StreamFlix: A Business Intelligence Architecture for Streaming Analytics

Jesús Betancourt
CISeD - Research Center in Digital Services
Polytechnic of Viseu
Viseu, Portugal
pv22987@alunos.estgv.ipv.pt

Leandro Dias
CISeD - Research Center in Digital Services
Polytechnic of Viseu
Viseu, Portugal
pv23028@alunos.estgv.ipv.pt

Miguel Batista
CISeD - Research Center in Digital Services
Polytechnic of Viseu
Viseu, Portugal
pv22976@alunos.estgv.ipv.pt

Rodrigo Correia
CISeD - Research Center in Digital Services
Polytechnic of Viseu
Viseu, Portugal
pv23006@alunos.estgv.ipv.pt

***Abstract*—The StreamFlix project presents a comprehensive Business Intelligence (BI) architecture tailored to the streaming media domain. By simulating a real-world platform through synthetic data generation, the system integrates four heterogeneous operational sources: CSV, MySQL, and two PostgreSQL instances into a centralized analytical environment. A custom Extract–Transform–Load (ETL) pipeline in Python supports change tracking via is "up_to_date" flags and Slowly Changing Dimensions (SCD), with Type 2.3 logic ensuring historical traceability. Transformed data is modeled in a star schema hosted on Microsoft SQL Server and exposed via an OLAP cube built in SQL Server Analysis Services (SSAS). Two Power BI dashboards provide multidimensional insights with interactive features such as drill-down, slicing, and user-level analysis. The project demonstrates scalable data integration, robust temporal modeling, and actionable reporting, serving as a template for BI solutions in dynamic content ecosystems.**

Keywords—business intelligence, etl, olap, slowly changing dimensions, streaming analytics, dimensional modeling, power bi

I. Introduction

The evolution of streaming media platforms has generated enormous volumes of heterogeneous data, encompassing user behavior, content metadata, device information, and temporal attributes. Effective analysis of this data is essential for informing strategic decisions, optimizing content delivery, and enhancing user engagement. However, achieving a unified, high-quality view of streaming operations poses significant challenges due to the diversity of source systems, the need to preserve historical records, and the requirement for rapid, interactive exploration of aggregated metrics. A robust Business Intelligence (BI) architecture is therefore critical to transform raw, disparate inputs into actionable insights.

In this context, a simulated streaming environment was developed to demonstrate a full BI pipeline, addressing the complexities associated with multi-source integration. Data is generated programmatically via Python scripts and persist in four distinct systems: a CSV file recording session logs, a MySQL instance containing content metadata, a primary PostgreSQL instance managing user profiles, and a secondary PostgreSQL instance replicating an autonomous operational dataset. Each of these environments exposes its own data formats, naming conventions, and update patterns, necessitating a flexible extraction, transformation, and loading (ETL) strategy that can reconcile schema discrepancies and enforce data consistency across sources. To automate database provisioning and data import, containerized environments are instantiated through Docker Compose, while SQL scripts and Python utilities populate the operational stores.

The ETL layer implements a multi-phase workflow in which raw data is first extracted and annotated with a flag indicating update status. Database triggers ensure that modified records are marked for reprocessing, enabling incremental synchronization. Subsequent transformation routines harmonize conflicting attribute values, such as differing genre classifications or user demographic representations, while incorporating slowly changing dimension (SCD) mechanisms to maintain full historical context. Specifically, a hybrid approach combining SCD Type 1 for rapid corrections and SCD Type 2.3 for preserving temporal snapshots is employed. Transformed data is then loaded into a star-schema model hosted on SQL Server, leveraging surrogate keys to optimize storage and query performance. The dimensional design comprises a core FACTS table (SESSIONS) joined to four conformed dimensions: USERS (tracking user history), CONTENTS (standardizing content attributes), DEVICES (cataloging viewing platforms), and TIMES (enabling fine-grained time-series analyses).

