IMDB MIDTERM PROJECT

GROUP 7: Je Sai Kailash Pulipati (002339774)
Deepthi Ramesh (002339269)
Geetika Barla (002372565)

DATASET OVERVIEW:

The Internet Movie Database (IMDb) dataset is a publicly available collection of structured information related to movies, TV shows, actors, directors, crew members, and ratings. It is primarily used for research, data analysis, and machine learning applications. The dataset is available for non-commercial use and is regularly updated.
IMDb provides this data in the TSV (Tab-Separated Values) format within compressed .gz files. The dataset includes a unique set of identifiers for each entity, enabling structured querying and relationship building between different files.

DATASET FILES AND THEIR CONTENTS:

Basic information for IMDB Datasets :

DATASET	NUMBER OF RECORDS	TOTAL FIELDS
Name.basics.tsv	1,41,95,120	6
Title.akas.tsv	5,14,09,880	8
title.crew.tsv	1,14,64,885	3
title.basics.tsv	11464885	9
title.episode.tsv	8815771	4
title.principals.tsv	9,09,84,102	6
title.ratings.tsv	15,36,010	3

1. **name.basics.tsv.gz** (People database)

Column Name	Data type	Description
nconst	String	Unique identifier for a person (e.g., nm0000102)
primaryName	String	Name of the person (e.g., “Robert De Niro”)

birthYear	Integer	Year of birth (or \N if unknown)
deathYear	Integer	Year of death (or \N if still alive)
primaryProfession	String (Array)	List of top three professions (e.g., “actor, producer, director”)
knownForTitles	String (Array)	List of IMDb title IDs this person is most known for (e.g., tt0111161, tt0068646)

• This file contains details of actors, directors, writers, and other professionals in the entertainment industry.

Use cases:

- Finding actors and directors associated with multiple movies.
- Analyzing trends in the careers of film industry professionals.

2. **title.bascis.tsv.gz** (Movies & TV Shows)

This file contains basic details about movies, TV series, and short films.

Column Name	Data Type	Description
tconst	String	Unique identifier for a title (e.g., tt0111161)
titleType	String	Type of title (e.g., “movie”, “short”, “tvSeries”)
primaryTitle	String	The most common title used
originalTitle	String	Title in the original language
isAdult	Boolean	0 = Not Adult, 1 = Adult content
startYear	Integer	Year of release or start of TV series
endYear	Integer	Year when a TV series ended (\N for movies)
runtimeMinutes	Integer	Duration of the title in minutes

genres	String (Array)	Up to three genres associated with the title (e.g., “Action, Drama”)

Use Cases:

- Finding the most common genres over time.
- Analyzing trends in movie durations.
- Identifying adult vs. non-adult content in films.

3. **title.akas.tsv.gz** (Alternate Titles)

This file contains different versions of movie and TV show titles in multiple languages and regions.

Column Name	Data Type	Description
titleId	String	Unique identifier of the title (maps to tconst)
ordering	Integer	Unique row identifier for a given titleId
title	String	Localized title
region	String	Region/country code (e.g., “US”, “IN”)
language	String	Language of the title
types	String (Array)	Categorization of the alternate title (e.g., “DVD”, “festival”)
attributes	String (Array)	Additional descriptive attributes
isOriginalTitle	Boolean	1 = Original title, 0 = Not original

Use Cases:

- Analyzing regional naming differences.
- Mapping different versions of movie titles across languages.

4. **title.crew.tsv.gz** ((Directors & Writers)

Column Name	Data Type	Description

tconst	String	Unique title identifier
directors	String (Array)	List of director(s) (maps to nconst)
writers	String (Array)	List of writer(s) (maps to nconst)

This file contains information about movie directors and writers.

Use Cases:

- Finding all movies directed by a specific person.
- Identifying collaborations between directors and writers.

5. **title.episode.tsv.gz** (TV Series & Episodes)

This file links individual TV episodes to their parent TV series.

Column Name	Data Type	Description
tconst	String	Unique identifier for the episode
parentTconst	String	Unique identifier for the TV series
seasonNumber	Integer	Season number of the episode
episodeNumber	Integer	Episode number within the season

Column Name	Data Type	Description
tconst	String	Unique title identifier
ordering	Integer	Ordering number for a person in the credits
nconst	String	Unique person identifier
category	String	Job category (e.g., “actor”, “director”)
job	String	Specific role (if applicable)
characters	String	Character names (if applicable)

Use Cases:

• Analyzing episode trends across seasons.
•. Linking specific episodes to their respective TV series.

6. **title.principals.tsv.gz** (Key Cast & Crew)

This file lists the primary cast and crew members for each title.

Column Name	Data Type	Description
tconst	String	Unique title identifier
ordering	Integer	Ordering number for a person in the credits
nconst	String	Unique person identifier
category	String	Job category (e.g., “actor”, “director”)
job	String	Specific role (if applicable)
characters	String	Character names (if applicable)

Use Cases:

• Finding lead actors in movies.
• Analyzing character appearances in different titles.

7. **title.ratings.tsv.gz** (Ratings & Votes)

Column Name	Data Type	Description
tconst	String	Unique title identifier
averageRating	Float	Average IMDb rating (0-10)
numVotes	Integer	Number of votes for the title

• This file contains IMDb ratings and the number of votes for each movie.

Use Cases:

•. Analyzing the most highly-rated movies.
•. Identifying movies with the most user engagement.

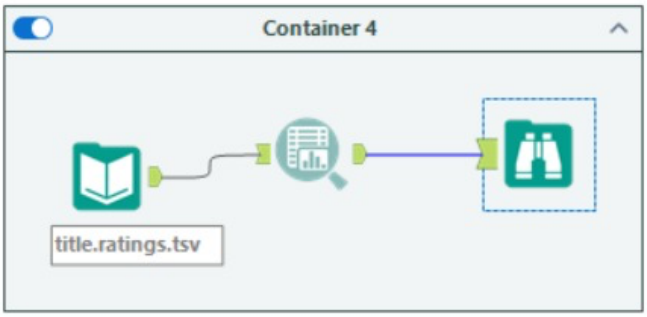
DATA FORMAT AND STRUCTURE

• The data is tab-separated and encoded in UTF-8.
•. Each dataset includes unique identifiers such as tconst (for titles) and nconst (for names).
• Missing values are represented as \N.

•. Data is updated daily to ensure accuracy and relevance.

DATA PROFILING AND ANALYSIS:

Title.rating.tsv

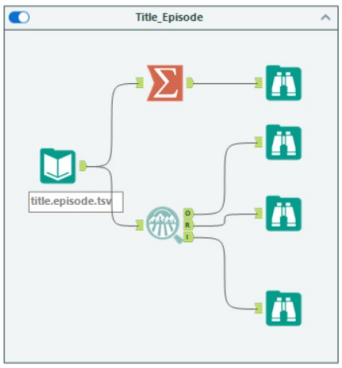


1. The dataset is well-structured, with no missing values in the tconst field, which serves as a unique identifier for movies/shows, containing over 10,000 unique records with a consistent format.

2. The averageRating field is stored as a string instead of a numeric type, which may cause issues in calculations and should be converted to a float/decimal* for proper analysis.

3. Overall, the data is clean and structured, but ensuring the correct data type for averageRating is crucial for accurate visualizations and insights.

Title.episode.tsv

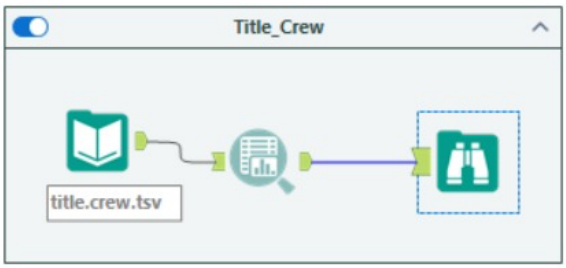


1. The title_episode table links individual TV episodes to their parent series, enabling detailed analysis of season and episode structures. However, a significant number of records contain NULL (\N) values in both seasonNumber and episodeNumber, indicating missing or inapplicable data.

