Data Report - IMDb Dataset

Team
Ishaan Samel (002301229)
Shruti Sen (002057639)
Ayush Fulsundar (002312108)

Github repo link - https://github.com/shrutisen/DAMG7370_Midterm

Data Overview

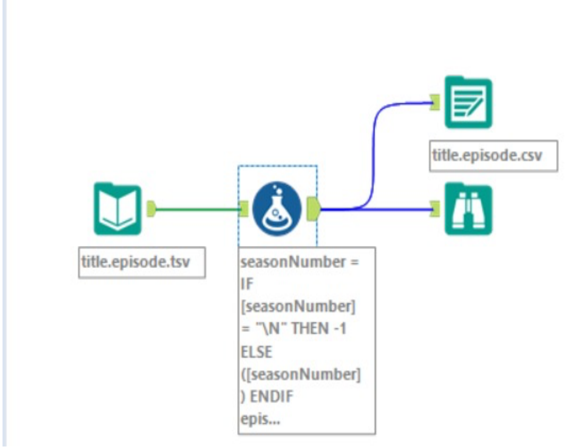
1. title.akas.tsv.gz - 8 fields
2. title.basics.tsv.gz - 9 fields
3. title.crew.tsv.gz - 3 fields
4. title.episode.tsv.gz - 4 fields
5. title.principals.tsv.gz - 6 fields
6. title.ratings.tsv.gz - 3 fields
7. name.basics.tsv.gz - 6 fields
8. Language.csv - 2 fields
9. Regions.csv - 2 fields

Data Handling -

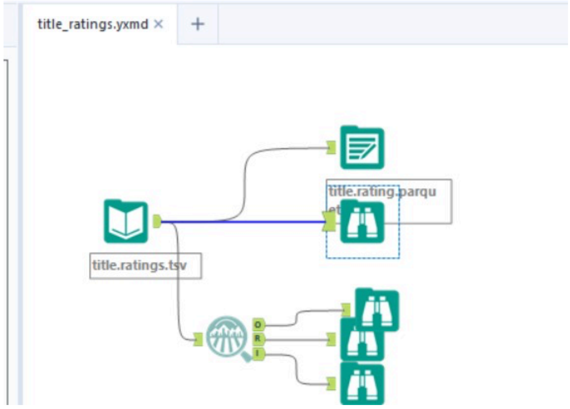
The dataset was cleaned and preprocessed using Python and Alteryx to ensure data quality and consistency. Missing values were handled systematically by replacing NULLs with appropriate placeholders such as `"Unknown"`, `"NA"`, `"Missing"`, or `"9999"` based on the column type. Additionally, extra spaces in text fields were removed to maintain data integrity. This preprocessing step enhances data reliability for further analysis, ensuring that the dataset is ready for downstream processing and insights extraction.
We used both Alteryx and Python for the data pre-processing.

Column Category	Observation	Cleaning Approach
Title-related Columns *(e.g., primaryTitle, originalTitle, genres)*	Some records have NULL values.	Replace NULLs with "Unknown".
Boolean Columns *(e.g., isAdult)*	Some records have NULL values.	Replace NULLs with "NA".
Year-related Columns *(e.g., startYear, endYear, birthYear, deathYear)*	Some records have NULL values.	Replace NULLs (start/birth) and (end/death) with "9999".
Duration Column *(e.g., runtimeMinutes)*	Some records have NULL values.	Replace NULLs with "00".
Profession & Known For Columns *(e.g., primaryProfession, knownForTitles)*	Some records have NULL values.	Replace NULLs with "Unknown" or "Missing".
All text columns	Some records contain extra spaces.	Trim extra spaces from text fields.

Data Profiling and Cleaning -



```
seasonNumber =
IF
[seasonNumber]
= "\N" THEN -1
ELSE
([seasonNumber]
) ENDIF
epis...
```



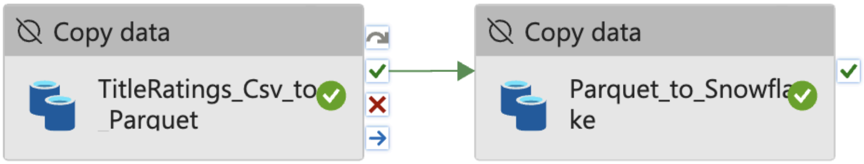
End-to-End Data Processing Pipeline using Azure & Snowflake

Step	**Description**
1. Data Ingestion (Medallion Architecture)	The raw data is loaded into the Bronze container in Azure Blob Storage. The data is then staged into Parquet format in the Silver container to ensure optimized storage and querying. Finally, the processed data from Silver is staged and loaded into Snowflake for further transformations and reporting.