To facilitate multidimensional analytics, the star-schema is exposed through SQL Server Analysis Services (SSAS) via an OLAP cube, which supports slice and dice operations, and drill-through to detailed fact records. By structuring the data cube to include measures such as total viewing duration, session counts, and active user tallies, the architecture enables rapid aggregation across multiple axes (e.g., content genre by age group). The BI layer is completed with two distinct Power BI dashboards. The first, provides enterprise-level overviews, including cross-tabulations of viewing metrics by demographic segments and device types, KPIs for average session duration and total active users, and trend visualizations. The second, focuses on per-user analytics, displaying personalized session histories, distribution of viewing time across content categories, and comparative metrics against cohort averages. Interactive elements such as slicers and drill-through buttons enable analysts to navigate from aggregated insights to detailed session logs seamlessly.

By integrating modular, containerized ETL processes with rigorous dimension modeling techniques and an OLAP

powered analytics layer, this work demonstrates a scalable framework for streaming BI implementations. The use of Docker ensures that each operational database can be instantiated reproducibly, while Python based transformation scripts and SQL driven SCD logic guarantee consistent handling of evolving data. The OLAP cube design enables stakeholders to query large datasets with sub-second response times, supporting both strategic and operational decision-making. Ultimately, the StreamFlix prototype validates the feasibility of orchestrating disparate data sources into a unified analytical solution, offering a template for future deployments in real-world, high-volume streaming environments.

II. Related work

The StreamFlix project rests on several data management and analytics areas, which include data integration, ETL workflows and dimensional modeling along with OLAP-based exploration. The project addresses a main problem which is integrating different data sources, a topic many people write about. Kimball in addition to Ross states that joining data from various operational systems, like flat files, MySQL as well as PostgreSQL, needs careful planning. This planning ensures that downstream analytical layers are consistent and reliable [1]. The project uses Python scripts for this instead of common ETL platforms like Talend or Pentaho. This approach fits the exact needs of each source system, so it allows more openness plus control for the data pipeline.

Handling slowly changing dimensions (SCDs) is another important aspect of the project. In particular, the adoption of SCD Type 2.3 allows tracking the evolution of user and content data over time through the use of validity intervals and status flags, preserving historical integrity in the dimensional model [2]. This approach is vital for analytical systems that aim to study user behavior trends and content lifecycle dynamics and is consistent with best practices suggested in both industry and academic works on data warehousing [3].

Dimensional modeling for streaming platforms is increasingly relevant given the growing demand for personalized content delivery. Research has shown that star schemas centered on user sessions, supported by user, content, time, and device dimensions, are effective for uncovering behavioral patterns and improving engagement metrics [4]. StreamFlix applies these principles to support use cases such as churn prediction, content popularity analysis, and temporal viewing trends.

To enable advanced analytics and reporting, the project leverages OLAP technology through Microsoft SQL Server Analysis Services (SSAS), which facilitates the creation of multidimensional cubes for real-time data exploration. Dashboards built in Power BI allow users to interact with the data using drill-downs, filters, and drill-through operations. As noted by Chaudhuri and Dayal, OLAP systems are fundamental for supporting complex queries over large datasets in a performant and user-friendly manner [5]

III. Methodology

The StreamFlix project followed a structured methodology to simulate a real-world streaming analytics environment using synthetic data and a fully integrated Business Intelligence (BI) pipeline. The methodology included controlled data generation, operational system provisioning, schema definition, and implementation of a robust Extract–Transform–Load (ETL) process culminating in an OLAP-based analytical layer.

A. Terminology and Abbreviations

Throughout this paper, the following abbreviations are used:

- ETL – Extract, Transform, Load
- OLAP – Online Analytical Processing
- SCD – Slowly Changing Dimensions
- DW – Data Warehouse
- CSV – Comma-Separated Values

Each of these terms is introduced in context when first mentioned and is consistently applied in all technical descriptions.

B. Data Units and Format

Data volumes are measured in rows/records (e.g., 10,000 user records), and time-based fields are represented using ISO 8601 standard formats (e.g., YYYY-MM-DD HH:MM:SS). Decimal values are expressed using a leading zero (e.g., 0.75), and categorical fields are harmonized using domain-specific dictionaries (e.g., mapping “Sci-Fi” to “Science Fiction”).