2. Season Numbers range widely, from 1 to at least 315, but many entries lack season details (~1.8M missing). The most frequent values are season 1, followed by seasons 2 and 3, suggesting that most series tend to have shorter runs.

3. Episode Numbers are highly variable, spanning from 1 to over 995, but ~319K entries have missing episode data, potentially indicating special episodes or unstructured series formats. This missing data may impact trend analysis of TV show structures

Title_crew

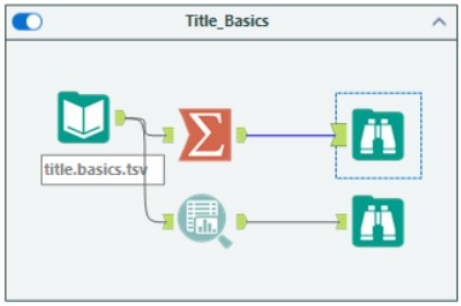


1. The title_crew table is crucial for analyzing directors and writers linked to movies and TV shows. The tconst field (title identifier) is well-structured, with over 11.4 million non-null records**, ensuring that each title has a valid identifier.

2. Director Data: A significant portion of the dataset (4.2 million records) has missing director information (\N values), but 944,672 unique directors are identified, with some appearing frequently across multiple titles.

3. Writer Data: More than 4.8 million entries have missing writers, but 1.4 million unique writers are recorded. To ensure data integrity, missing values can be replaced with "Unknown" , allowing for consistent analysis.

Title_Basics

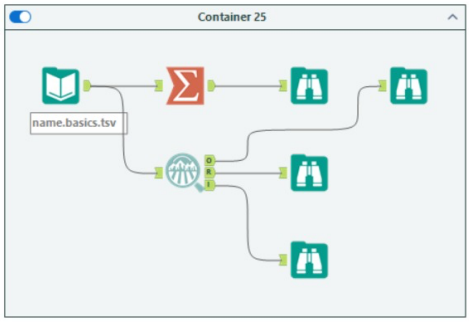


1.Year Data Issues – The dataset includes titles from 1894 to 1947, but many records have missing start (~606K) and end years (~3.65M), affecting time-based analysis.

2. Missing & Inconsistent Titles – Some primary and original titles are missing, while the longest primary title is "Tekken 4" and "petta reddast" is the longest original title.

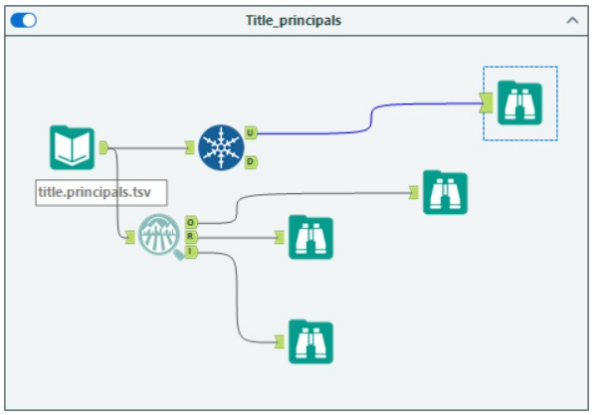
3. Data Cleanup Needed – Null values in genres (~145K) and runtime (~2.38M) should be handled for better categorization and trend analysis.

Name.basics



Some values of this field have a small number of value counts. If Appropriate, consider combining some value levels together.

Title_principals



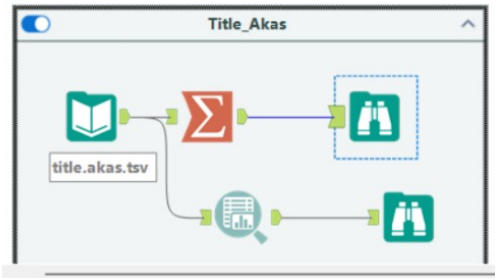
Insights from Title Principals Data Profiling

1.Job Field Issues – A significant portion of records (5.85M out of 6M) lack job titles, indicating that many individuals are credited without a specified role. However, key roles such as Producers (412K) and Writers (114K) are frequently listed.

2. Character Field Gaps – Around 3.6M entries do not have character names, which affects analysis of cast roles. Common values like "[Self]" (257K) and "[Narrator]" (15K) suggest a high number of documentary or real-life portrayals.

3.Data Cleanup Needed – The high number of missing job and character fields requires a standardized approach (e.g., replacing \N with "Unknown"), ensuring better classification and analysis of production roles.

Title_Akas



Insights from Title Akas Data Profiling
1.Title Variations & Data Gaps – The dataset includes over 5.1 million unique title records, but many entries lack regional (~1.4M) and language (~3.2M) information, making regional and linguistic analysis challenging.

2. Common Regions & Languages – Titles are widely available in France (438K), the US (437K), Germany (418K), and Spain (417K), with Japanese (385K), French (373K), and Hindi (345K) as dominant languages.

3.Missing Metadata & Cleanup Needs – A large portion of records lacks type (~1M missing) and attribute (~5.9M missing) classifications, impacting content categorization. Standardizing missing values will improve searchability and analysis.

DATA CLEANING AND TRANSFORMATIONS:

To ensure the IMDb dataset was structured, standardized, and ready for analysis, we performed a series of data cleaning and transformation steps using Alteryx. These steps aimed to enhance data quality, improve consistency, and facilitate easier analysis. The key transformations applied are outlined below:

1. **Handling Missing Values**

The dataset contained missing values, which were represented as \N. To maintain data consistency and avoid null-related errors in analysis, we replaced:

- Text-based missing values with "Unknown"
- Numerical missing values with 0

This standardization ensured that every field had a meaningful or neutral placeholder instead of undefined entries.

2. Trimming Trailing White Spaces

Inconsistent spacing in textual fields can lead to data discrepancies and affect downstream analyses. To mitigate this, we used the Data Cleaning tool in Alteryx to:

- Remove leading and trailing white spaces from all text fields.
- Standardize text formatting, ensuring uniformity across the dataset.

3. Flattening and Normalizing the Dataset

The dataset contained columns with multiple values stored in array-like formats (e.g., lists of directors, writers, or genres). To improve accessibility and facilitate structured querying, we:

- Utilized the Text to Columns tool in Alteryx to split delimited values into separate rows.
- Normalized categorical fields such as genres, professions, and known titles to improve relational mapping.

This process enabled a more granular analysis of relationships between different entities in the dataset.

4. Standardizing Data Types

To align with best practices in data modeling and ensure compatibility with analytical tools, we adjusted column data types:

- Converted VMString (variable-length string) fields to standard String format.
- Changed numerical fields (such as ratings and vote counts) from text-based storage to Float or Integer types as needed.

These conversions improved data integrity and query efficiency for downstream processing.
By applying these data cleaning and transformation steps, we ensured that the IMDb dataset was well-structured, free from inconsistencies, and optimized for further analysis. This refined dataset now enables robust querying, accurate trend identification, and effective insights generation.

5. Mapping Country and Language Data for Enhanced Insights

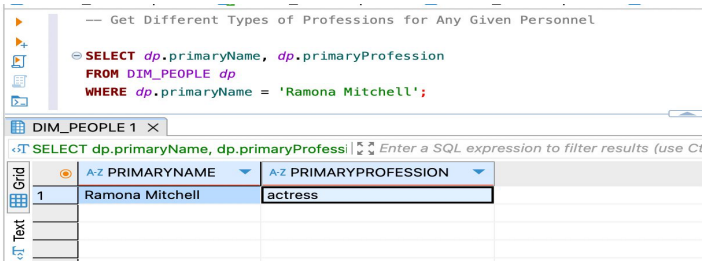
To enhance the geographical and linguistic analysis of the IMDb dataset, we integrated multiple external reference files and performed data mapping:

- One dataset contained Country Codes and Country Names.
- Another dataset included Language Codes and Language Names.
- A newly mapped Excel file was introduced, containing:
- Country Code and Country Name for global standardization.
- A third column listing all languages spoken in each country.
- A fourth column indicating the most spoken language in each country.

