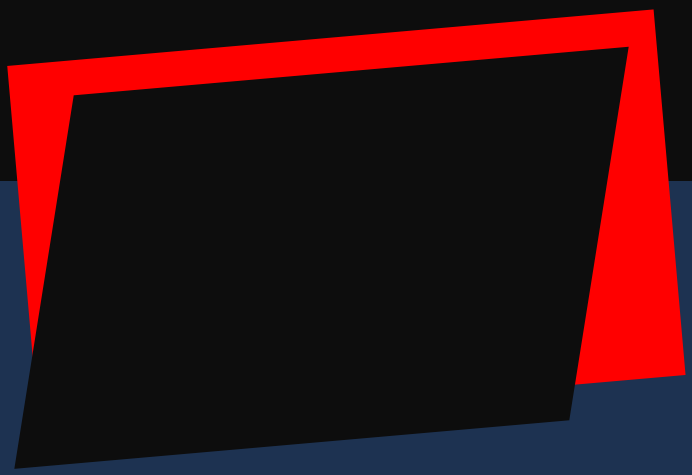
NETFLIX

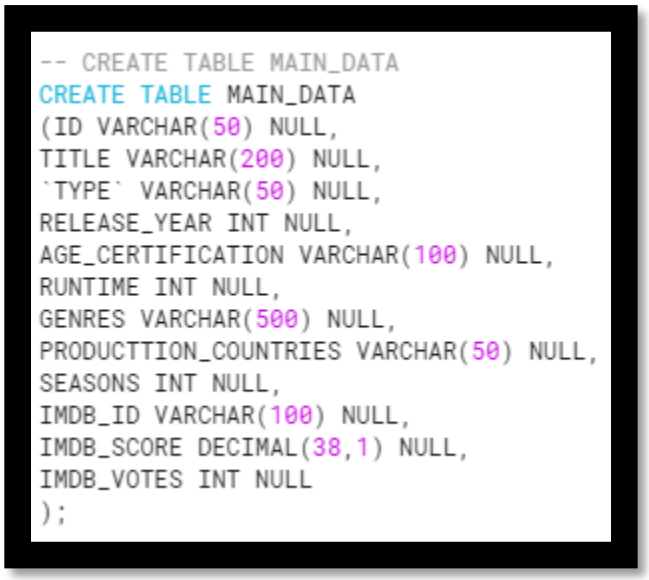


Data Analysis in Snowflake From 2018 till 2022

1: Creating and using NETFLIX Database in Snowflake

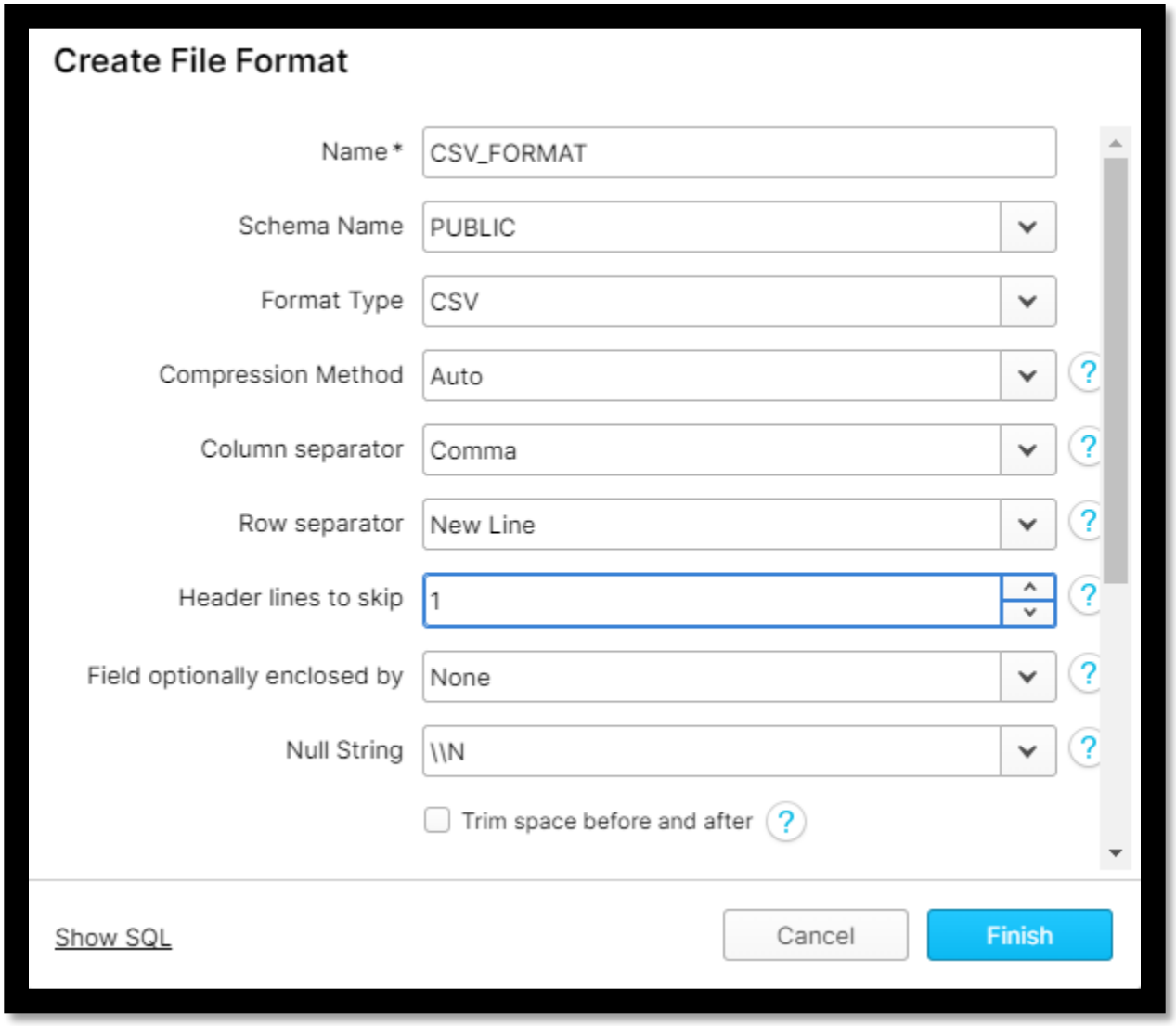
```
-- CREATE DATABASE
CREATE DATABASE NETFLIX;

-- USE DATABASE
USE NETFLIX;
```

