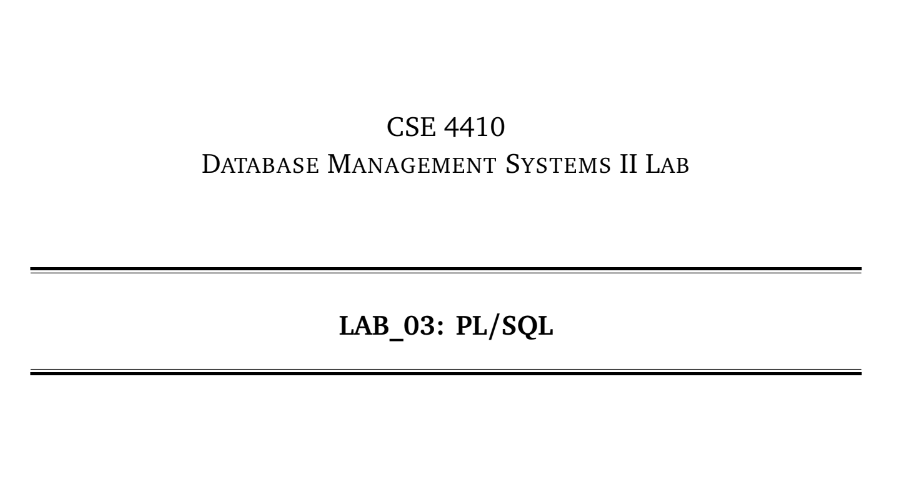
**LAB REPORT**



**NAME: CHOWDHURY ASHFAQ**

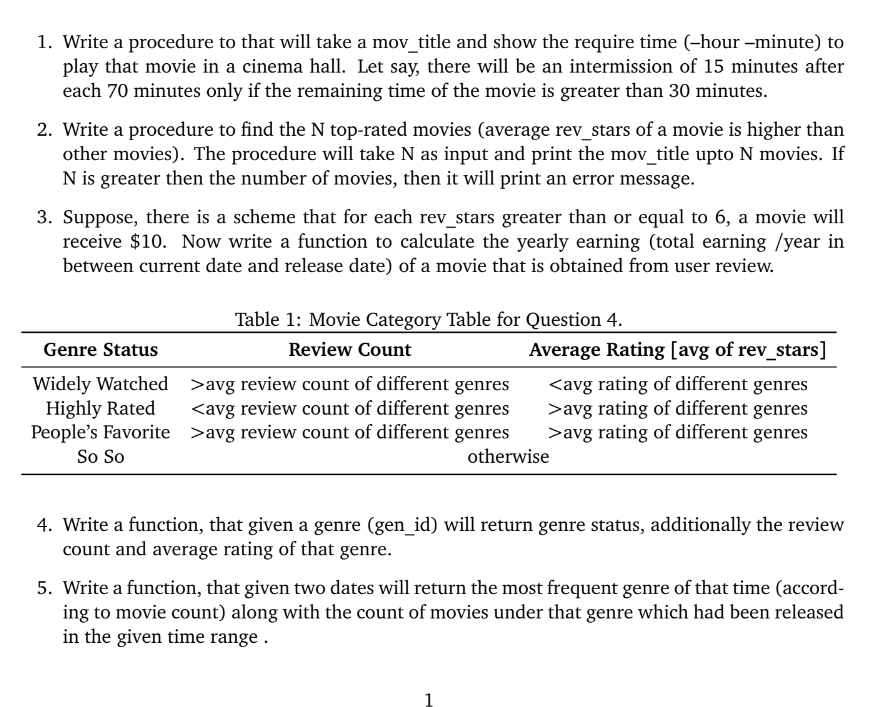
**STUDENT ID: 200042123**

**PROGRAM: SWE**

**GROUP: 1A**

**DATE: 30/01/23**