This integration enabled accurate country-language relationships, allowing for better localization insights, regional trend analysis, and multilingual content segmentation.

BUSINESS REQUIREMENTS:

1. Get different types of professions for any given personnel (Primary and other professions if applicable)



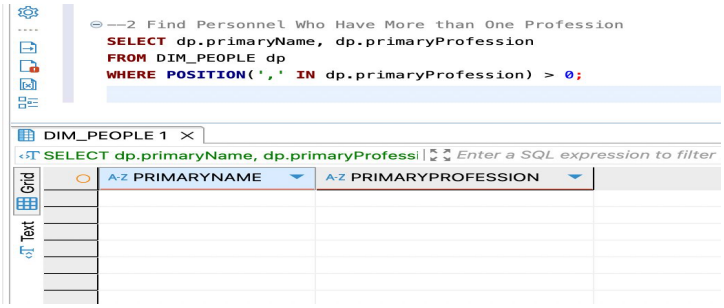
```
-- Get Different Types of Professions for Any Given Personnel

SELECT dp.primaryName, dp.primaryProfession
FROM DIM_PEOPLE dp
WHERE dp.primaryName = 'Ramona Mitchell';
```

DIM_PEOPLE 1

SELECT dp.primaryName, dp.primaryProfessi | Enter a SQL expression to filter results (use Ct

	PRIMARYNAME	PRIMARYPROFESSION
1	Ramona Mitchell	actress

2. Generate a report to find personnel who have more professions



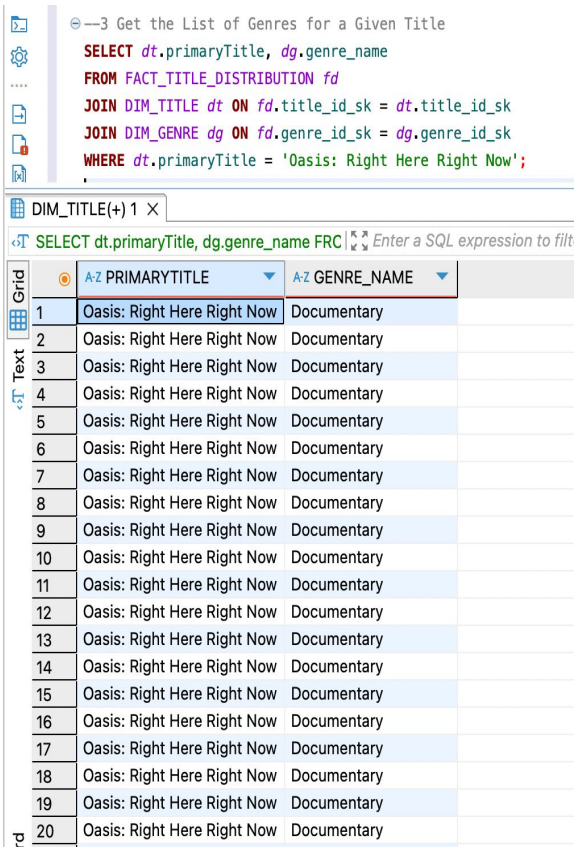
```
--2 Find Personnel Who Have More than One Profession
SELECT dp.primaryName, dp.primaryProfession
FROM DIM_PEOPLE dp
WHERE POSITION(',' IN dp.primaryProfession) > 0;
```

DIM_PEOPLE 1

SELECT dp.primaryName, dp.primaryProfessi | Enter a SQL expression to filter

PRIMARYNAME	PRIMARYPROFESSION

3. Get the list of genres for a given title and identify a list of movies based on a given genre



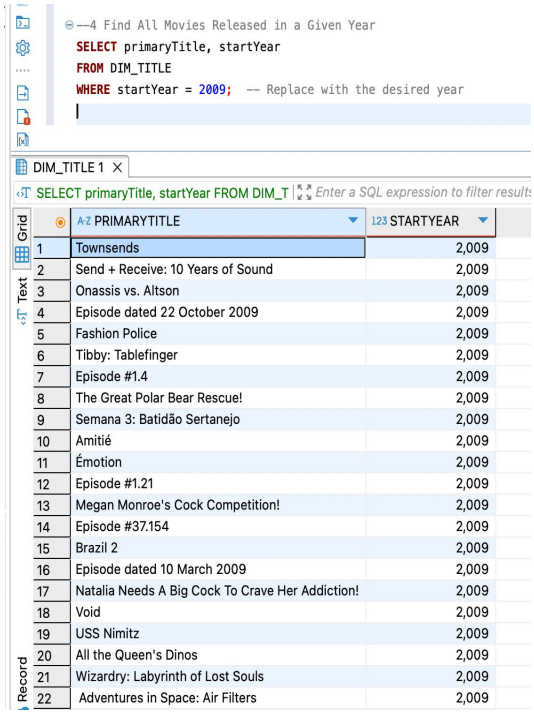
```
--3 Get the List of Genres for a Given Title
SELECT dt.primaryTitle, dg.genre_name
FROM FACT_TITLE_DISTRIBUTION fd
JOIN DIM_TITLE dt ON fd.title_id_sk = dt.title_id_sk
JOIN DIM_GENRE dg ON fd.genre_id_sk = dg.genre_id_sk
WHERE dt.primaryTitle = 'Oasis: Right Here Right Now';
```

DIM_TITLE(+) 1

SELECT dt.primaryTitle, dg.genre_name FRC | Enter a SQL expression to filt

	PRIMARYTITLE	GENRE_NAME
1	Oasis: Right Here Right Now	Documentary
2	Oasis: Right Here Right Now	Documentary
3	Oasis: Right Here Right Now	Documentary
4	Oasis: Right Here Right Now	Documentary
5	Oasis: Right Here Right Now	Documentary
6	Oasis: Right Here Right Now	Documentary
7	Oasis: Right Here Right Now	Documentary
8	Oasis: Right Here Right Now	Documentary
9	Oasis: Right Here Right Now	Documentary
10	Oasis: Right Here Right Now	Documentary
11	Oasis: Right Here Right Now	Documentary
12	Oasis: Right Here Right Now	Documentary
13	Oasis: Right Here Right Now	Documentary
14	Oasis: Right Here Right Now	Documentary
15	Oasis: Right Here Right Now	Documentary
16	Oasis: Right Here Right Now	Documentary
17	Oasis: Right Here Right Now	Documentary
18	Oasis: Right Here Right Now	Documentary
19	Oasis: Right Here Right Now	Documentary
20	Oasis: Right Here Right Now	Documentary

