**Movie Data Warehouse: Creation and Documentation Manual**

This manual, documents the creation of the normalized **Movie Box Office Data Warehouse**, detailing the Extract, Transform, and Load (ETL) processes used to convert four messy staging tables into a robust, relational schema.

**1. System Setup and Data Extraction (E)**

This section covers the prerequisite tools and the initial process of pulling raw data.

**1.1 Prerequisites and Connectivity**

The pipeline relies on Python libraries to interact with the PostgreSQL database.

| Component | Library/Detail | Purpose |
| --- | --- | --- |
| **Data Manipulation** |  | pandas | Provides the DataFrame object for structured data handling. |
| **SQL Toolkit** |  | sqlalchemy | Object-Relational Mapper (ORM) for simplified database interactions. |
| **DB Driver** |  | psycopg2 | PostgreSQL database adapter used by sqlalchemy |  |

1.2 Database Configuration

The script uses the following local connection parameters:

* **Database Name:** movies\_db
* **User:** postgres
* **Host/Port:** localhost:5432
* **Username and Password**

**1.3 Data Sources**

Four staging tables were extracted from the database into DataFrames for processing :

expert\_reviews ,

meta ,

sales , and user\_reviews.

**2. Data Transformation and Standardization (T)**

The transformation phase focused on cleaning data types, standardizing movie titles, and normalizing dimensions (lookups).

**2.1 General Cleanup and Title Normalization**

1. **Empty Column Removal:** All staging DataFrames had columns that were completely empty dropped.
2. **Title Cleaning:** A cleaning method was applied to create the **title\_clean** column. This column is the standard identifier used for all merging operations. For review tables, the title was fetched from the

url.

**2.2 Table-Specific Transformations**

| Table | Column(s) | Transformation Logic |
| --- | --- | --- |
| **sales** | international\_box\_office, etc. | Cleaned text (removed currency symbols and letters), converted to numeric type, and cast to integer. |  |
| **meta** | RelDate | Read the date and extracted only the **year** to populate a new year column. |  |
|  | rating | Cleaned data by clearing special characters to standardize rating names. |  |
|  | userscore | Cleaned text and converted to Int64. |  |
| **user\_reviews** | idvscore | Rows were dropped if the value was blank or non-numeric. |  |
|  | reviewer | Rows were dropped if the cell contained no data. |  |
| **expert\_reviews** | idvscore | Values were **multiplied by 10** for consistent scaling (0-100), and invalid/blank rows were dropped. |  |

**2.3 Normalization: Lookup Table Creation**

Lookup tables were created from unique values in the staging tables, enforcing a one-to-many relationship structure (Normalization).

| Table Name | Source Column | Sources |
| --- | --- | --- |
| **genre** | sales.genre | genre\_id (Primary Key) |  |
| **director** | meta.director | director\_id (Primary Key) |  |

**3. Data Loading and Relational Schema (L)**

The final phase created the normalized tables and established all Primary and Foreign Key relationships.

**3.1 The Central movies Table**

The **movies** table serves as the primary entity and linking hub.

1. **Consolidation:** Unique movie identifiers were consolidated from sales and meta based on the cleaned title and year.
2. **Primary Key:** The table was defined with a **SERIAL Primary Key** named **movie\_id**.
3. **Attribute Fetching:** Relevant attributes were fetched via a Left Merge:
   * **From meta:** rating ,

metascore ,

studio ,

runtime.

* + **From sales:** genre.

**3.2 Relational Tables and Foreign Keys**

The final schema involves several specific relational and analytical tables:

| Table Name | Primary Key (PK) | Foreign Key(s) (FK) | Source Data |
| --- | --- | --- | --- |
| **reviews** |  | review\_id (SERIAL PK) | Consolidated from user and expert reviews. | |  |  |
| **reviewer** |  | reviewer\_id (SERIAL PK) | Fetched from reviews table | |  |  |
| **awards** |  | award\_id (SERIAL PK) | Unique, cleaned values from | |  |  |
| **movie\_awards** | None (Junction Table) |  | Links movies to their awards and ranking. | |  |  |
| **movie\_director** | None (Junction Table) |  | Links movies to their corresponding directors. | |  |  |

**3.3 Final Linking of Staging Tables**

**movie\_id** Primary Key was added as a **Foreign Key** to all four original staging tables (sales, meta, expert\_reviews, user\_reviews). This was achieved by merging the staging DataFrames with the new

movies table, matching rows on the composite key of **title\_clean and year**.

**3.4 Post-Load Fixes (Documentation)**

The following corrections were documented after the initial load to correct score scaling in the

reviews table:

* **User Scores:** (Applied to scale 10-point user scores to 100).
* **SQL Script:**

UPDATE reviews SET idvscore = (idvscore \* 10) WHERE review\_type = 'user'

* **Expert Scores:** (Applied to correct scores if the prior *10* step was run, or to ensure they are on a 0-10 scale depending on the final standard).
* **SQL Script:**

UPDATE reviews SET idvscore = (idvscore / 10) WHERE review\_type = 'expert'