2: Creating Main Table Structure Which Contains Movies & Shows Data

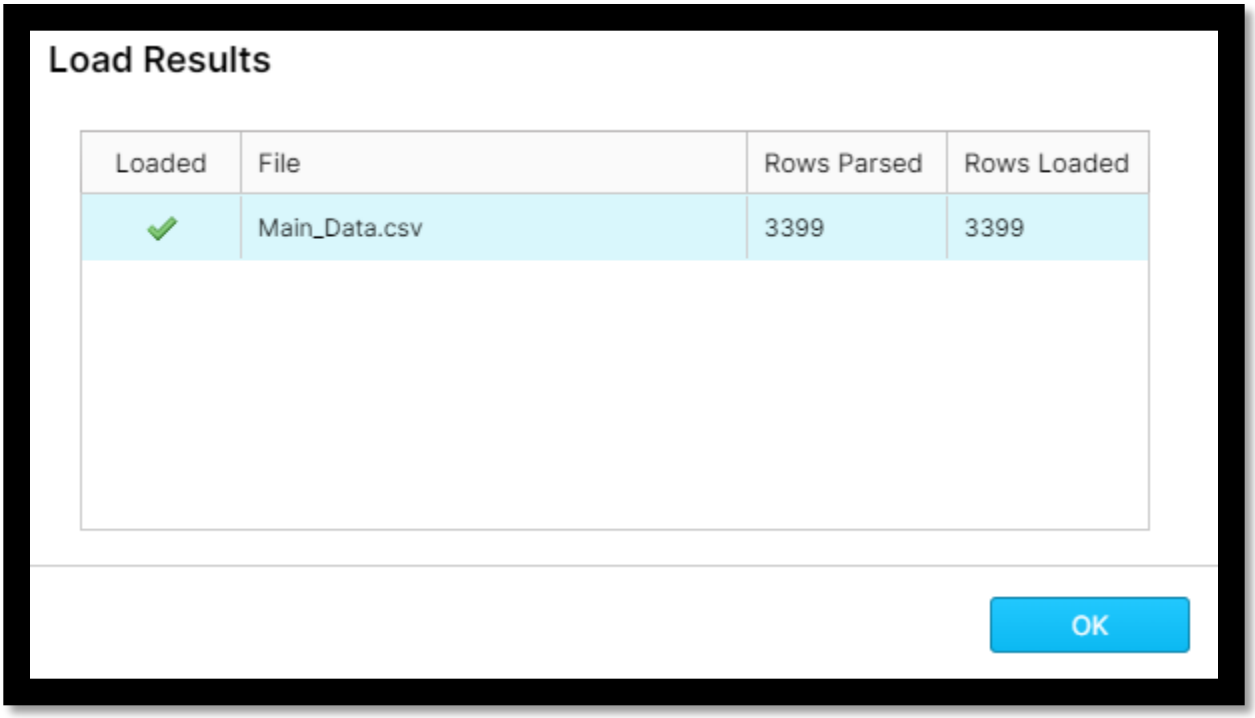


```
-- CREATE TABLE MAIN_DATA
CREATE TABLE MAIN_DATA
(ID VARCHAR(50) NULL,
TITLE VARCHAR(200) NULL,
`TYPE` VARCHAR(50) NULL,
RELEASE_YEAR INT NULL,
AGE_CERTIFICATION VARCHAR(100) NULL,
RUNTIME INT NULL,
GENRES VARCHAR(500) NULL,
PRODUCTTION_COUNTRIES VARCHAR(50) NULL,
SEASONS INT NULL,
IMDB_ID VARCHAR(100) NULL,
IMDB_SCORE DECIMAL(38,1) NULL,
IMDB_VOTES INT NULL
);
```