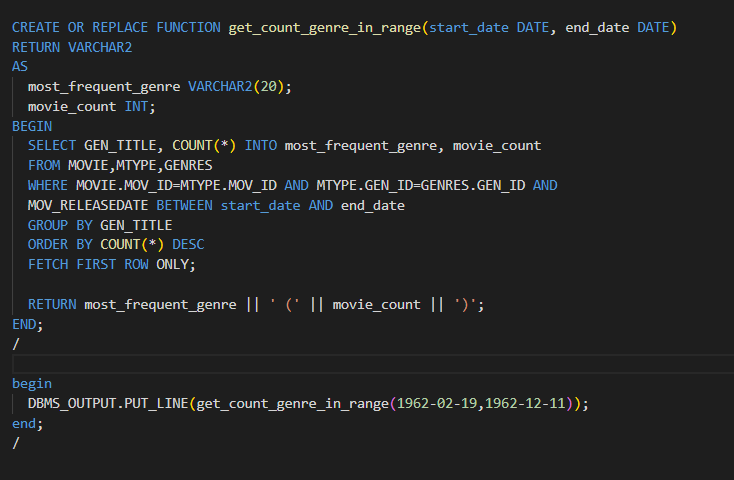
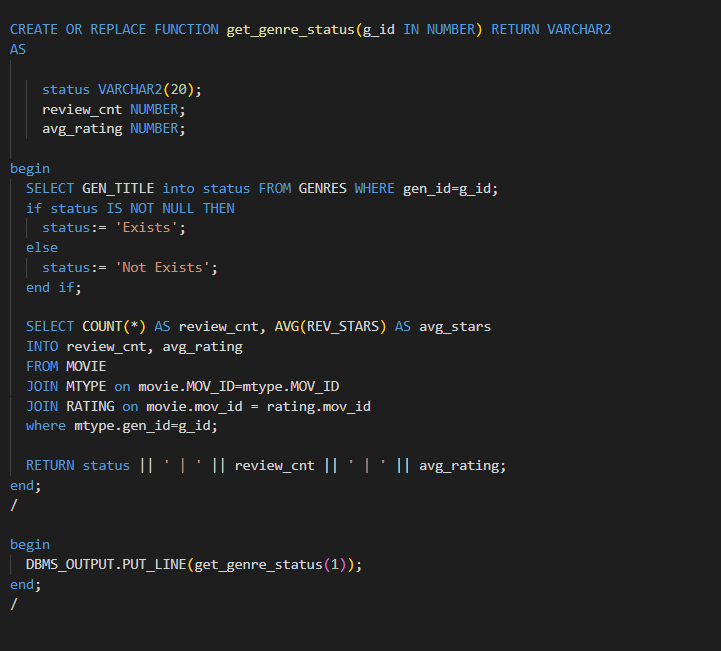
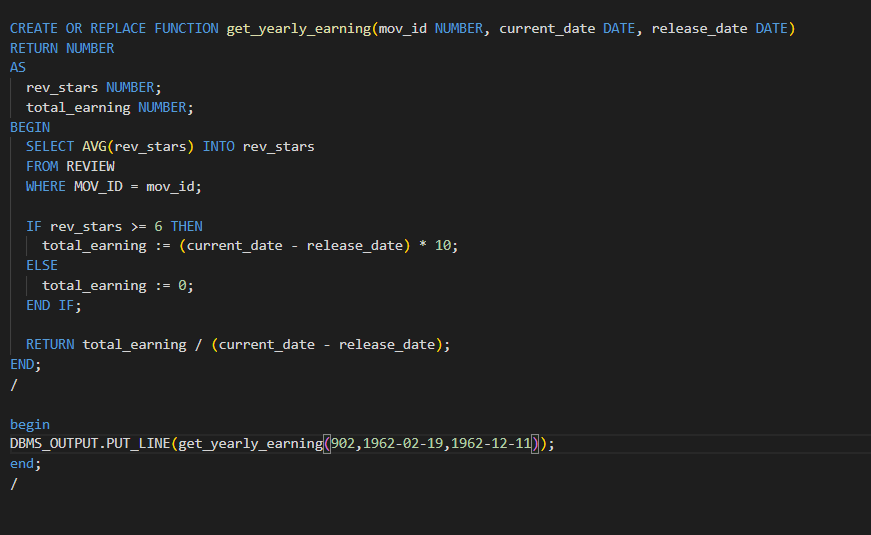
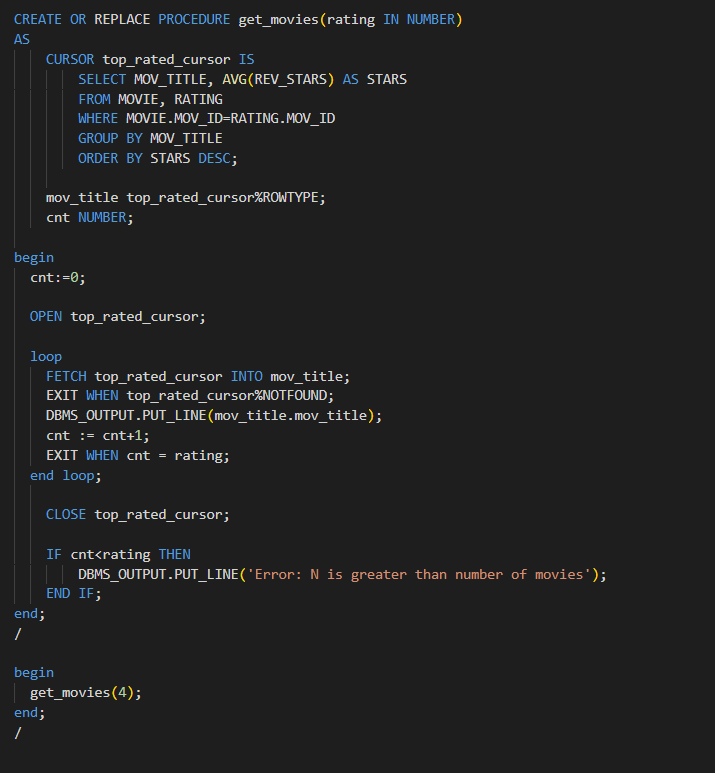
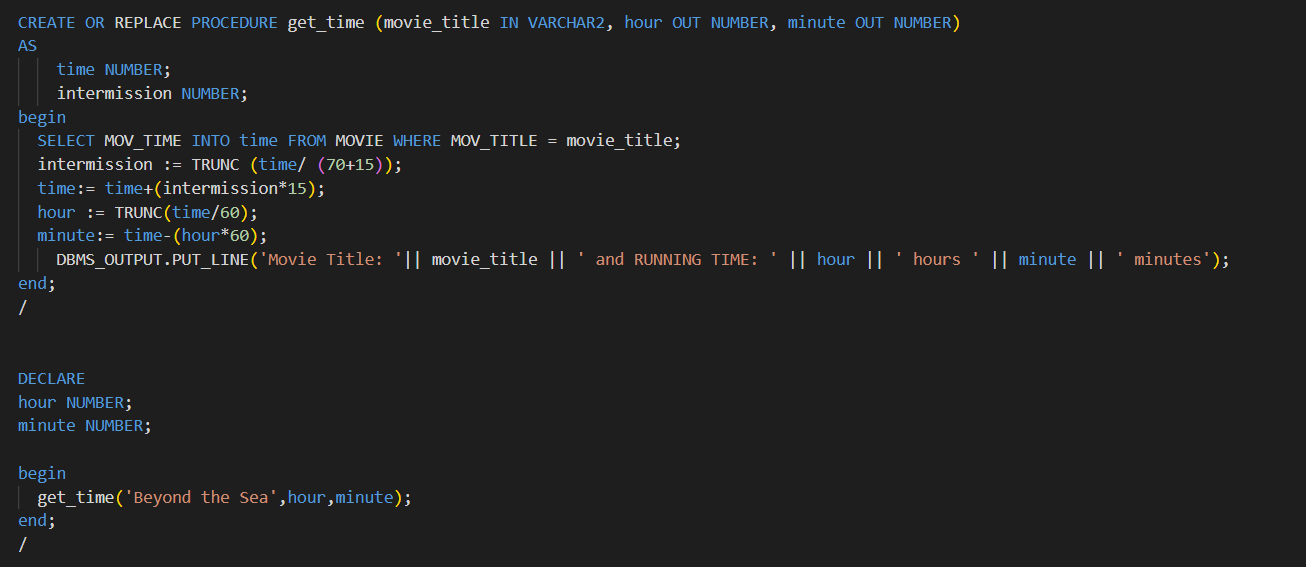
**Tasks:**