4. Find all moves that are released in a given year



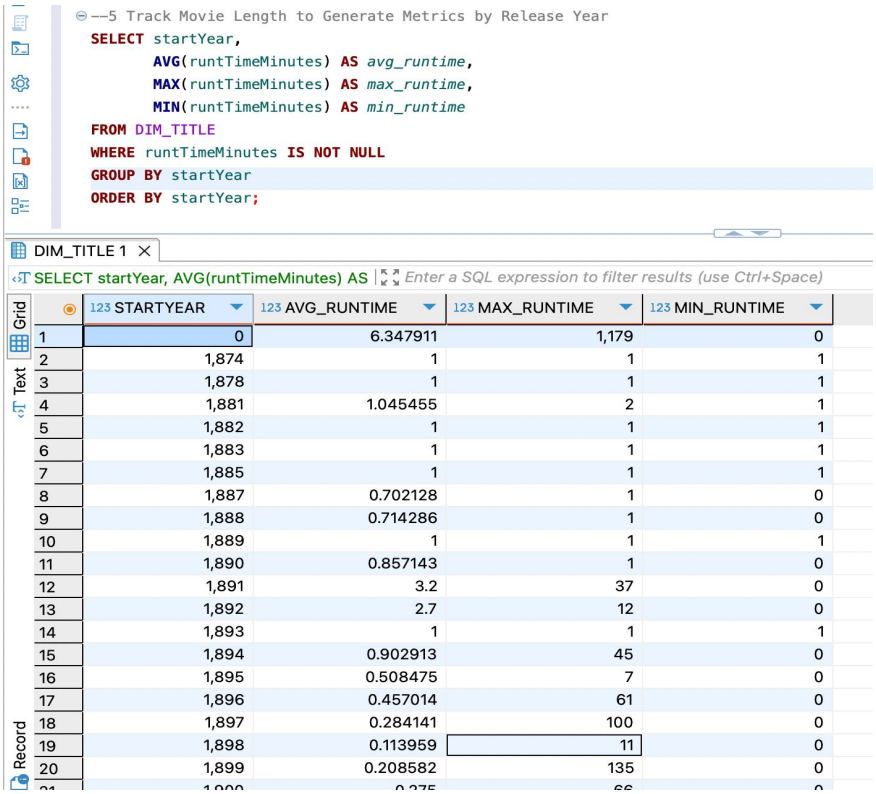
```sql
--4 Find All Movies Released in a Given Year
SELECT primaryTitle, startYear
FROM DIM_TITLE
WHERE startYear = 2009;  -- Replace with the desired year
```

DIM_TITLE 1

SELECT primaryTitle, startYear FROM DIM_T | Enter a SQL expression to filter results

	PRIMARYTITLE	STARTYEAR
1	Townsends	2,009
2	Send + Receive: 10 Years of Sound	2,009
3	Onassis vs. Altson	2,009
4	Episode dated 22 October 2009	2,009
5	Fashion Police	2,009
6	Tibby: Tablefinger	2,009
7	Episode #1.4	2,009
8	The Great Polar Bear Rescue!	2,009
9	Semana 3: Batidão Sertanejo	2,009
10	Amitié	2,009
11	Émotion	2,009
12	Episode #1.21	2,009
13	Megan Monroe's Cock Competition!	2,009
14	Episode #37.154	2,009
15	Brazil 2	2,009
16	Episode dated 10 March 2009	2,009
17	Natalia Needs A Big Cock To Crave Her Addiction!	2,009
18	Void	2,009
19	USS Nimitz	2,009
20	All the Queen's Dinos	2,009
21	Wizardry: Labyrinth of Lost Souls	2,009
22	Adventures in Space: Air Filters	2,009

5. Track the movie length to generate metrics on movie length based on release year.



```sql
--5 Track Movie Length to Generate Metrics by Release Year
SELECT startYear,
       AVG(runtimeMinutes) AS avg_runtime,
       MAX(runtimeMinutes) AS max_runtime,
       MIN(runtimeMinutes) AS min_runtime
FROM DIM_TITLE
WHERE runtimeMinutes IS NOT NULL
GROUP BY startYear
ORDER BY startYear;
```

DIM_TITLE 1

SELECT startYear, AVG(runtimeMinutes) AS | Enter a SQL expression to filter results (use Ctrl+Space)

	STARTYEAR	AVG_RUNTIME	MAX_RUNTIME	MIN_RUNTIME
1	0	6.347911	1,179	0
2	1,874	1	1	1
3	1,878	1	1	1
4	1,881	1.045455	2	1
5	1,882	1	1	1
6	1,883	1	1	1
7	1,885	1	1	1
8	1,887	0.702128	1	0
9	1,888	0.714286	1	0
10	1,889	1	1	1
11	1,890	0.857143	1	0
12	1,891	3.2	37	0
13	1,892	2.7	12	0
14	1,893	1	1	1
15	1,894	0.902913	45	0
16	1,895	0.508475	7	0
17	1,896	0.457014	61	0
18	1,897	0.284141	100	0
19	1,898	0.113959	11	0
20	1,899	0.208582	135	0

6. List all adult and non-adult movies

```sql
--6 List All Adult and Non-Adult Movies
SELECT primaryTitle, isAdult
FROM DIM_TITLE
WHERE isAdult = 1;
```

DIM_TITLE 1

SELECT primaryTitle, isAdult FROM DIM_TITL | Enter a SQL expression to filter results (use Ctrl+Space)

	PRIMARYTITLE	ISADULT
1	Young and Petite Slut in Grueling Bondage and Tormented	1
2	Morgan	1
3	Lydia Black Tormented In Brutal Restriction	1
4	Introducing Ai Okamoto	1
5	Chrissy Marie- The Audition	1
6	Cougar Sex Club	1
7	Jordan's POV Facials	1
8	Fucking Maria Marley's Soles	1
9	This Fine Woman Stared at Me with A Tempting Gaze and Hit Me with A Steady Stream of Ass-Shaking Filthy Dirty Talk Nanami Matsumoto	1
10	Great Tits On Army Brat Abby	1
11	Melissa and Hannah Blow our Fucking Minds	1
12	Zack Acland & Julian Torres	1
13	I've Got A Feeling	1

7. Generate all different languages in which a title is associated

```sql
--7 Generate All Different Languages in Which a Title is Associated
SELECT dt.primaryTitle, dl.language_name
FROM FACT_TITLE_DISTRIBUTION fd
JOIN DIM_TITLE dt ON fd.title_id_sk = dt.title_id_sk
JOIN DIM_LANGUAGES dl ON fd.language_id_sk = dl.language_id_sk
WHERE dt.primaryTitle = 'Morgan';
```

DIM_TITLE(+) 1

SELECT dt.primaryTitle, dl.language_name F | Enter a SQL expression to filter results (

	PRIMARYTITLE	LANGUAGE_NAME
1	Morgan	SL
2	Morgan	VI
3	Morgan	VG
4	Morgan	CH
5	Morgan	MO
6	Morgan	AE
7	Morgan	BUMM
8	Morgan	CW
9	Morgan	VC
10	Morgan	XWW
11	Morgan	HK
12	Morgan	SD
13	Morgan	DDDE
14	Morgan	SN
15	Morgan	XEU
16	Morgan	GP
17	Morgan	JE
18	Morgan	XKO
19	Morgan	ES

8. Get a list of regions in which a move is released

```
--8 Get a List of Regions in Which a Movie is Released
SELECT dt.primaryTitle, dr.region_name
FROM FACT_TITLE_DISTRIBUTION fd
JOIN DIM_TITLE dt ON fd.title_id_sk = dt.title_id_sk
JOIN DIM_REGION dr ON fd.region_id_sk = dr.region_id_sk
WHERE dt.primaryTitle = 'Morgan';
```

DIM_TITLE(+) 1

SELECT dt.primaryTitle, dr.region_name FRC | Enter a SQL expression to filter result

	PRIMARYTITLE	REGION_NAME
1	Morgan	Australia
2	Morgan	Australia
3	Morgan	Australia
4	Morgan	Australia
5	Morgan	Australia
6	Morgan	Australia
7	Morgan	Australia
8	Morgan	Australia
9	Morgan	Australia
10	Morgan	Australia
11	Morgan	Australia
12	Morgan	Australia
13	Morgan	Australia
14	Morgan	Australia
15	Morgan	Australia
16	Morgan	Australia
17	Morgan	Australia
18	Morgan	Australia
19	Morgan	Australia

9.Get the list of all directors and writers involved in making a movie so that I can identify the popular directors and writers

```
--9 Get a List of All Directors and Writers for a Movie
SELECT dt.primaryTitle, dp.primaryName, fc.category_id_sk
FROM FACT_TITLE_CAST_CREW fc
JOIN DIM_TITLE dt ON fc.title_id_sk = dt.title_id_sk
JOIN DIM_PEOPLE dp ON fc.person_id_sk = dp.person_id_sk
JOIN DIM_CATEGORY dc ON fc.category_id_sk = dc.category_id_sk
WHERE dc.category_name IN ('director', 'writer')
AND dt.primaryTitle = 'Reunion 2';
```