2. Data Preprocessing (Handling Comma-Separated Values)	Certain columns contained comma-separated values (CSV format), which were split and flattened before being loaded into the stage tables. This ensured that each value was stored in a structured format, allowing for efficient querying and transformation.
3. Data Modeling (ER Studio)	The data model was designed using ER Studio, following a dimensional modeling approach. The schema includes 2 Fact tables, 8 Dimension tables, and 3 Bridge tables to ensure flexibility and efficient querying.
4. Data Loading (Azure Data Flows - ADFs)	We utilized Azure Data Flows (ADFs) to extract data from stage tables, transforming and loading it into Fact, Dimension, and Bridge tables in Snowflake. This ensures data is structured correctly for analytics and reporting.
5. Data Processing and Key Handling	Within ADFs, we implemented inner joins where necessary to maintain data integrity. Additionally, we generated surrogate keys, created derived columns, and applied other required transformations to standardize the dataset before final loading.

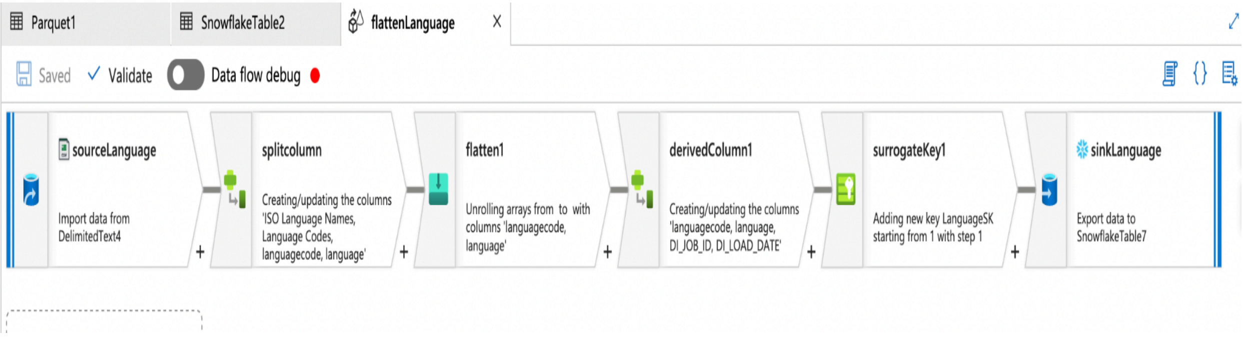
Data Ingestion (Medallion Architecture)

By leveraging **Azure Data Lake** storage capabilities, we ensured data was optimized for both storage efficiency and processing speed in the Bronze container. Once the cleaned and structured data was available in Silver, it was staged and finally loaded into **Snowflake**, where further transformations, aggregations, and analytics were applied. The structured Snowflake tables enabled seamless integration with downstream BI tools for reporting and visualization.



Data Preprocessing (Handling Comma-Separated Values)

This transformation ensured that each value was stored separately in the stage tables, improving data normalization and query efficiency. Additionally, we implemented data type standardization and handled nested values to maintain consistency across tables. The cleaned and structured data was then staged in Azure Data Lake, making it easier to integrate into Snowflake. This approach allowed us to maintain the relational integrity of the dataset while making multi-value fields easier to process in downstream transformations.



Import data from DelimitedText4

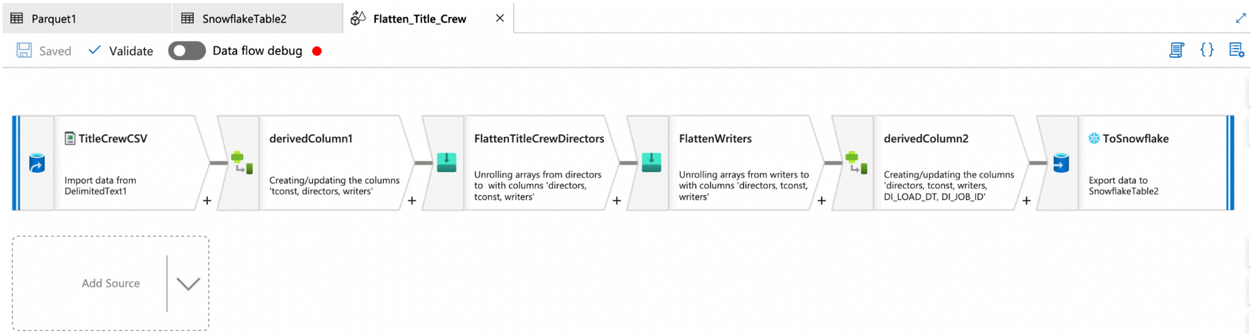
Creating/updating the columns 'ISO Language Names, Language Codes, languagecode, language'