3: Defining The CSV Format in Snowflake Before Uploading the Dataset



4: Upload the Dataset in Main_Data Table

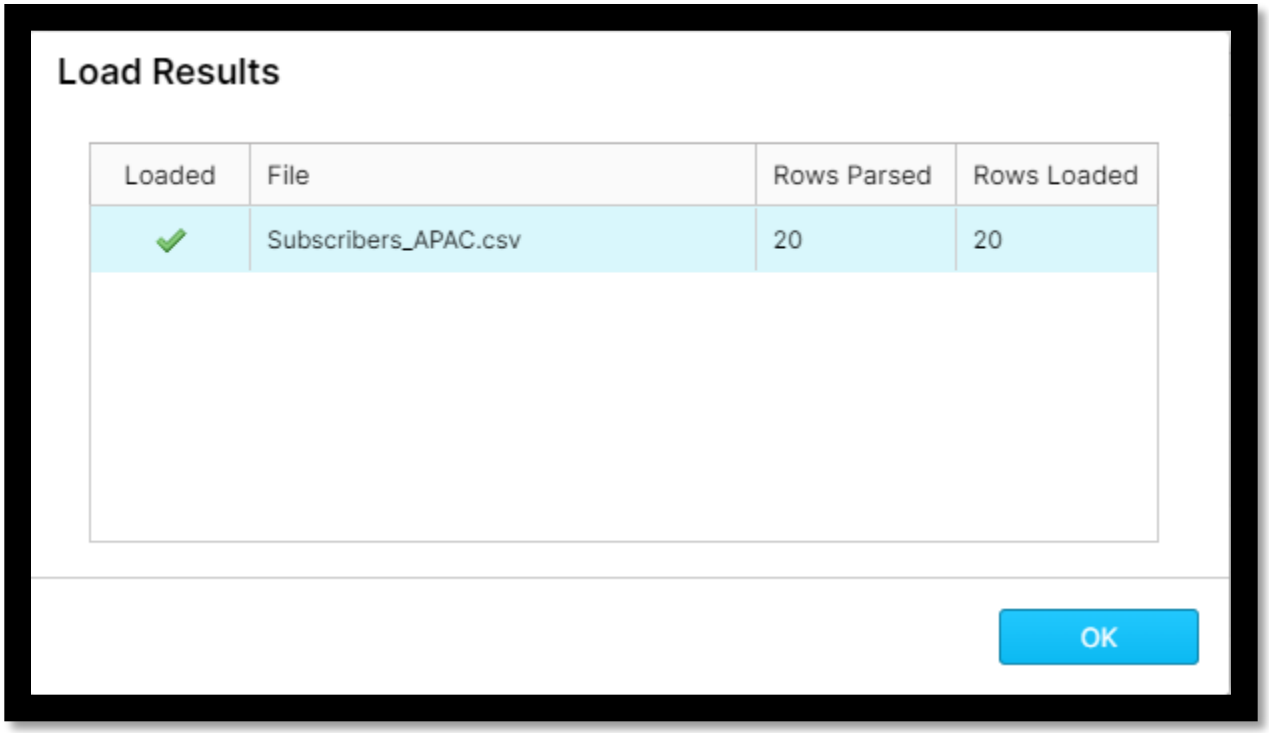


Similarly, we need to upload other tables as well and then we will be ready to do our Analysis on the Data in Snowflake using SQL queries.

5: Create a table for Asia Pacific Region known as APAC

```
-- CREATE TABLE APAC
CREATE TABLE APAC
(YEARS INT,
QUARTERS VARCHAR(20),
`DATE` DATE,
QUARTER_REVENUE_USD INT,
SUBSCRIBERS_ADDITION INT,
TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD INT,
AVERAGE_PAYING_MEMBERS INT,
AVERAGE_REVENUE_PER_MEMBERSHIP_USD DECIMAL(38,2),
PERCENT_CHANGE_COMPARED_TO_PREVIOUS_YEAR INT,
CURRENCY_PERCENT_CHANGE_AS_COMPARED_TO_PREVIOUS_YEAR_PERIOD INT
);
```