DIM_TITLE(+) 1

SELECT dt.primaryTitle, dp.primaryName, fc. | Enter a SQL expression to filter results (use Ctrl+Spac

	PRIMARYTITLE	PRIMARYNAME	CATEGORY_ID_SK
1	Reunion 2	Renato Bartilotti	11
2	Reunion 2	Mark Cronin	10
3	Reunion 2	Joyce Giraud	11
4	Reunion 2	Joyce Giraud	11
5	Reunion 2	Niki Iliev	10
6	Reunion 2	Niki Iliev	11
7	Reunion 2	Renato Bartilotti	11
8	Reunion 2	Mark Cronin	10
9	Reunion 2	Cris Abrego	10
10	Reunion 2	Cris Abrego	10
11	Reunion 2	Mark Cronin	10
12	Reunion 2	Niki Iliev	10
13	Reunion 2	Niki Iliev	11
14	Reunion 2	Renato Bartilotti	11
15	Reunion 2	Niki Iliev	10
16	Reunion 2	Niki Iliev	11
17	Reunion 2	Cris Abrego	10
18	Reunion 2	Niki Iliev	10
19	Reunion 2	Niki Iliev	11
20	Reunion 2	Joyce Giraud	11
21	Reunion 2	Tom Maguire	11

10.Get the number of episodes associated with a season for a given title

```sql
--10 Get the Number of Episodes Per Season for a Given Title
SELECT dt.primaryTitle AS Series_Name, fe.seasonNumber, COUNT(fe.episodeNumber) AS total_episodes
FROM FACT_EPISODES fe
JOIN DIM_TITLE dt ON fe.title_id_sk = dt.title_id_sk
WHERE dt.primaryTitle = 'Reunion 2'
GROUP BY dt.primaryTitle, fe.seasonNumber
ORDER BY fe.seasonNumber;
```

DIM_TITLE(+) 1

SELECT dt.primaryTitle AS Series_Name, fe.s | Enter a SQL expression to filter results (use Ctrl+Space)

	SERIES_NAME	SEASONNUMBER	TOTAL_EPISODES
1	Reunion 2	1	1
2	Reunion 2	2	1
3	Reunion 2	3	1

11. Find all different types of crew/cast, directors, writers, etc, for a given title

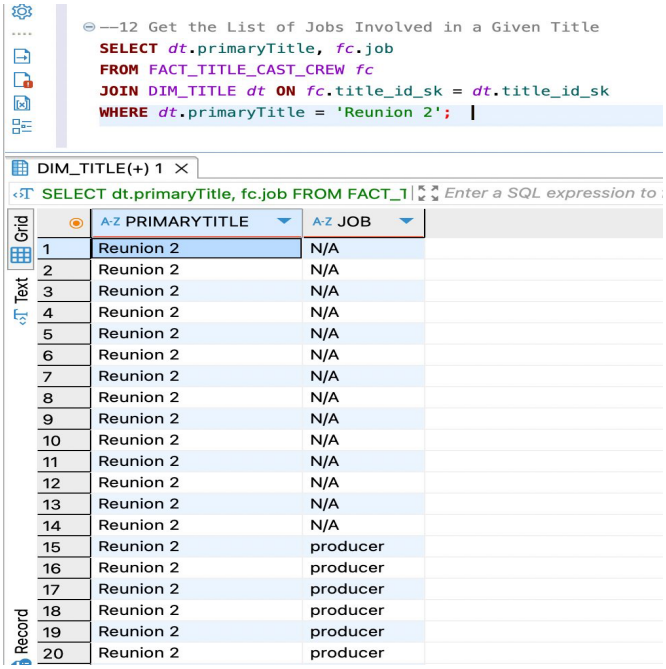
```sql
--11 Find All Different Types of Crew/Cast, Directors, Writers, etc., for a Given Title
SELECT dt.primaryTitle, dp.primaryName, dc.category_name
FROM FACT_TITLE_CAST_CREW fc
JOIN DIM_TITLE dt ON fc.title_id_sk = dt.title_id_sk
JOIN DIM_PEOPLE dp ON fc.person_id_sk = dp.person_id_sk
JOIN DIM_CATEGORY dc ON fc.category_id_sk = dc.category_id_sk
WHERE dt.primaryTitle = 'Reunion 2';
```

DIM_TITLE(+) 1

SELECT dt.primaryTitle, dp.primaryName, dc | Enter a SQL expression to filter results (use Ctrl+Space)

	PRIMARYTITLE	PRIMARYNAME	CATEGORY_NAME
1	Reunion 2	Scott Storey	production_designer
2	Reunion 2	Amy Johnson	self
3	Reunion 2	Lyubomir Iliev	editor
4	Reunion 2	Connie Arias	self
5	Reunion 2	Adam Ullberg	producer
6	Reunion 2	José Luis González	self
7	Reunion 2	José Luis González	self
8	Reunion 2	Niki Iliev	producer
9	Reunion 2	Niki Iliev	writer
10	Reunion 2	Niki Iliev	director
11	Reunion 2	Niki Iliev	actor
12	Reunion 2	Tiaunte Kelly	actor
13	Reunion 2	Lisa Lopez	producer
14	Reunion 2	Laurent Basset	cinematographer
15	Reunion 2	Renato Bartilotti	director
16	Reunion 2	Mark Cronin	writer
17	Reunion 2	Dilyana Popova	actress
18	Reunion 2	Adriana Gallardo	self
19	Reunion 2	Mark Cronin	writer
20	Reunion 2	Wes Denton	editor
21	Reunion 2	Cameron Teisher	editor

12. Get the list of jobs involved in a given title



```
--12 Get the List of Jobs Involved in a Given Title
SELECT dt.primaryTitle, fc.job
FROM FACT_TITLE_CAST_CREW fc
JOIN DIM_TITLE dt ON fc.title_id_sk = dt.title_id_sk
WHERE dt.primaryTitle = 'Reunion 2';
```

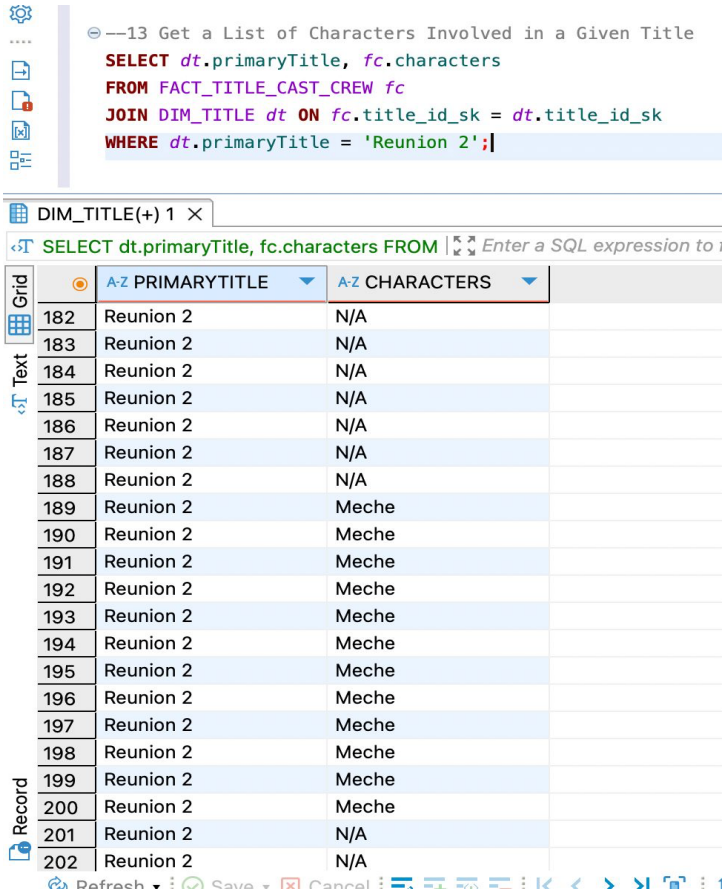
DIM_TITLE(+) 1

SELECT dt.primaryTitle, fc.job FROM FACT_T | Enter a SQL expression to

Grid | Text | Record

	PRIMARYTITLE	JOB
1	Reunion 2	N/A
2	Reunion 2	N/A
3	Reunion 2	N/A
4	Reunion 2	N/A
5	Reunion 2	N/A
6	Reunion 2	N/A
7	Reunion 2	N/A
8	Reunion 2	N/A
9	Reunion 2	N/A
10	Reunion 2	N/A
11	Reunion 2	N/A
12	Reunion 2	N/A
13	Reunion 2	N/A
14	Reunion 2	N/A
15	Reunion 2	producer
16	Reunion 2	producer
17	Reunion 2	producer
18	Reunion 2	producer
19	Reunion 2	producer
20	Reunion 2	producer