Unrolling arrays from to with columns 'languagecode, language'

Creating/updating the columns 'languagecode, language, DI_JOB_ID, DI_LOAD_DATE'

Adding new key LanguageSK starting from 1 with step 1

Export data to SnowflakeTable7



Import data from DelimitedText1

Creating/updating the columns 'tconst, directors, writers'

Unrolling arrays from directors to with columns 'directors, tconst, writers'

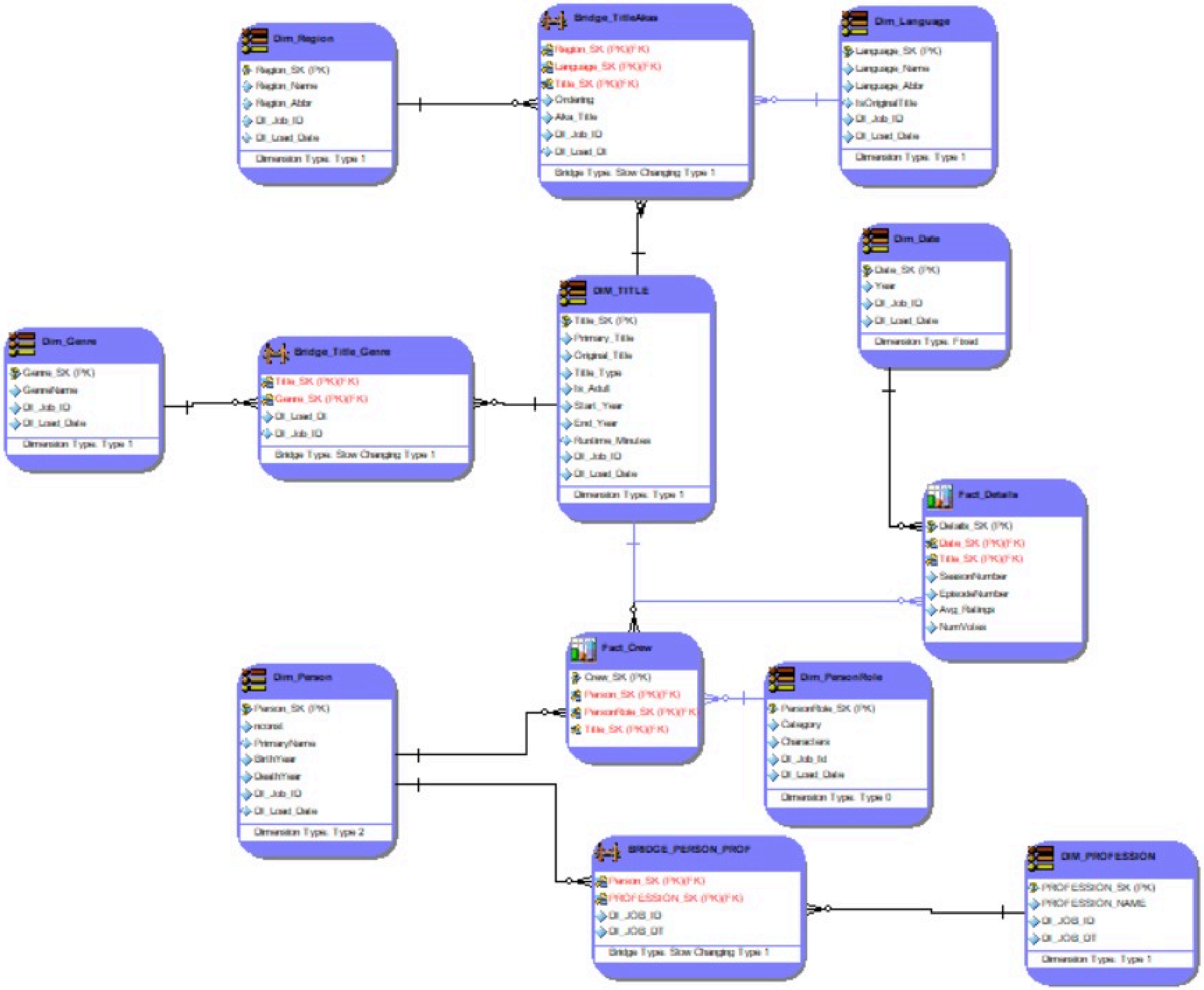
Unrolling arrays from writers to with columns 'directors, tconst, writers'