C. Tools and Technologies

The following tools and technologies were used in the implementation:

- Python: Main language for data generation and ETL scripting, using libraries such as “pandas”, “faker”, “psycopg2”, and “pyodbc”.
- Docker & Docker Compose: Used to deploy containerized versions of MySQL and PostgreSQL operational systems.
- Microsoft SQL Server: Hosting the Data Warehouse and supporting SSAS for OLAP modeling.
- Power BI: Used to design interactive dashboards connected to the OLAP cube.
- SQL: All schema creation, triggers, and dimension logic implemented using standard SQL scripts.

D. Process Methodology

The methodology followed a modular and incremental development approach:

1. Data Modeling: Initial conceptual models were defined for all the operational systems. These guided both synthetic data creation and schema design.
2. Synthetic Data Generation: Python scripts simulated realistic datasets for each operational system, controlling consistency, referential integrity, and variability.
3. Operational Setup: Docker containers instantiated isolated environments for each source system. SQL scripts created the tables and populated them with generated data.

4. Incremental Change Simulation: Triggers and flags (up_to_date) were added to operational tables to simulate real-time data evolution and enable partial refreshes.
5. ETL Implementation: Python-based ETL processes extracted only updated records, applied necessary transformations (e.g., SCD logic, value mapping), and loaded data into a star-schema model in SQL Server.
6. OLAP & Dashboards: An SSAS cube was built on top of the star schema. Two Power BI dashboards were developed to visualize aggregated metrics and user-level details.

IV. SYSTEM DESIGN

The architecture of the StreamFlix analytical platform was structured to enable seamless integration of heterogeneous data sources, robust historical tracking, and real-time analytical capabilities. This section presents the system's design across its major components: operational systems, ETL pipeline, dimensional model, OLAP infrastructure, and reporting layer.

A. Authors and Affiliations

The data architecture integrates four operational systems:

- CSV System: Stores user viewing sessions with fields such as device type, OS, app version, and watch duration.
- MySQL Database: Contains metadata for video content, including type, genre, age rating, and director.
- PostgreSQL1: Manages user data, including demographics, location, and subscription status.
- PostgreSQL2: An additional, independent system simulating an alternative data source, aggregating information from all entities with renamed fields and different allowed values.

All systems include a technical attribute “is_up_to_date” and triggers to flag outdated records upon updates, enabling incremental processing.

B. ETL Pipeline

The ETL (Extract, Transform, Load) process was fully implemented in Python:

- Extract: Custom scripts retrieve data from CSV, MySQL, and both PostgreSQL databases. Connections were managed using “psycopg2”, “mysql-connector-python”, and standard CSV readers.
- Transform: Data from each source is harmonized using mapping dictionaries to unify nomenclature and value domains (e.g., mapping “Adventure” to “Action”).
    - SCD Type 2.3 is applied to dimensions USERS and CONTENTS, enabling historical tracking through “initial_date”, “final_date”, and “active” fields.
    - Devices follow SCD Type 2.1, and Time is static (Type 0).
- Load: The transformed data is loaded into a SQL Server Data Warehouse using “pyodbc”, preserving surrogate keys and data lineage (“source” field).

C. Dimensional Model

The Data Warehouse follows a star schema with the SESSIONS fact table at the center. It connects to four dimensions:

- Users: Contains age group, gender, full address hierarchy, and subscription details, with historical versioning.
- Contents: Includes genres (multi-valued), type, duration, age rating, and director.
- Devices: Captures the full device taxonomy (platform, device type, OS family, OS name, app version).
- Times: Provides temporal granularity down to minute level, supporting trend analysis and peak usage detection.

Fact metrics include watched duration and watched percent. Surrogate keys were used throughout to optimize storage and querying.