13. Get a list of characters involved in a given title



```
--13 Get a List of Characters Involved in a Given Title
SELECT dt.primaryTitle, fc.characters
FROM FACT_TITLE_CAST_CREW fc
JOIN DIM_TITLE dt ON fc.title_id_sk = dt.title_id_sk
WHERE dt.primaryTitle = 'Reunion 2';
```

DIM_TITLE(+) 1

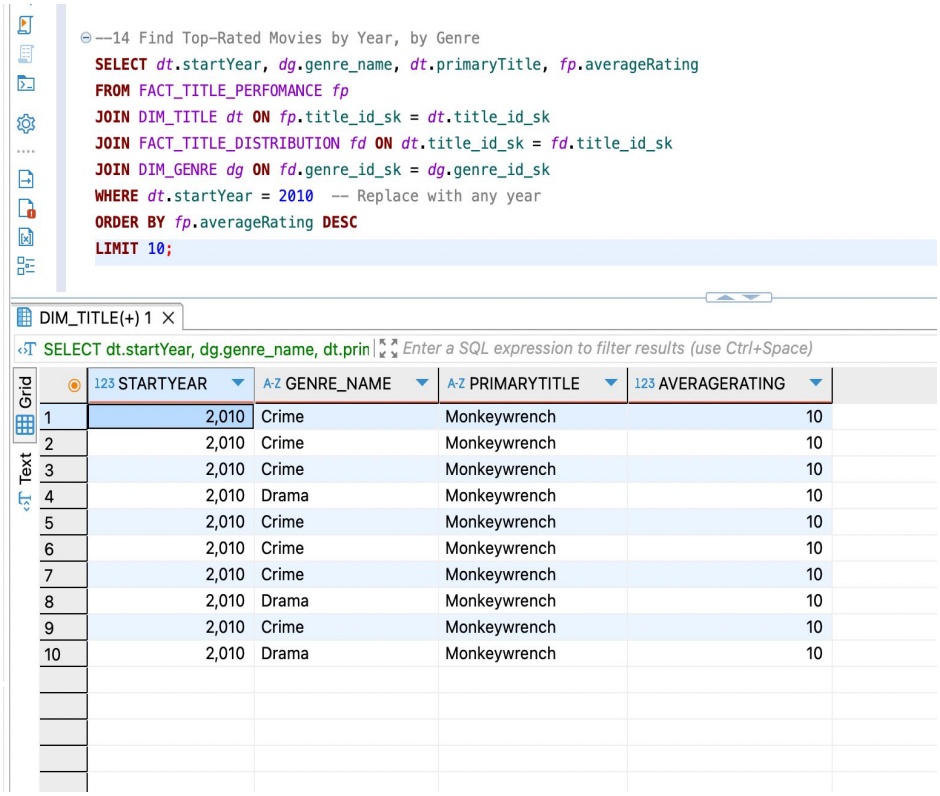
SELECT dt.primaryTitle, fc.characters FROM | Enter a SQL expression to

Grid | Text | Record

	PRIMARYTITLE	CHARACTERS
182	Reunion 2	N/A
183	Reunion 2	N/A
184	Reunion 2	N/A
185	Reunion 2	N/A
186	Reunion 2	N/A
187	Reunion 2	N/A
188	Reunion 2	N/A
189	Reunion 2	Meche
190	Reunion 2	Meche
191	Reunion 2	Meche
192	Reunion 2	Meche
193	Reunion 2	Meche
194	Reunion 2	Meche
195	Reunion 2	Meche
196	Reunion 2	Meche
197	Reunion 2	Meche
198	Reunion 2	Meche
199	Reunion 2	Meche
200	Reunion 2	Meche
201	Reunion 2	N/A
202	Reunion 2	N/A

Refresh | Save | Cancel

14. Find top-rated movies by year, by genre



```sql
--14 Find Top-Rated Movies by Year, by Genre
SELECT dt.startYear, dg.genre_name, dt.primaryTitle, fp.averageRating
FROM FACT_TITLE_PERFOMANCE fp
JOIN DIM_TITLE dt ON fp.title_id_sk = dt.title_id_sk
JOIN FACT_TITLE_DISTRIBUTION fd ON dt.title_id_sk = fd.title_id_sk
JOIN DIM_GENRE dg ON fd.genre_id_sk = dg.genre_id_sk
WHERE dt.startYear = 2010  -- Replace with any year
ORDER BY fp.averageRating DESC
LIMIT 10;
```

DIM_TITLE(+) 1

SELECT dt.startYear, dg.genre_name, dt.prin | Enter a SQL expression to filter results (use Ctrl+Space)

Grid | Text

	123 STARTYEAR	A-Z GENRE_NAME	A-Z PRIMARYTITLE	123 AVERAGERATING
1	2,010	Crime	Monkeywrench	10
2	2,010	Crime	Monkeywrench	10
3	2,010	Crime	Monkeywrench	10
4	2,010	Drama	Monkeywrench	10
5	2,010	Crime	Monkeywrench	10
6	2,010	Crime	Monkeywrench	10
7	2,010	Crime	Monkeywrench	10
8	2,010	Drama	Monkeywrench	10
9	2,010	Crime	Monkeywrench	10
10	2,010	Drama	Monkeywrench	10

SNOWFLAKE SCRIPT:

