP BY customer\_id, store\_id, first\_name, last\_name, email, address\_id, activebool, create\_date, last\_update

**--Group by will make each row unique**

**Missing values from customer table**

--looking for missing values from customer table

SELECT customer\_id, store\_id, first\_name, last\_name, email, address\_id, activebool, create\_date, last\_update

FROM customer

**There are no missing values in the customer table. If there are high percentage of missing values, its best to ignore the column in a query. If there are few missing values in a column, then, we can impute values such as estimates to fill in the missing values. The following query can be used;**

--imputing missing values in language\_id with the MODE value in customer table

UPDATE customer SET = MODE(activebool) WHERE activebool IS NULL

**Descriptive Analysis**

**2a. Film table**

--select and display minimum, maximum and average of film\_id, release\_year, language\_id, rental\_duration, rental\_rate, replacement\_cost, count of rows, and modes of title, description, rating, last\_update, special\_features and fulltext from the film table

SELECT MIN(film\_id) AS min\_film\_id,

MAX(film\_id) AS max\_film\_id,

AVG(film\_id) AS avg\_film\_id,

COUNT(film\_id) AS count\_rent\_values,

MIN(release\_year) AS min\_release\_year,

MAX(release\_year) AS max\_release\_year,

AVG(release\_year) AS avg\_release\_year,

MIN(language\_id) AS min\_language\_id,

MAX(language\_id) AS max\_language\_id,

AVG(language\_id) AS avg\_language\_id,

MIN(rental\_duration) AS min\_rental\_duration,

MAX(rental\_duration) AS max\_rental\_duration,

AVG(rental\_duration) AS avg\_rental\_duration,

MIN(rental\_rate) AS min\_rental\_rate,

MAX(rental\_rate) AS max\_rental\_rate,

AVG(rental\_rate) AS avg\_rental\_rate,

MIN(replacement\_cost) AS MIN\_replacement\_cost,

MAX(replacement\_cost) AS max\_replacement\_cost,

AVG(replacement\_cost) AS avg\_replacement\_cost,

COUNT(\*) AS count\_rows,

MODE () WITHIN GROUP (ORDER BY title) AS mode\_title,

MODE () WITHIN GROUP (ORDER BY description) AS mode\_description,

MODE () WITHIN GROUP (ORDER BY rating) AS mode\_rating,

MODE () WITHIN GROUP (ORDER BY last\_update) AS mode\_last\_update,

MODE () WITHIN GROUP (ORDER BY special\_features) AS mode\_special\_features,

MODE () WITHIN GROUP (ORDER BY fulltext) AS mode\_fulltext FROM film;

**2b. Customer table**

--select and display minimum, maximum and average of customer\_id, store\_id, address\_id, active, count of rows, and modes of first\_name, last\_name, email, activebool and create\_date from the customer table

SELECT MIN(customer\_id) AS min\_customer\_id,

MAX(customer\_id) AS max\_customer\_id,

AVG(customer\_id) AS avg\_customer\_id,

MIN(store\_id) AS min\_store\_id,

MAX(store\_id) AS max\_store\_id,

AVG(store\_id) AS avg\_store\_id,

MIN(address\_id) AS min\_address\_id,

MAX(address\_id) AS max\_address\_id,

AVG(address\_id) AS avg\_address\_id,

MIN(active) AS min\_active,

MAX(active) AS max\_active,

AVG(active) AS avg\_active,

COUNT(\*) AS count\_rows,

MODE () WITHIN GROUP (ORDER BY first\_name) AS mode\_first\_name,

MODE () WITHIN GROUP (ORDER BY last\_name) AS mode\_last\_name,

MODE () WITHIN GROUP (ORDER BY email) AS mode\_email,

MODE () WITHIN GROUP (ORDER BY activebool) AS mode\_activebool,

MODE () WITHIN GROUP (ORDER BY create\_date) AS mode\_create\_date

FROM customer;

**INNER JOIN**

*--select and display customer\_id and count of country, count of country top 10, arranged in descending order.*

*SELECT D.country, COUNT(A.customer\_id) AS customer\_numbers*

*FROM customer A*

*INNER JOIN address B ON A.address\_id = B.address\_id*

*INNER JOIN city C ON B.city\_id = C.city\_id*

*INNER JOIN country D ON C.country\_ID = D.country\_ID*

*GROUP BY D.country*

*ORDER BY COUNT (A.customer\_id) DESC LIMIT 10;*

*--select and display customer\_id, count of country, top 10 cities from count of country top 10, arranged in descending order.*

*SELECT D.country, C.city,*

*COUNT(A.customer\_id) AS customer\_numbers*

*FROM customer A*

*INNER JOIN address B ON A.address\_id = B.address\_id*

*INNER JOIN city C ON B.city\_id = C.city\_id*

*INNER JOIN country D ON C.country\_ID = D.country\_ID*

*WHERE D.country IN('India', 'China', 'United States', 'Japan', 'Mexico',*

*'Brazil', 'Russian Federation', 'Phillipines', 'Turkey', 'Indonesia')*

*GROUP BY D. country, C.city*

*ORDER BY COUNT (A.customer\_id) DESC LIM****IT 10;***

*--select and display first name, last name, customer\_id, sum of amount from 5 top customers from top 10 cities, city, and country, arranged in descending order.*