D. OLAP and Reporting Layer

For multidimensional analysis, Microsoft SQL Server Analysis Services (SSAS) was used to define and deploy an OLAP cube. The cube supports:

- drill-through operations
- Slicing and dicing across all dimensions
- Aggregation over temporal and categorical hierarchies

Reports were developed in Power BI, leveraging native integration with SSAS. The dashboards allow business users to filter content by demographics, platforms, time periods, and content types, facilitating actionable insights into user behavior, churn risks, and content popularity.

E. System Architecture Overview

TABLE I. SUMMARY OF THE OPERATIONAL SYSTEMS USED

System	Technology	Data Stored	Data Stored
CSV File	Flat File	Viewing sessions	Simulates raw session logs from frontend apps
MySQL	Relational DB	Content metadata (title, genre, etc.)	Uses normalized schema and many-to-many relations
PostgreSQL1	Relational DB	User profiles and subscriptions	Rich demographic data with relational modeling
PostgreSQL2	Relational DB	Integrated data from other systems	Alternative schema with renamed fields and values

Table 1 summarizes the four operational systems used in the StreamFlix platform. Each source provides distinct data types and modeling characteristics, influencing the ETL design.

Figure 1 illustrates the complete architecture of the StreamFlix analytical system, highlighting the data flow from source to consumption. The design follows a modular, layered approach that separates concerns and ensures maintainability.

At the base of the architecture are the data generation scripts, which replace the originally proposed TPC-H tool. These Python scripts synthesize realistic user, content, and session data, which are then distributed across four operational systems: a CSV file for session records, a MySQL database for content metadata, and two PostgreSQL instances for user profiles and integrated data respectively.

The ETL layer, developed entirely in Python, handles extraction, transformation, and loading of data into a centralized Data Warehouse implemented in Microsoft SQL Server. It includes:

- Change detection mechanisms based on "is_up_to_date" flags and triggers,
- SCD (Slowly Changing Dimensions) support (Types 0, 1, 2.1, and 2.3),
- Value harmonization and source tracking via a "source" attribute.

Following the ETL process, the data is processed by SQL Server Analysis Services (SSAS), where an OLAP cube is constructed. This cube enables multidimensional queries and exposes data to Power BI, where interactive dashboards provide decision-makers with insights into viewing habits, churn indicators, and platform performance.

The architecture thus combines synthetically generated data, distributed operational sources, a custom ETL engine, a dimensional model optimized for historical tracking, and modern BI tools to form a complete and scalable analytical solution.

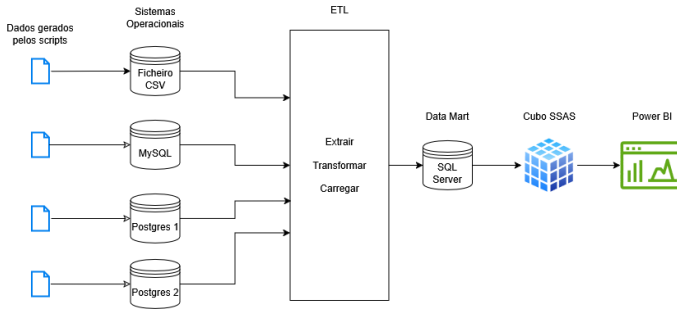

Figure 1- architecture of the StreamFlix analytical system

V. IMPLEMENTATION

The implementation phase of the StreamFlix project consisted of designing and deploying a complete Business Intelligence (BI) solution for a simulated video streaming platform. The architecture integrates multiple heterogeneous operational systems, applies a robust ETL process, and culminates in a multidimensional analytical layer supported by OLAP technologies. The implementation was divided into four main components: data source generation, operational database deployment, ETL processing, and OLAP-based visualization.

A. Data Source Generation

The project began with the development of synthetic datasets to simulate real-world activity on a streaming platform. Unlike the original plan to use TPC-H benchmarks, the team opted to generate custom data using Python scripts. This approach provided greater flexibility in designing data aligned with the domain requirements, such as video content data, user demographics, viewing sessions, device specifications, and time dimensions.