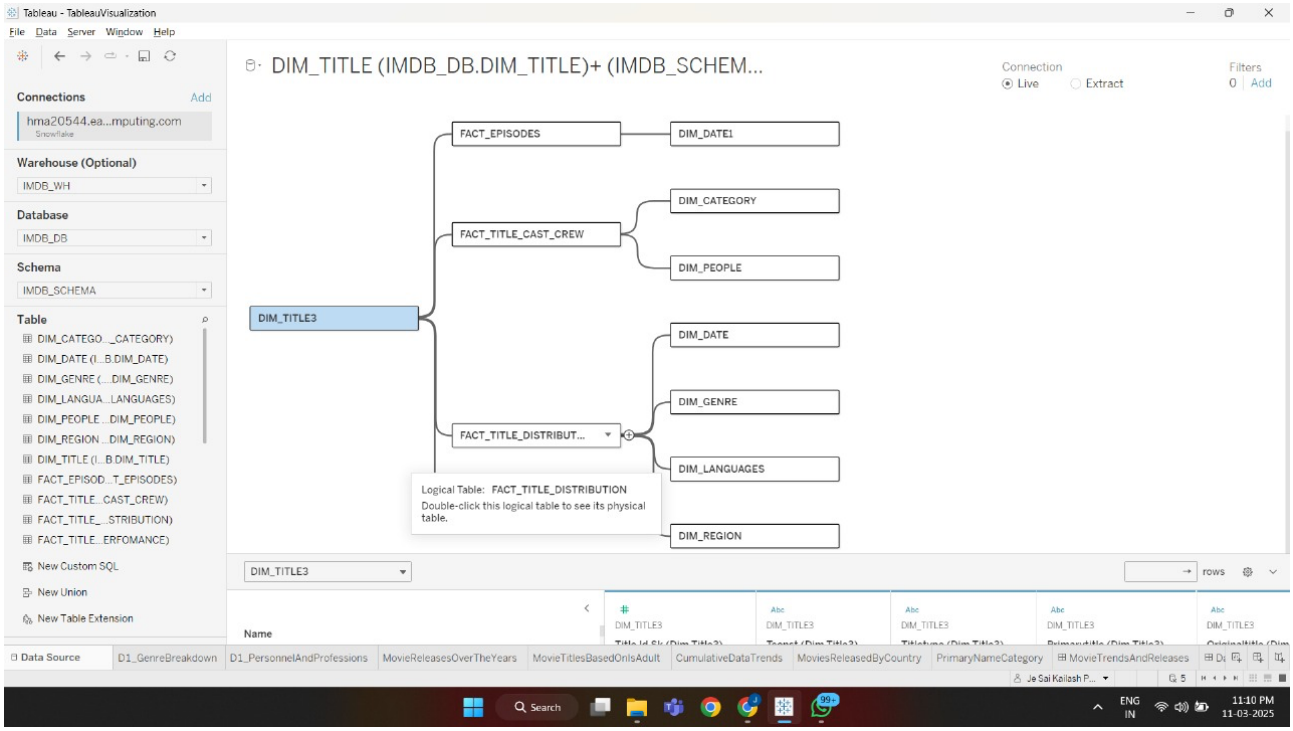
```
-- To prove you logged in as an admin account on your Snowflake
account
select current_date,
current_warehouse(),
current_database(),
current_Schema(),
current_role(),
current_user();

create or replace warehouse IMDB_WH warehouse_size= 'XSMALL';

create or replace database IMDB_DB;

create or replace schema IMDB_DB.IMDB_SCHEMA;

CREATE OR replace role IMDB_ROLE;

grant ownership on database IMDB_DB to role ACCOUNTADMIN;
```

```
GRANT USAGE ON WAREHOUSE IMDB_WH TO ROLE IMDB_ROLE;

grant usage on database IMDB_DB to role IMDB_ROLE;

--Schema
grant ownership on schema IMDB_DB.IMDB_SCHEMA to role IMDB_ROLE;
grant usage on schema IMDB_DB.IMDB_SCHEMA to role IMDB_ROLE;

--Creating a user and assigning warehouse and rote
create or replace user IMDB_USER password= 'snowflake123#'
default_role=IMDB_ROLE default_warehouse=IMDB_WH;

grant role IMDB_ROLE to user IMDB_USER;
-- grant warehouse IP_NYPD_WH to user IP_NYPD_USER;

GRANT SELECT ON ALL TABLES IN SCHEMA IMDB_DB.IMDB_SCHEMA TO ROLE
ACCOUNTADMIN;

USE SCHEMA IMDB_SCHEMA;

GRANT USAGE ON SCHEMA IMDB_DB.IMDB_SCHEMA TO ROLE IMDB_ROLE;
-- GRANT SELECT, INSERT, UPDATE, DELETE ON
IP_NYPD_DB.IP_NYPD_SCHEMA.Fact_Arrests TO ROLE TEMP_ROLE;

GRANT USAGE ON SCHEMA IMDB_DB.IMDB_SCHEMA TO ROLE ACCOUNTADMIN;

SHOW GRANTS ON SCHEMA IMDB_SCHEMA;

-- select count(*) from FACT_ARRESTS;

-- GRANT SELECT ON TABLE IMDB_DB.IP_NYPD_SCHEMA.FACT_ARRESTS TO
ROLE IP_NYPD_ROLE;

-- GRANT USAGE ON SCHEMA IMDB_DB.IP_NYPD_SCHEMA TO ROLE
IP_NYPD_ROLE;
-- GRANT SELECT ON ALL TABLES IN SCHEMA IMDB_DB.IP_NYPD_SCHEMA TO
ROLE IP_NYPD_ROLE;

SHOW GRANTS ROLE IMDB_ROLE;

SELECT CURRENT_ROLE();

-- SHOW GRANTS ON TABLE IP_NYPD_DB.IP_NYPD_SCHEMA.FACT_ARRESTS;

GRANT USAGE ON DATABASE IMDB_DB TO ROLE IMDB_ROLE;
```

```
GRANT USAGE ON SCHEMA IMDB_DB.IMDB_SCHEMA TO ROLE IMDB_ROLE;

-- SELECT * FROM IP_NYPD_DB.IP_NYPD_SCHEMA.FACT_ARRESTS LIMIT 10;


SHOW GRANTS TO USER IMDB_USER;
GRANT ROLE IP_NYPD_ROLE TO USER IP_NYPD_USER;
-- USE ROLE IP_NYPD_ROLE;
-- SHOW USERS LIKE 'IP_NYPD_USER';
GRANT ROLE IMDB_ROLE TO ROLE SYSADMIN;

SHOW GRANTS ON TABLE IMDB_DB.IMDB_SCHEMA;

GRANT USAGE ON SCHEMA IMDB_SCHEMA TO IMDB_ROLE;
GRANT SELECT ON ALL TABLES IN SCHEMA IMDB_SCHEMA TO IMDB_ROLE;
```

VISUALIZATIONS:

Created Tableau Visualizations. Initially, we performed Tableau data loading

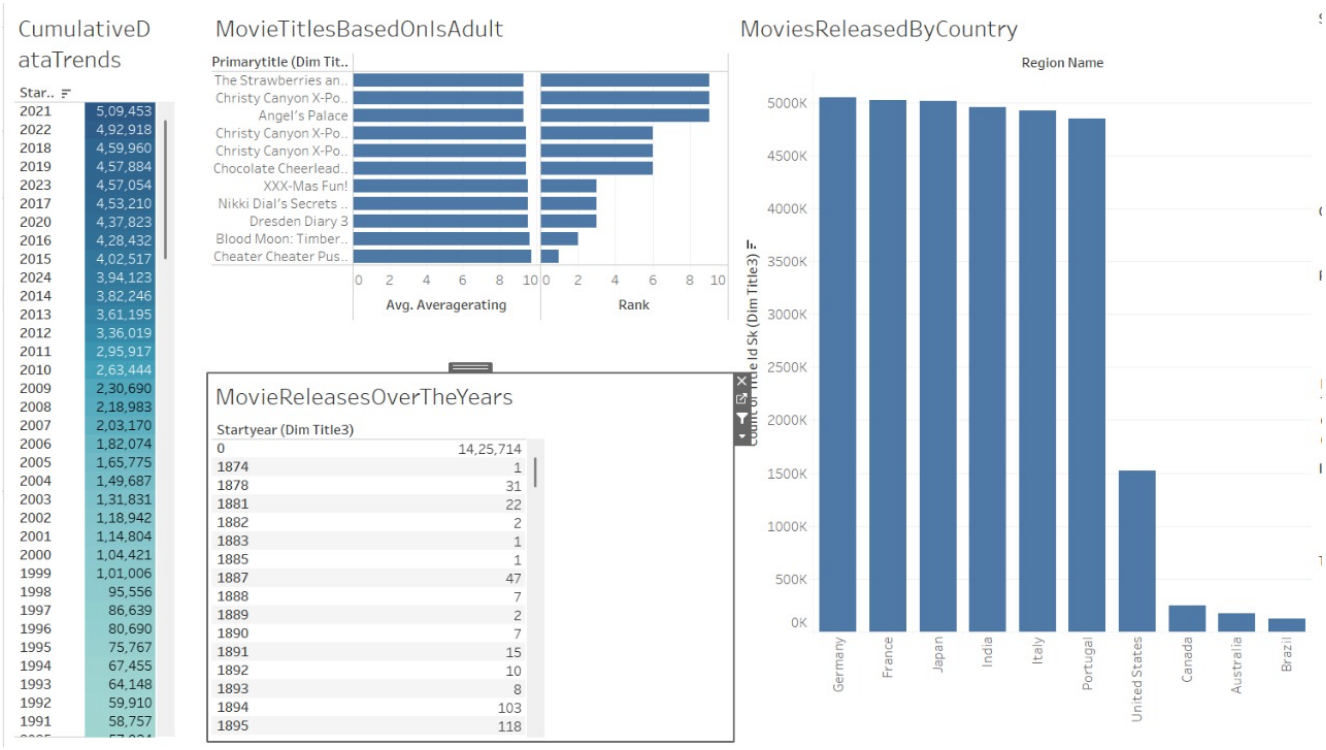


Then , Performed various visualizations and combined them into 2 dashboards

Dashboard 1:
Movie Trend and Releases

- Movie Releases Over the Years → (Trend Analysis)
- Cumulative Data Trends → (Line/Area Chart)
- No of Movies Released by Country → (Map or Bar Chart)