6: Upload the dataset in APAC table



7: Create a table for Europe, Middle-East & Africa Region known as EMEA

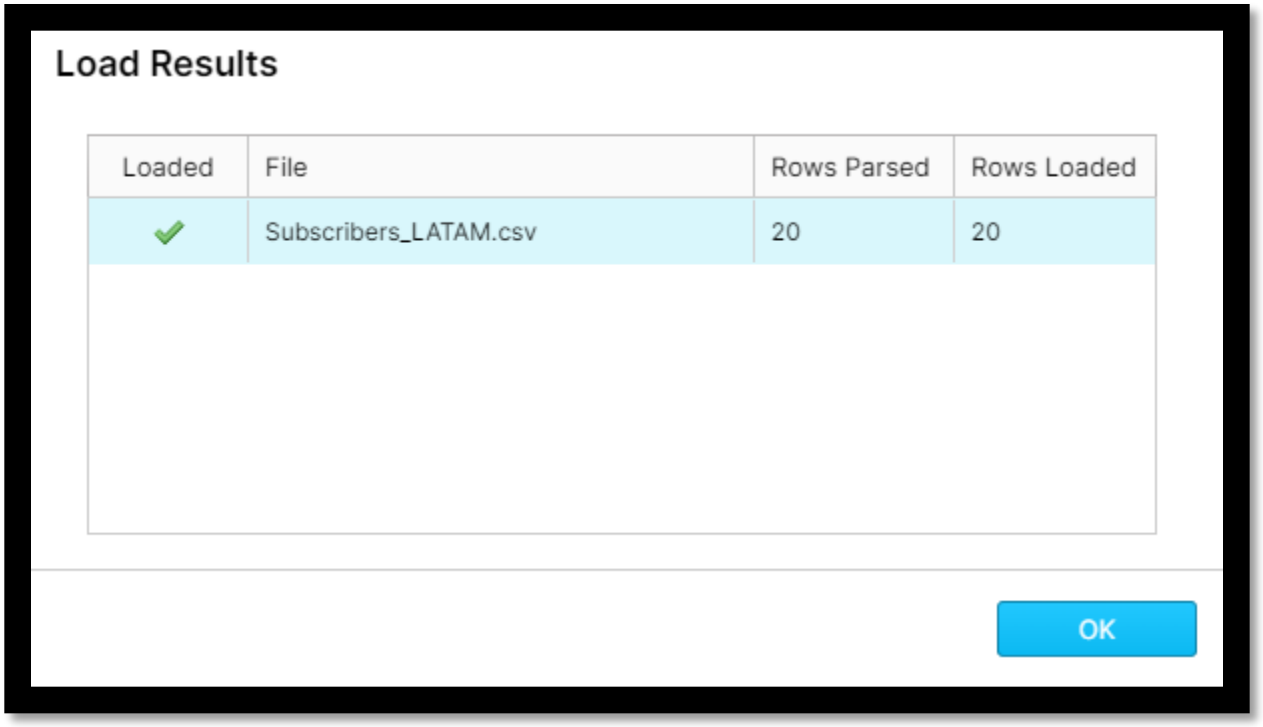
```
-- CREATE TABLE EMEA
CREATE TABLE EMEA
(YEARS INT,
QUARTERS VARCHAR(20),
`DATE` DATE,
QUARTER_REVENUE_USD INT,
SUBSCRIBERS_ADDITION INT,
TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD INT,
AVERAGE_PAYING_MEMBERS INT,
AVERAGE_REVENUE_PER_MEMBERSHIP_USD DECIMAL(38,2),
PERCENT_CHANGE_COMPARED_TO_PREVIOUS_YEAR INT,
CURRENCY_PERCENT_CHANGE_AS_COMPARED_TO_PREVIOUS_YEAR_PERIOD INT
);
```

8: Upload the dataset in EMEA table

9: Create a table for Latin America Region known as LATAM

```
-- CREATE TABLE LATAM
CREATE TABLE LATAM
(YEARS INT,
QUARTERS VARCHAR(20),
`DATE` DATE,
QUARTER_REVENUE_USD INT,
SUBSCRIBERS_ADDITION INT,
TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD INT,
AVERAGE_PAYING_MEMBERS INT,
AVERAGE_REVENUE_PER_MEMBERSHIP_USD DECIMAL(38,2),
PERCENT_CHANGE_COMPARED_TO_PREVIOUS_YEAR INT,
CURRENCY_PERCENT_CHANGE_AS_COMPARED_TO_PREVIOUS_YEAR_PERIOD INT
);
```