Each data source was designed to follow an independent schema. The data generation scripts utilized libraries such as Faker, random, and pandas to ensure controlled randomness, data consistency, and compliance with the previously defined conceptual models. The scripts produced CSV files for four distinct sources:

1. CSV file: representing the raw log of user viewing sessions.
2. MySQL: representing the catalog of video content.
3. PostgreSQL1: simulating user-related information.
4. PostgreSQL2: an additional PostgreSQL instance aggregating multiple entity types into a single operational schema.

All identifiers followed a deterministic format (e.g., CONTENT_00000001, USER_00000001) to facilitate tracking across systems. Moreover, each script ensured referential integrity and aligned attribute values with the business rules defined in the data dictionary.

B. Operational Database Deployment

To simulate realistic data integration challenges, the team deployed each operational system in isolated Docker containers using docker-compose. Each container included the database engine and associated credentials, aligned with typical configurations used in production environments.

After container initialization, SQL scripts were used to instantiate all tables. Following this, the generated CSVs were loaded into their respective databases. A set of alteration scripts was also executed to:

- Add a column is_up_to_date to each table to flag stale versus current data.
- Create triggers to automatically set is_up_to_date = 0 whenever a record was updated, thus marking it for reprocessing during ETL.

This mechanism supported efficient incremental loading and was critical for later validation scenarios involving data mutations.

C. ETL Processing

The ETL pipeline was developed in Python and consisted of three primary stages: extraction, transformation, and loading.

Extraction: Data was extracted from each source system using a Python script.

Transformation: This stage was responsible for:

- Normalizing data formats (e.g., date formats, value mappings).
- Generating surrogate keys for all dimensions.
- Applying Slowly Changing Dimensions (SCD) logic, particularly SCD Type 2.3 for USERS and CONTENTS, which preserved history via initial_date, final_date, and active flags.
- Consolidating entities into unified structures (e.g., GENRES vs. CATEGORIES).
- Assigning a source field to track data origin (postgresql1, postgresql2).

Loading: The final step involved loading the cleaned and transformed data into a dimensional model implemented in Microsoft SQL Server. The model followed a star schema with the following structure:

- Fact table: SESSIONS, containing viewing metrics and foreign keys.
- Dimension tables: USERS, CONTENTS, DEVICES, and TIMES, each with unique keys and attributes.

D. OLAP Cube Development and Visualization

With the Data Mart populated, the next step involved developing an OLAP cube using SQL Server Analysis Services (SSAS). The cube, named CuboStreamFlix, was created via Visual Studio and imported the SQL Server schema as its data source.

Two calculated columns, UserSourceKey and ContentSourceKey, were defined to allow aggregation across SCD versions. Time-related attributes (e.g., month_name) were also configured with proper sorting to support time-series analysis.

The cube was deployed and connected to Power BI, where two dashboards were developed:

- General Dashboard**:** Included filters (slice and dice), charts, KPIs, and drill-through buttons.
- Individual Dashboard**:** Provided detailed views for a selected user, including preferences by genre, device, and time.

VI. RESULTS AND ANALYSIS

The outcomes of the StreamFlix project were assessed through both functional validation and analytical capabilities. The system was tested on its ability to integrate data, apply version control, and generate actionable insights.

A. Functional Validation

To validate the ETL logic, particularly the management of updates and insertions, a series of controlled modifications were introduced into each data source. A Python script (alter_os_data.py) and accompanying SQL files were used to simulate updates to user profiles, content data, and viewing sessions.

After rerunning the ETL pipeline, checks were performed in SQL Server Management Studio. The system correctly applied the SCD 2.3 rules:

- Previous records had active = 0 and a populated final_date.
- New records reflected updated data with active = 1 and a new initial_date.

This confirmed that the system maintained historical integrity and allowed temporal analysis over evolving data.

B. Dashboard Interactivity and Analytical Depth

The Power BI dashboards provided multi-dimensional views into the platform's data. In the General Dashboard, stakeholders could filter by year, gender, content type, subscription status, and more. Metrics such as total sessions, average watch time, and content popularity were displayed dynamically.

Drill-through functionality enabled user-specific exploration. For example, selecting a user revealed:

- Individual preferences (by content type and device).
- Time-based viewing trends.
- Subscription and demographic information.