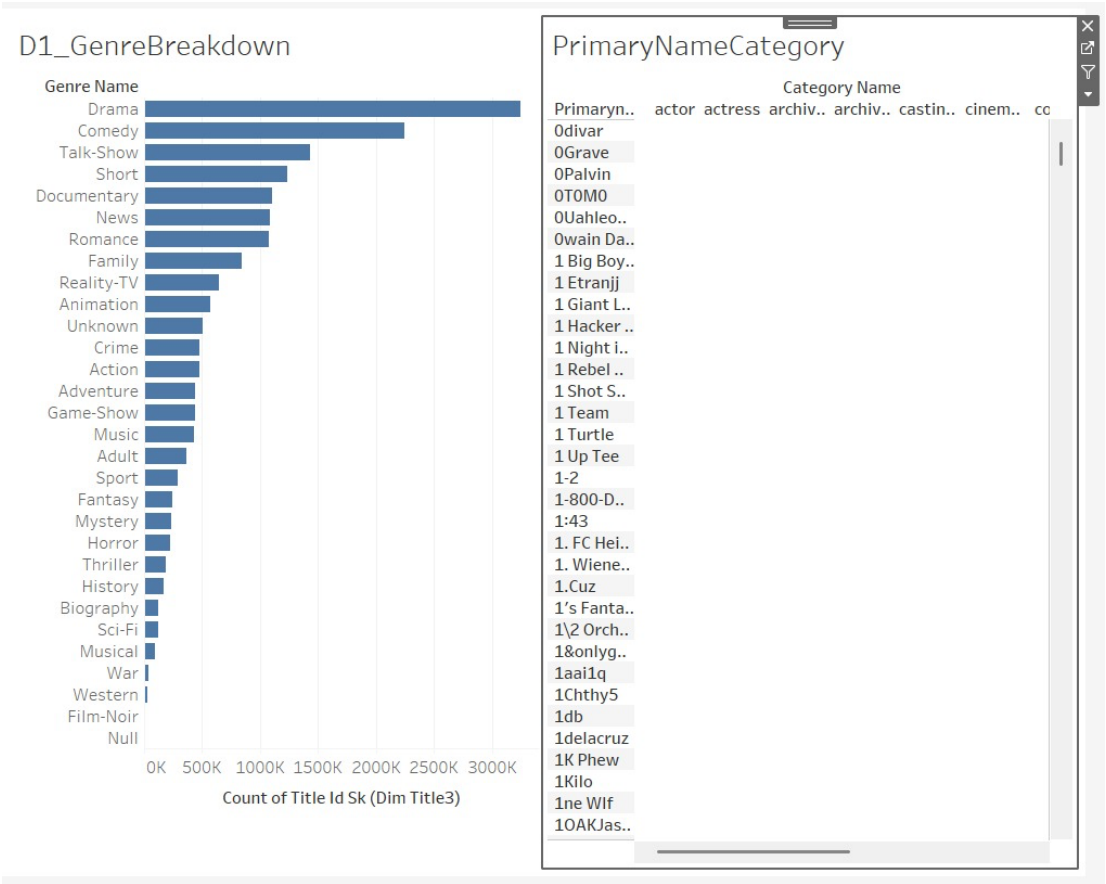
- Movie Titles Based on IsAdult → (Pie or Bar Chart)



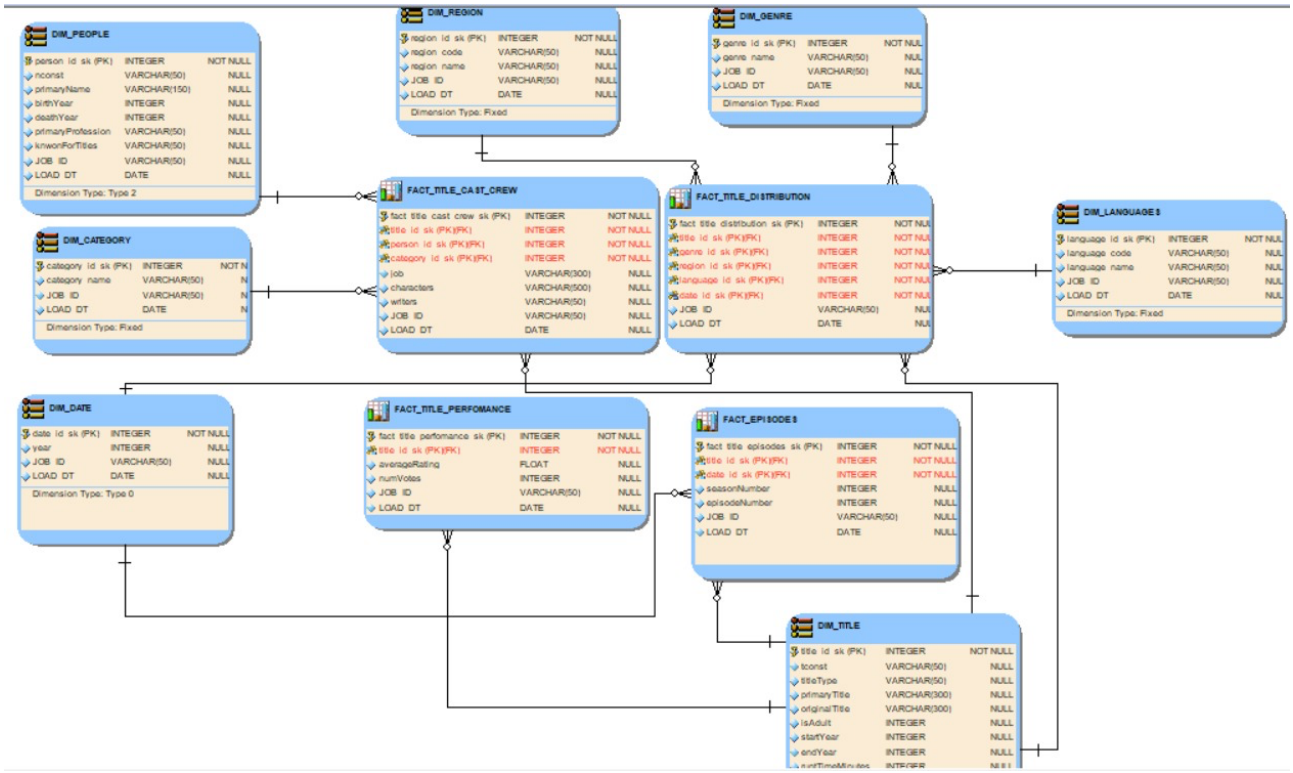
Dashboard 2:

Genre and People Analysis:

- Genre Breakdown → (Bar Chart / Treemap)
- Primary Name & Category → (People & Job Roles)



DATA MODELLING



GITHUB REPOSITORY LINK:

https://github.com/pjsk02/DAMG7370_DADABI_MidTerm