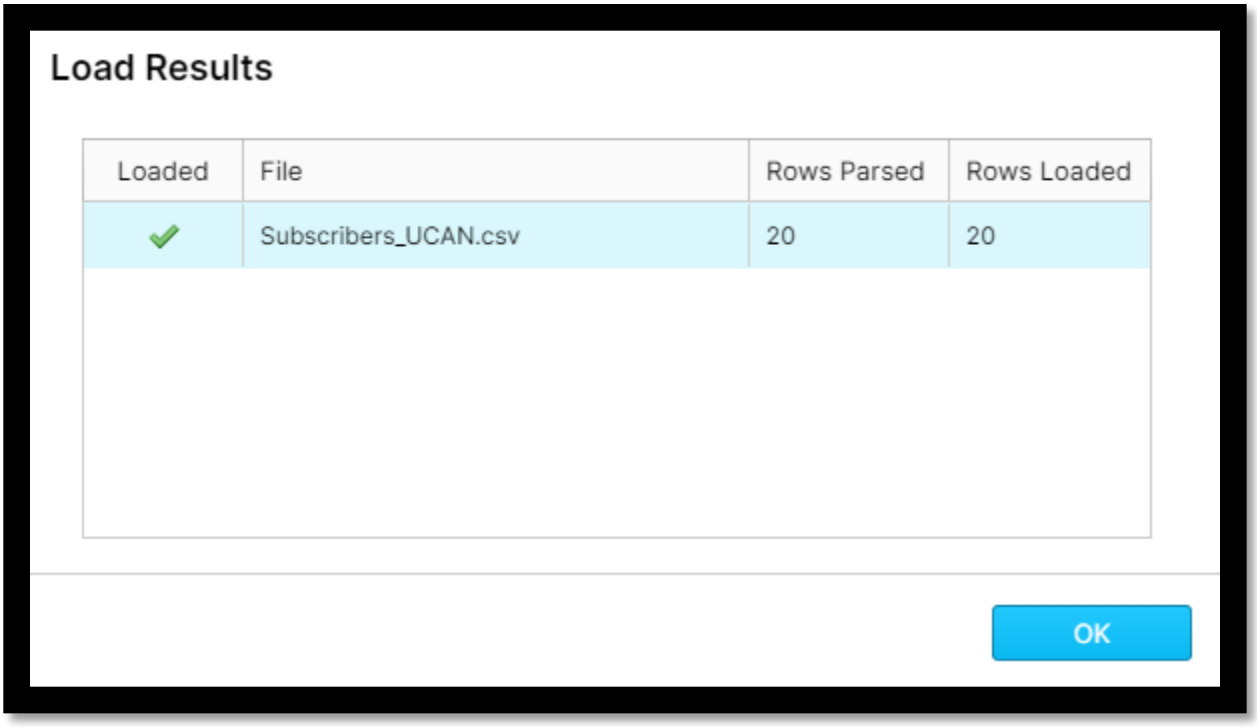
10: Upload the dataset in LATAM table



11: Create a table for the United States & Canada Region known as UCAN

```
-- CREATE TABLE UCAN
CREATE TABLE UCAN
(YEARS INT,
QUARTERS VARCHAR(20),
`DATE` DATE,
QUARTER_REVENUE_USD INT,
SUBSCRIBERS_ADDITION INT,
TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD INT,
AVERAGE_PAYING_MEMBERS INT,
AVERAGE_REVENUE_PER_MEMBERSHIP_USD DECIMAL(38,2),
PERCENT_CHANGE_COMPARED_TO_PREVIOUS_YEAR INT,
CURRENCY_PERCENT_CHANGE_AS_COMPARED_TO_PREVIOUS_YEAR_PERIOD INT
);
```

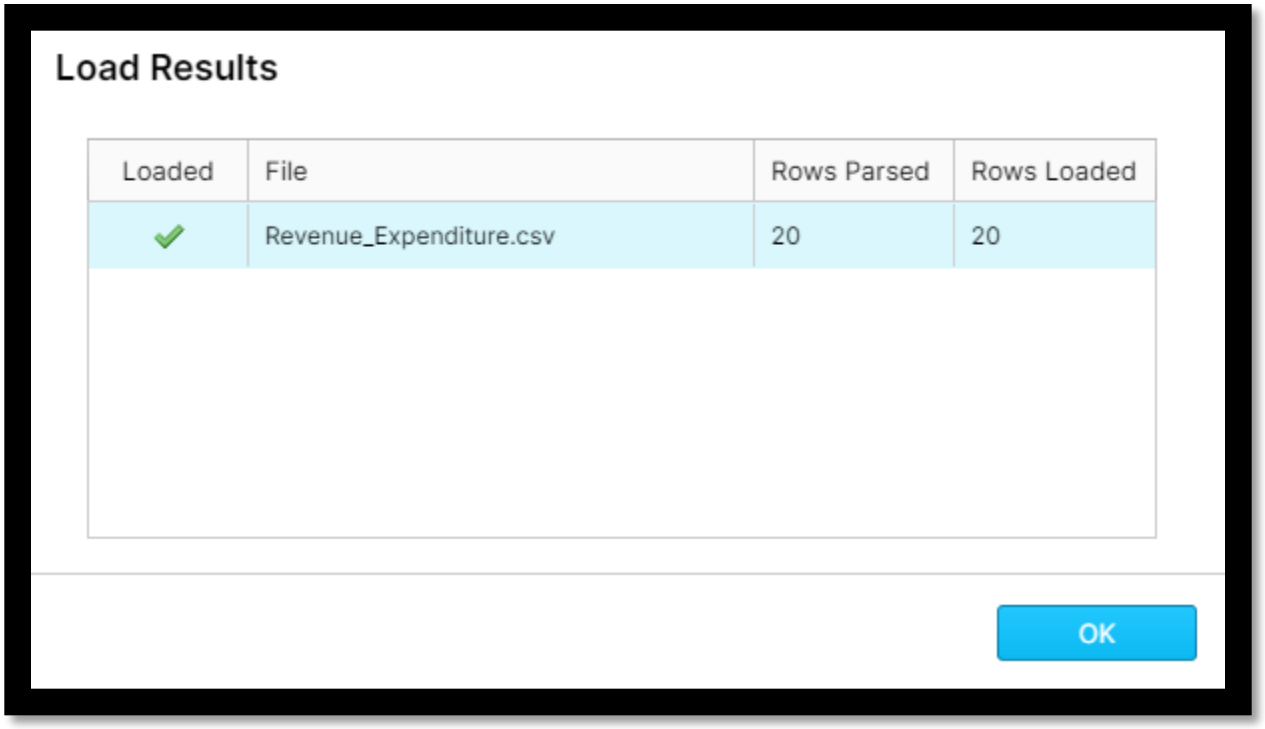
12: Upload the dataset in UCAN table



13: Create Revenue_Expenditure Table

```
-- CREATE TABLE NAMED REVENUE_EXPENDITURE
CREATE TABLE REVENUE_EXPENDITURE
( YEARS INT,
  QUARTER VARCHAR(20),
  `DATE` DATE,
   REVENUE INT,
   COST_OF_REVENUE INT,
   MARKETING_EXPEDITURE INT,
   TECHNOLOGY_AND_DEVELOPMENT_COST INT,
   GENERAL_AND_ADMINISTRATIVE_EXPENSES INT,
   NET_INCOME_AFTER_TAXES INT
);
```