Creating/updating the columns 'directors, tconst, writers, DI_LOAD_DT, DI_JOB_ID'

Export data to SnowflakeTable2

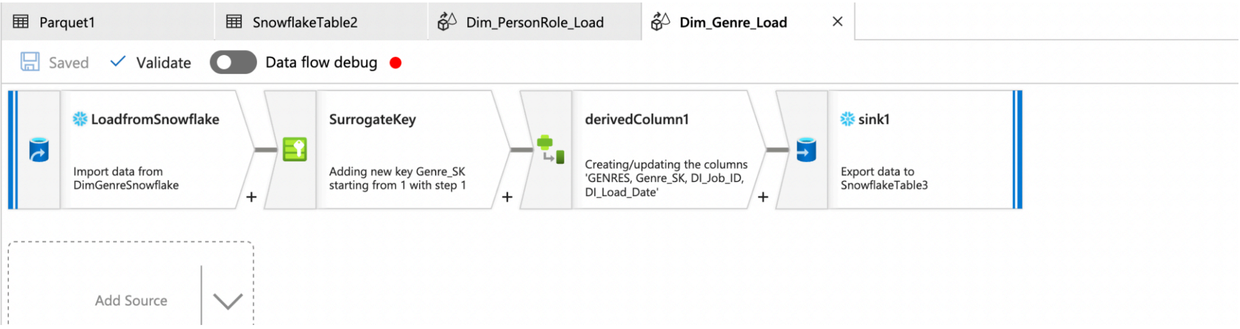
Data Modeling (ER Studio)

The Fact tables stored the core transactional data, while the Dimension tables contained descriptive attributes, making it easier to analyze trends and patterns. Bridge tables were introduced to handle complex relationships between entities, such as handling multi-valued attributes. Additionally, we carefully defined foreign key relationships to maintain referential integrity and improve data retrieval speed.



Data Transformation (Azure Data Flows - ADFs)

Surrogate keys were created to provide unique identifiers for records, helping maintain data consistency across multiple tables. Derived columns were generated based on business logic, enabling additional computed attributes to be stored for advanced analysis. Date and time-based transformations were also applied to standardize formats and allow seamless time-based aggregations. These processing steps helped ensure that the final dataset was structured, clean, and ready for analytics and reporting.