*SELECT C.city AS City, D.country AS Country, A.customer\_id AS Customer\_ID,*

*A.first\_name AS Customer\_First\_Name, A.last\_name as Last\_Name,*

*SUM(F.amount) AS Total\_Amount\_Paid*

*FROM customer A*

*INNER JOIN address B ON A.address\_id = B.address\_id*

*INNER JOIN city C ON B.city\_id = C.city\_id*

*INNER JOIN country D ON C.country\_ID = D.country\_ID*

*INNER JOIN rental E ON A.customer\_id = E.customer\_id*

*INNER JOIN payment F ON E.rental\_id = F.rental\_ID*

*WHERE C.city IN('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Ambattur', 'Shanwei',*

*'So Leopoldo', 'Teboksary', 'Tianjin', 'Cianjur')*

*GROUP BY C.city, D. country, A.customer\_id, A.first\_name, A.last\_name*

*ORDER BY SUM(F.amount) DESC LIMIT 5;*

**LEFT JOIN**

*SELECT DISTINCT(A.country),*

*COUNT(DISTINCT D.customer\_id) AS all\_customer\_count,*

*COUNT(DISTINCT A.country)AS top\_customer\_count*

*FROM country A*

*INNER JOIN city B ON A.country\_ID =B.country\_ID*

*INNER JOIN address C ON B.city\_id =C.city\_id*

*INNER JOIN customer D ON C.address\_id = D.address\_id*

*LEFT JOIN(SELECT C.city AS City, D.country AS Country, A.customer\_id AS Customer\_ID,*

*A.first\_name AS Customer\_First\_Name, A.last\_name as Last\_Name,*

*SUM(F.amount) AS Total\_Amount\_Paid*

*FROM customer A*

*INNER JOIN address B ON A.address\_id = B.address\_id*

*INNER JOIN city C ON B.city\_id = C.city\_id*

*INNER JOIN country D ON C.country\_ID = D.country\_ID*

*INNER JOIN rental E ON A.customer\_id = E.customer\_id*

*INNER JOIN payment F ON E.rental\_id = F.rental\_ID*

*WHERE D.country IN('India', 'China', 'United States', 'Japan', 'Mexico',*

*'Brazil', 'Russian Federation', 'Phillipines', 'Turkey', 'Indonesia')*

*GROUP BY C.city, D. country, A.customer\_id, A.first\_name, A.last\_name*

*ORDER BY SUM(F.amount) DESC LIMIT 5) AS top\_5\_customers*

*ON A.country=top\_5\_customers.country*

*GROUP BY A.country, top\_5\_customers*

*ORDER BY all\_customer\_count DESC*

*LIMIT 5;*

**CTE Queries**

*WITH average\_total\_amount\_cte(city, country, customer\_id, first\_name, last\_name, amount) AS*

*(SELECT C.city, D.country, A.customer\_id, A.first\_name, A.last\_name, SUM(F.amount)*

*FROM customer A*

*INNER JOIN address B ON A.address\_id =B.address\_id*

*INNER JOIN city C ON B.city\_id =C.city\_id*

*INNER JOIN country D ON C.country\_Id =D.country\_Id*

*INNER JOIN rental E ON A.customer\_id =E.customer\_id*

*INNER JOIN payment F ON E.rental\_id =F.rental\_Id*

*WHERE C.city IN('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Ambattur', 'Shanwei',*

*'So Leopoldo', 'Teboksary', 'Tianjin', 'Cianjur')*

*GROUP BY C.city, D.country, A.customer\_id, A.first\_name, A.last\_name*

*ORDER BY SUM(F.amount) DESC LIMIT 5)*

*SELECT AVG(amount)*

*FROM average\_total\_amount\_cte*

*B. WITH top\_5\_customers\_cte(amount, customer\_id, first\_name, Last\_name, city, country, Total\_Amount\_Paid) AS*

*(SELECT B.customer\_id, B.first\_name, A.amount, B.last\_name,*

*D.city, E.country, SUM(amount) AS Total\_Amount\_Paid*

*FROM payment A*

*INNER JOIN customer B ON A.customer\_id =B.customer\_id*

*INNER JOIN address C ON B.address\_id =C.address\_id*

*INNER JOIN city D ON C.city\_id =D.city\_id*

*INNER JOIN country E ON D.country\_id =E.country\_id*

*WHERE country IN('India', 'China', 'United States', 'Japan', 'Mexico',*

*'Brazil', 'Russian Federation', 'Phillipines', 'Turkey', 'Indonesia')*

*GROUP BY D.city, E.country, B.customer\_id, B.first\_name, B.last\_name, A.amount*

*ORDER BY SUM(amount) DESC LIMIT 5), top\_5\_customers AS (SELECT D.country,*

*COUNT(DISTINCT A.customer\_id) AS all\_customer\_count, COUNT(DISTINCT D.country) AS top\_customer\_count*

*FROM customer A*

*INNER JOIN address B ON A.address\_id =B.address\_id*

*INNER JOIN city C ON B.city\_id =C.city\_id*

*INNER JOIN country D ON C.country\_id =D.country\_id*

*GROUP BY D.country)*

*SELECT D.country,*

*COUNT(DISTINCT A.customer\_id) AS all\_customer\_count,*

*COUNT(DISTINCT top\_5\_customers\_cte.customer\_id) AS top\_customer\_count*

*FROM customer A*

*INNER JOIN address B ON A.address\_id =B.address\_id*

*INNER JOIN city C ON B.city\_id =C.city\_id*

*INNER JOIN country D ON C.country\_id =D.country\_id*

*LEFT JOIN top\_5\_customers\_cte ON D.country =top\_5\_customers\_cte.country*

*GROUP BY D.country*

*ORDER BY all\_customer\_count DESC*