Figure 2 and Figure 3 provide visual evidence of the dashboards' interactivity and insight delivery.

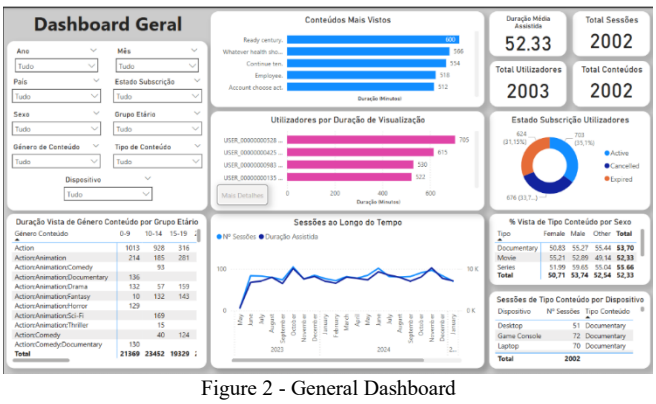

Figure 2 - General Dashboard

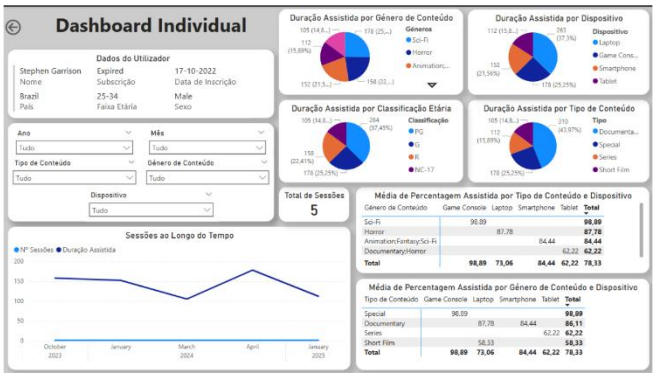

Figure 3 - Individual Dashboard

These features demonstrate the system's capability to support key business questions, such as:

- What content is most engaging for a specific demographic?
- Are certain devices correlated with higher watch durations?
- How does user behavior evolve over time?

C. *Performance and Scalability Considerations*

Although not the primary focus, the implementation showed good performance with data volumes in the order of tens of thousands of records. The use of incremental extraction (is_up_to_date), Docker-based parallelism, and indexing in the dimensional model contributed to efficient data flow.

VII. CONCLUSION

The StreamFlix project successfully illustrates the design, implementation, and validation of end-to-end Business Intelligence architecture tailored to a streaming platform scenario. By generating realistic operational data and building a modular ETL process, the system achieved a seamless integration of heterogeneous data sources.

The adoption of a star schema with SCD Type 2.3 ensured both data consistency and historical traceability. The deployment of an OLAP cube and its integration with Power BI enabled multidimensional analysis through intuitive dashboards, empowering stakeholders to explore complex usage patterns interactively.

The system proved robust in handling data updates and showed capacity to evolve with growing datasets. Future work could focus on:

- Automating ETL orchestration (e.g., using Apache Airflow).
- Benchmarking performance under larger volumes.
- Extending the model with predictive analytics modules.

In summary, StreamFlix provides a solid foundation for advanced analytics in digital content platforms and serves as a replicable model for educational and professional BI environments.

REFERENCES

[1] Kimball, R., & Ross, M. (2013). *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling* (3rd ed.). Wiley.

[2] Inmon, W. H. (2005). *Building the Data Warehouse* (4th ed.). Wiley.

[3] Golfarelli, M., & Rizzi, S. (2009). *Data Warehouse Design: Modern Principles and Methodologies*. McGraw-Hill.

[4] Zhou, T., He, Z., & Pei, J. (2020). Mining User Behavior in Subscription-Based Video Platforms. *IEEE Transactions on Knowledge and Data Engineering*.

[5] Chaudhuri, S., & Dayal, U. (1997). An Overview of Data Warehousing and OLAP Technology. *SIGMOD Record*, 26(1), 65–74.