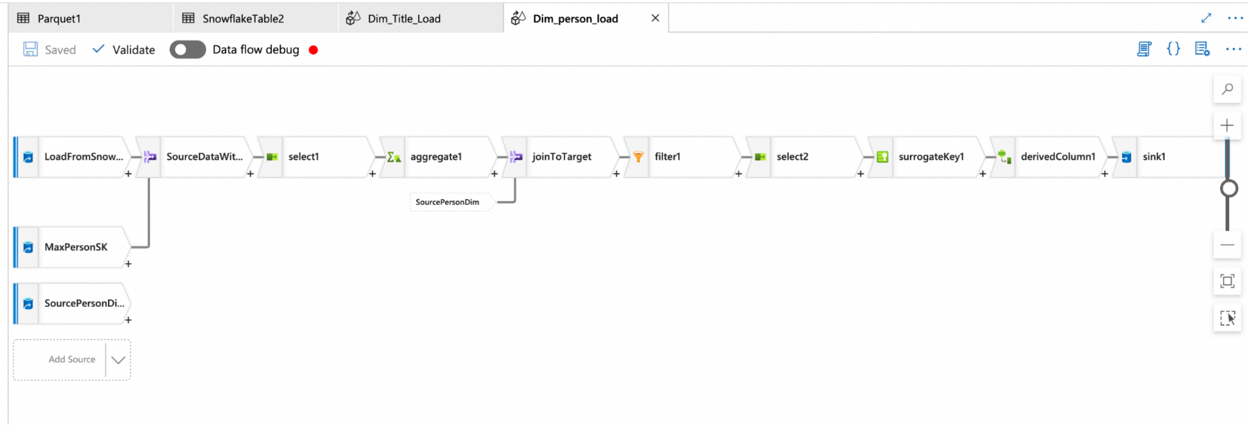


Import data from DimGenreSnowflake

Adding new key Genre_SK starting from 1 with step 1

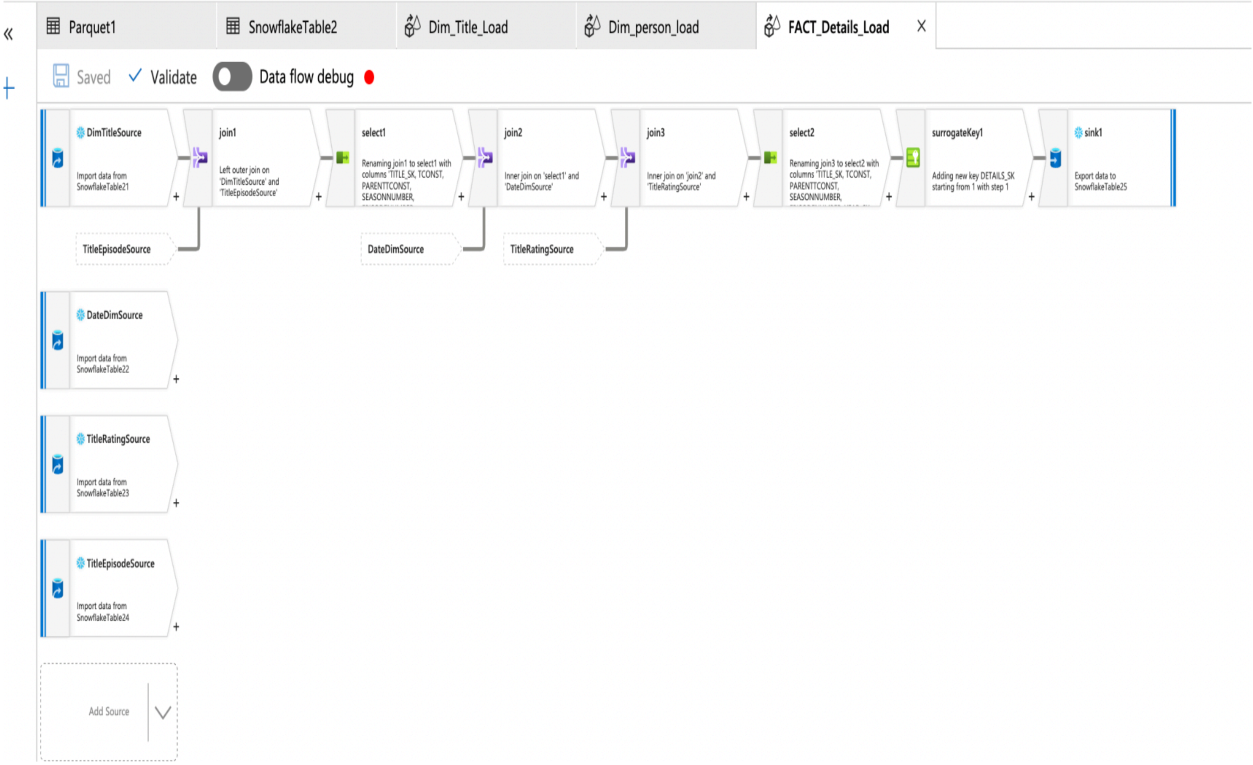
Creating/updating the columns 'GENRES, Genre_SK, DI_Job_ID, DI_Load_Date'

Export data to SnowflakeTable3



Data Loading (Dimensions & Facts)

Incremental loading strategies were implemented to optimize processing time and avoid redundant data ingestion. ADF pipelines were also designed to handle large-scale data processing while ensuring data consistency and integrity.



DimTitleSource
Import data from SnowflakeTable21

join1
Left outer join on 'DimTitleSource' and 'TitleEpisodeSource'

select1
Renaming join1 to select1 with columns 'TITLE_SK, TCONST, PARENTTCONST, SEASONNUMBER,

join2
Inner join on 'select1' and 'DateDimSource'

join3
Inner join on 'join2' and 'TitleRatingSource'

select2
Renaming join3 to select2 with columns 'TITLE_SK, TCONST, PARENTTCONST, SEASONNUMBER,