14: Upload data in Revenue_Expenditure Table



15: Create MASTER_NETFLIX_REVENUE_EXPENDITURE Table With the help of inner join

```
-- CONSOLIDATE ALL THE DATA OF REVENUE, EXPENDITURE AND REGION_WISE REVENUE
CREATE OR REPLACE TABLE MASTER_NETFLIX_REVENUE_EXPENDITURE AS
SELECT A.*, B.QUARTER_REVENUE_USD AS APAC_REVENUE,B.SUBSCRIBERS_ADDITION AS APAC_SUBSCRIBERS_ADDITION,B.TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD AS
APAC_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD ,B.AVERAGE_PAYING_MEMBERS AS APAC_AVG_PAYING_MEMBERS,
B.AVERAGE_REVENUE_PER_MEMBERSHIP_USD AS APAC_AVG_REV_PER_MEMBER,
C.QUARTER_REVENUE_USD AS LATAM_REVENUE, C.SUBSCRIBERS_ADDITION AS LATAM_SUBSCRIBERS_ADDITION,C.TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD AS
LATAM_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD ,C.AVERAGE_PAYING_MEMBERS AS LATAM_AVG_PAYING_MEMBERS,
C.AVERAGE_REVENUE_PER_MEMBERSHIP_USD AS LATAM_AVG_REV_PER_MEMBER,
D.QUARTER_REVENUE_USD AS EMEA_REVENUE, D.SUBSCRIBERS_ADDITION AS EMEA_SUBSCRIBERS_ADDITION,D.TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD AS
EMEA_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD ,D.AVERAGE_PAYING_MEMBERS AS EMEA_AVG_PAYING_MEMBERS,
D.AVERAGE_REVENUE_PER_MEMBERSHIP_USD AS EMEA_AVG_REV_PER_MEMBER,
E.QUARTER_REVENUE_USD AS UCAN_REVENUE, E.SUBSCRIBERS_ADDITION AS UCAN_SUBSCRIBERS_ADDITION,E.TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD AS
UCAN_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD ,E.AVERAGE_PAYING_MEMBERS AS UCAN_AVG_PAYING_MEMBERS,
E.AVERAGE_REVENUE_PER_MEMBERSHIP_USD AS UCAN_AVG_REV_PER_MEMBER
FROM REVENUE_EXPENDITURE AS A
INNER JOIN APAC AS B ON A.`DATE` = B.`DATE`
INNER JOIN LATAM AS C ON B.`DATE` = C.`DATE`
INNER JOIN EMEA AS D ON C.`DATE` = D.`DATE`
INNER JOIN UCAN AS E ON D.`DATE` = E.`DATE`;
```

Inner Join is performed on the date column as it’s a common column in all tables across all regions.

16: Create MASTER NETFLIX_DATA_MOVIES_SHOWS Table from MAIN_DATA table with the help of Window functions

```
-- MASTER NETFLIX_DATA_MOVIES_SHOWS
CREATE TABLE NETFLIX_DATA_MOVIES_SHOWS AS
SELECT RELEASE_YEAR, COUNT(`TYPE`) AS MOVIES_SHOWS_COUNT, AVG(RUNTIME) AS AVG_RUNTIME_MINUTES, MAX(RUNTIME) AS HIGHEST_RUNTIME_MINUTES,
MIN(RUNTIME) AS LOWEST_RUNTIME_MINUTES,
AVG(IMDB_SCORE) AS AVG_IMDB_SCORE, MAX(IMDB_SCORE) AS HIGHEST_IMDB_SCORE, MIN(IMDB_SCORE) AS LOWEST_IMDB_SCORE,
AVG(IMDB_VOTES) AS AVG_IMDB_VOTES, MAX(IMDB_VOTES) AS HIGHEST_IMDB_VOTES, MIN(IMDB_VOTES) AS LOWEST_IMDB_VOTES
FROM MAIN_DATA
GROUP BY RELEASE_YEAR
ORDER BY RELEASE_YEAR;
```

Here, we have used Average, Minimum, Maximum and have grouped by Release_Year.

17: Create SUMMARY_REVENUE_EXPENDITURE from MASTER_NETFLIX_REVENUE_EXPENDITURE Year wise

```
-- SUMMARY_REVENUE_EXPENDITURE
CREATE TABLE SUMMARY_REVENUE_EXPENDITURE AS

SELECT YEARS, SUM(REVENUE) AS TOTAL_REVENUE_USD, SUM(COST_OF_REVENUE) AS TOTAL_COST_OF_REVENUE_USD, SUM(MARKETING_EXPEDITURE) AS MARKETING_EXP_USD,
SUM(TECHNOLOGY_AND_DEVELOPMENT_COST) AS TECH_DEV_COST_USD, SUM(GENERAL_AND_ADMINISTRATIVE_EXPENSES) AS ADMIN_EXP_USD, SUM(NET_INCOME_AFTER_TAXES) AS PROFITS_USD,

SUM(APAC_REVENUE) AS APAC_REVENUE_USD, SUM(APAC_SUBSCRIBERS_ADDITION) AS APAC_SUBSCRIBERS_ADDITION, SUM(APAC_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD) AS
APAC_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD,
AVG(APAC_AVG_PAYING_MEMBERS) AS APAC_AVG_PAYING_MEMBERS, AVG(APAC_AVG_REV_PER_MEMBER) AS APAC_AVG_REV_PER_MEMBER_USD,

SUM(LATAM_REVENUE) AS LATAM_REVENUE_USD, SUM(LATAM_SUBSCRIBERS_ADDITION) AS LATAM_SUBSCRIBERS_ADDITION, SUM(LATAM_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD) AS
LATAM_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD,
AVG(LATAM_AVG_PAYING_MEMBERS) AS LATAM_AVG_PAYING_MEMBERS, AVG(LATAM_AVG_REV_PER_MEMBER) AS LATAM_AVG_REV_PER_MEMBER_USD,

SUM(EMEA_REVENUE) AS EMEA_REVENUE_USD, SUM(EMEA_SUBSCRIBERS_ADDITION) AS EMEA_SUBSCRIBERS_ADDITION, SUM(EMEA_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD) AS
EMEA_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD,
AVG(EMEA_AVG_PAYING_MEMBERS) AS EMEA_AVG_PAYING_MEMBERS, AVG(EMEA_AVG_REV_PER_MEMBER) AS EMEA_AVG_REV_PER_MEMBER_USD,

SUM(UCAN_REVENUE) AS UCAN_REVENUE_USD, SUM(UCAN_SUBSCRIBERS_ADDITION) AS UCAN_SUBSCRIBERS_ADDITION, SUM(UCAN_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD) AS
UCAN_TOTAL_SUBSCRIBERS_AT_END_OF_PERIOD,
AVG(UCAN_AVG_PAYING_MEMBERS) AS UCAN_AVG_PAYING_MEMBERS, AVG(UCAN_AVG_REV_PER_MEMBER) AS UCAN_AVG_REV_PER_MEMBER_USD

FROM MASTER_NETFLIX_REVENUE_EXPENDITURE
GROUP BY YEARS
ORDER BY YEARS;
```

