Name: Jingxi Hu, Sofia Brazda, Andrew Brockenborough, Louis Miguel Sabaricos
Class: CSC-675
Section IV: Testing and Analysis

1. Testing Requirements

Create a script named `Tests.sql` to implement the following components. Ensure that all the components are well documented in your script

a. Unit Testing

- Test individual SQL components such as **Triggers**, **Functions**, and **Procedures** to ensure they function as expected.
- Use test cases for each requirement implemented in the previous section.
    - For **Triggers**: Test scenarios where the trigger conditions are met, and ensure they fire correctly (e.g., removing items from a Watchlist when it exceeds the limit).
    - For **Functions**: Verify that the functions return the correct results based on test data (e.g., ranking genres by watch hours).
    - For **Procedures**: Ensure that the stored procedures generate correct reports or perform updates (e.g., generating user activity reports, handling failed payments).

b. Integration Testing

- Test the interaction between the components of your system. Ensure that:
    - Data flows correctly between tables.
    - Triggers fire in response to the correct actions.
    - Procedures and functions return expected results when used together.
- For example, after inserting a new review, check if the **Content_Availability** status is updated appropriately based on the average rating.

c. Data Integrity Testing

- Verify that data integrity is maintained throughout the operations. For example:
    - Ensure no duplicate **Director** entries are added for the same **Content**.
    - Validate that expired subscriptions are correctly removed from the **User_Subscription** table.

d. Performance Testing

- Test the performance of your database, especially for larger datasets. Ensure that:
    - Triggers, functions, and procedures execute within an acceptable time frame.

    - Scheduled events run efficiently, even when handling large amounts of data (e.g., updating content rankings daily).

2. Analysis

Write a final analysis report `Report.pdf` summarizing your testing process for the following components.

- Include screenshots or logs of any testing done, especially if issues were resolved during the process.
- Upload the final report to the `Project/Reports/` directory of your project.

a. Results of Testing

- Document the results of each test and compare them to the expected outcomes.
    - For each test case, list the input, expected output, actual output, and whether the test passed or failed.
    - If a test failed, describe the issue and how you resolved it.

b. Performance Insights

- Analyze the performance of your queries, especially for large datasets. Consider:
    - Query execution time and optimization (e.g., adding indexes, adjusting query structure).
    - The impact of scheduled events on database performance.

c. Improvements and Recommendations

- Based on your testing and analysis, suggest any improvements or optimizations for your database design or logic. Consider:
    - Indexing strategies to speed up queries.
    - Refactoring inefficient queries or logic in triggers, functions, or procedures.

3. Documentation

- Ensure that all tests and analysis are well-documented. This includes:
    - Test cases, their expected and actual results.
    - Performance benchmarks and optimization strategies.
    - Any changes made to the database schema or logic after testing.