surrogateKey1
Adding new key DETAILS_SK starting from 1 with step 1

sink1
Export data to SnowflakeTable25

TitleEpisodeSource

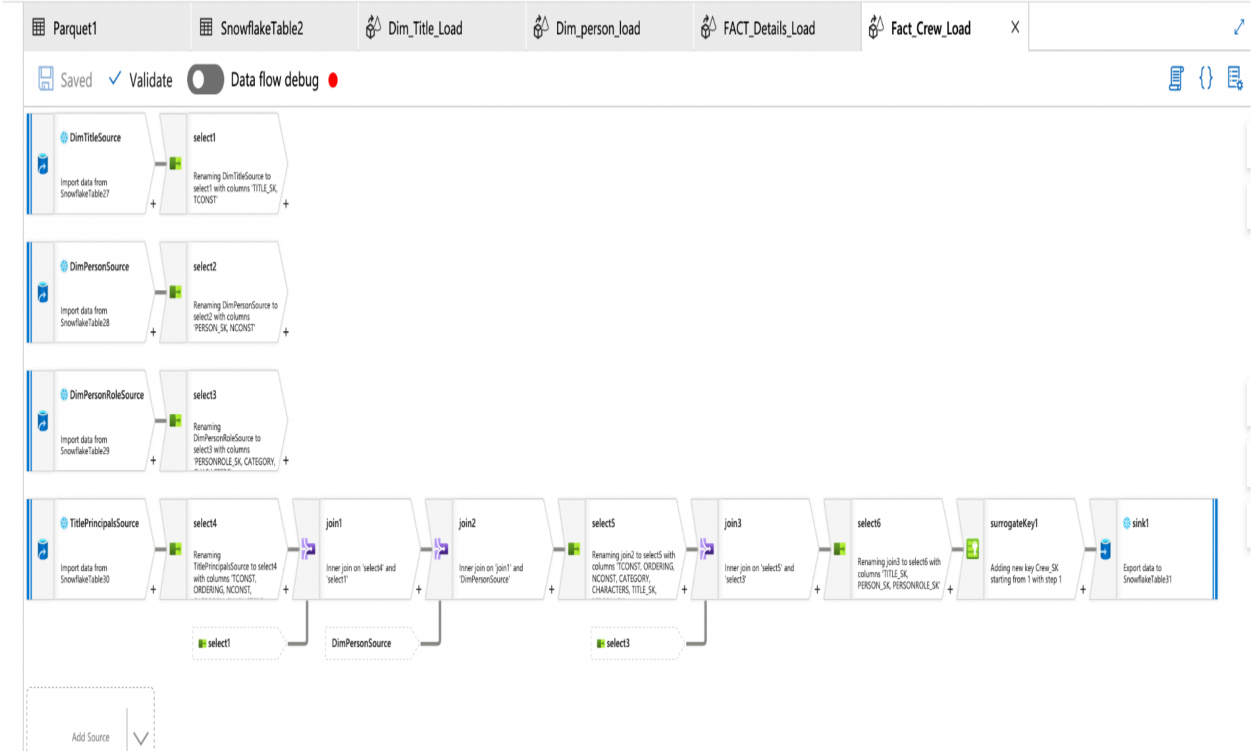
DateDimSource

TitleRatingSource

DateDimSource
Import data from SnowflakeTable22

TitleRatingSource
Import data from SnowflakeTable23

TitleEpisodeSource
Import data from SnowflakeTable24



DimTitleSource
Import data from SnowflakeTable27

select1
Renaming DimTitleSource to select1 with columns 'TITLE_SK, TCONST'

DimPersonSource
Import data from SnowflakeTable28

select2
Renaming DimPersonSource to select2 with columns 'PERSON_SK, NCONST'

DimPersonRoleSource
Import data from SnowflakeTable29

select3
Renaming DimPersonRoleSource to select3 with columns 'PERSONROLE_SK, CATEGORY,

TitlePrincipalsSource
Import data from SnowflakeTable30

select4
Renaming TitlePrincipalsSource to select4 with columns 'TCONST, ORDERING, NCONST,

join1
Inner join on 'select4' and 'select1'

join2
Inner join on 'join1' and 'DimPersonSource'

select5
Renaming join2 to select5 with columns 'TCONST, ORDERING, NCONST, CATEGORY, CHARACTERS, TITLE_SK,

join3
Inner join on 'select5' and 'select3'

select6
Renaming join3 to select6 with columns 'TITLE_SK, PERSON_SK, PERSONROLE_SK'

surrogateKey1
Adding new key Crew_SK starting from 1 with step 1

sink1
Export data to SnowflakeTable31

select1

DimPersonSource

select3