Summing all the values in the regions and grouping year wise to get the summary of the Revenue_Expenditure

18: Removing the delimiter in Genres and putting it in GENRES_2018_22 Table

```
-- SPLIT THE TEXT WITH DELIMITER FOR GENRES
CREATE TABLE GENRES_2018_22 AS
SELECT RELEASE_YEAR, C.value::string AS GENRES
FROM MAIN_DATA,
     LATERAL FLATTEN(input=>split(GENRES, ';')) C;
```

19: Find Out which genres are more in respective areas

```
-- FIND OUT WHICH GENRES ARE MORE IN RESPECTIVE YEARS
SELECT * FROM (
SELECT RELEASE_YEAR, GENRES, COUNT(*) AS MAX_GENRES,
ROW_NUMBER() OVER (PARTITION BY RELEASE_YEAR ORDER BY MAX_GENRES DESC) AS ROW_NUM
FROM GENRES_2018_22
GROUP BY RELEASE_YEAR, GENRES
ORDER BY RELEASE_YEAR)
AS GENRES_DATA
WHERE ROW_NUM = 1;
```

RELEASE_YEAR	GENRES	MAX_GENRES	ROW_NUM
2018	drama	186	1
2019	drama	218	1
2020	drama	185	1
2021	drama	188	1
2022	drama	53	1

We used concept of row numbers which helps us get the highest genres in each year and we can clearly see that Drama genre is the highest in Netflix across all years from 2018 till 2022.

20: Split the delimiter for production_countries and put it in table PRODUCTION_COUNTRIES_2018_22

```
-- SPLIT THE TEXT WITH DELIMITER FOR PRODUCTTION_COUNTRIES
CREATE TABLE PRODUCTION_COUNTRIES_2018_22 AS
SELECT RELEASE_YEAR, C.value::string AS PRODUCTION_COUNTRIES
FROM MAIN_DATA,
     LATERAL FLATTEN(input=>split(PRODUCTTION_COUNTRIES, ';')) C;
```

21: Count Production Countries

```
SELECT RELEASE_YEAR, PRODUCTION_COUNTRIES, COUNT(*) AS COUNT_PRODUCTION_COUNTRIES
FROM PRODUCTION_COUNTRIES_2018_22
GROUP BY RELEASE_YEAR, PRODUCTION_COUNTRIES;
```

22; Which Production Countries are more in respective years

```
-- FIND OUT WHICH PRODUCTION_COUNTRIES ARE MORE IN RESPECTIVE YEARS
SELECT * FROM (
SELECT RELEASE_YEAR, PRODUCTION_COUNTRIES, COUNT(*) AS MAX_PRODUCTION_COUNTRIES,
ROW_NUMBER() OVER (PARTITION BY RELEASE_YEAR ORDER BY MAX_PRODUCTION_COUNTRIES DESC) AS ROW_NUM
FROM PRODUCTION_COUNTRIES_2018_22
GROUP BY RELEASE_YEAR, PRODUCTION_COUNTRIES
ORDER BY RELEASE_YEAR)
AS PRODUCTION_COUNTRIES_DATA
WHERE ROW_NUM = 1;
```

RELEASE_YEAR	PRODUCTION_COUNTRIES	MAX_PRODUCTION_COUNTRIES	ROW_NUM
2018	US	257	1
2019	US	286	1
2020	US	291	1
2021	US	231	1
2022	US	66	1

Here, we are again using row numbers to get the result. We get to know US is contributing highest in Netflix shows and movies production.

23: Creating a Master table with All Region Data in One table with Region Names

```
-- CREATE AN ADDITIONAL COLUMNS IN EACH SUBSCRIBERS SHEET
ALTER TABLE APAC ADD COLUMN REGION VARCHAR;
ALTER TABLE EMEA ADD COLUMN REGION VARCHAR;
ALTER TABLE LATAM ADD COLUMN REGION VARCHAR;
ALTER TABLE UCAN ADD COLUMN REGION VARCHAR;
```

First Alter table to add a region column and then Update the columns with region names

```
--UPDATE THE REGION COLUMN IN EACH SHEET
UPDATE APAC SET REGION = 'APAC';
UPDATE EMEA SET REGION = 'EMEA';
UPDATE LATAM SET REGION = 'LATAM';
UPDATE UCAN SET REGION = 'UCAN';
```

Now Create a table as MASTER_REGION with all region data in one

```
CREATE TABLE MASTER_REGION AS
SELECT * FROM APAC
UNION ALL
SELECT * FROM EMEA
UNION ALL
SELECT * FROM LATAM
UNION ALL
SELECT * FROM UCAN;
```

We have used the concept of union all which gives data below the previous table and includes all records.

24: Count of Movie & Shows in Netflix Data

```
-- COUNT OF MOVIES AND SHOWS IN NETFLIX DATA
SELECT `TYPE`, COUNT(`TYPE`) AS COUNT_MOVIE_SHOW
FROM MAIN_DATA
GROUP BY `TYPE`;
```

`TYPE`	COUNT_MOVIE_SHOW
SHOW	1327
MOVIE	2072

Movies are 2072 and Shows are 1327. Thus, Movies are more compared to Shows from 2018 to 2022.

25: What is the highest runtime in shows and movies

```
-- What is the highest runtime in shows and movies
SELECT `TYPE`, MAX(RUNTIME) AS MAX_RUNTIME FROM MAIN_DATA
GROUP BY `TYPE`;
```

`TYPE`	MAX_RUNTIME
SHOW	199
MOVIE	209

The maximum run-time for movies is 209 minutes and shows is 199 minutes

26: How many shows and movies are there whose Genre is comedy

```
-- HOW MANY SHOWS AND MOVIES ARE THERE WHOSE GENRE IS COMEDY
SELECT `TYPE`, COUNT(GENRES) AS COMEDY FROM MAIN_DATA
WHERE GENRES LIKE '%comedy%'
GROUP BY `TYPE`;
```

`TYPE`	COMEDY
SHOW	391
MOVIE	799

There are 799 Movies and 391 Shows which has genres as Comedy.

27: Which Movie or Show has got highest IMDB score

```
-- WHICH MOVIE OR SHOW HAS GOT HIGHEST IMDB SCORE
SELECT RELEASE_YEAR ,TITLE, `TYPE`, IMDB_SCORE
FROM MAIN_DATA
WHERE IMDB_SCORE = (SELECT MAX(IMDB_SCORE) FROM MAIN_DATA);
```

RELEASE_YEAR	TITLE	`TYPE`	IMDB_SCORE
2018	#ABtalks	SHOW	9.6

Highest IMDB score is 9.6 scored by #ABtalks SHOW

28: What’s the highest IMDB score for movies

```
-- WHATS THE HIGHEST IMDB SCORE FOR MOVIE
SELECT RELEASE_YEAR, TITLE,`TYPE`,GENRES, IMDB_SCORE FROM MAIN_DATA
WHERE IMDB_SCORE IN
(SELECT MAX(IMDB_SCORE)
FROM MAIN_DATA
WHERE `TYPE` = 'MOVIE');
```

RELEASE_YEAR	TITLE	`TYPE`	GENRES	IMDB_SCORE
2018	C/o Kancharapalem	MOVIE	drama	9.0
2020	David Attenborough: A Life on Our Pla...	MOVIE	documentation	9.0

C/o Kancharapalem & A Life on Our planet has highest IMDB score of 9

29: Which Movie has got the highest votes

```
-- HIGHEST VOTE
SELECT RELEASE_YEAR ,TITLE, `TYPE`, IMDB_VOTES
FROM MAIN_DATA
WHERE IMDB_VOTES IN (SELECT MAX(IMDB_VOTES) FROM MAIN_DATA);
```

RELEASE_YEAR	TITLE	`TYPE`	IMDB_VOTES
2021	Dont Look Up	MOVIE	498447

Don’t Look Up Movie has got highest vote in 2021. The Number of Votes are 498447

30: Which Show has got the highest votes

```
-- HIGHEST VOTE FOR SHOWS
SELECT RELEASE_YEAR ,TITLE, `TYPE`, IMDB_VOTES
FROM MAIN_DATA
WHERE IMDB_VOTES IN (SELECT MAX(IMDB_VOTES) FROM MAIN_DATA WHERE `TYPE`= 'SHOW');
```

RELEASE_YEAR	TITLE	`TYPE`	IMDB_VOTES
2019	The Witcher	SHOW	465949

The Witcher Show has got highest votes 465949

31: Number of movies and Shows released year wise

```
-- COUNT OF MOVIES AND SHOWS RELEASED YEAR WISE
SELECT RELEASE_YEAR, `TYPE`, COUNT(TITLE) AS NUMBER_OF_RELEASES
FROM MAIN_DATA
GROUP BY 2,1;
```

RELEASE_YEAR	`TYPE`	NUMBER_OF_RELEASES
2018	SHOW	301
2018	MOVIE	472
2019	SHOW	308
2019	MOVIE	539
2020	SHOW	306
2020	MOVIE	499
2022	SHOW	109
2022	MOVIE	108
2021	MOVIE	454
2021	SHOW	303

FURTHER EXPLORATORY DATA ANALYSIS IS DONE IN PYTHON.
CHECK THE PYTHON EDA FOLDER

GITHUB LINK: - https://github.com/vishalchugh24/Netflix

LINKEDIN LINK ; - https://www.linkedin.com/in/vishal-chugh-profile/