**Analysis of the problem:**

Functions and Procedures are an important part of PL/SQL. We were given a scenario and based on that scenario we had to write 2 Procedures and 3 functions.

**Solution:**



**Explanation:**

1)

get\_time is a procedure which gives the playing time of a movie as OUT parameter based on movie\_title. The procedure starts by using a SELECT statement to retrieve the time of the movie from the MOVIE table. The intermission is calculated as per the format (70 + 15) minutes. Finally, the hour and minute are calculated by dividing the time by 60 and storing the result in hour and subtracting the result of hour \* 60 from time and storing it in minute.

2)

The procedure get\_movies() retrieves a list of top rated movies from the MOVIE and RATING tables, based on the average rating (REV\_STARS) of each movie. The procedure accepts a parameter 'rating' which represents the number of top rated movies that the user wants to retrieve.

The procedure has a cursor 'top\_rated\_cursor' which retrieves the movie title and the average rating of each movie. The cursor is opened and looped through to retrieve each movie title. The loop will fetch each row from the cursor and store it in the mov\_title record. The loop continues until the cursor reaches the end of the data or until the number of rows fetched is equal to the value of 'rating' specified by the user. The cursor is closed when the loop completes.

 DBMS\_OUTPUT.PUT\_LINE(mov\_title.mov\_title);

This line generates the required output.

3)

The function "get\_yearly\_earning" calculates the yearly earning of a movie based on its review stars. The function first retrieves the release date and current date of the movie. It then calculates the number of years between the current date and the release date by subtracting the year part of the release date from the year part of the current date. The function then calculates the total earning of the movie by multiplying the number of years by $10, assuming that for each review star greater than or equal to 6, the movie will receive $10. The function returns the yearly earning of the movie by dividing the total earning by the number of years.

4)

The function first checks if the genre exists by querying the GENRES table using the input g\_id. If the genre exists, the status is set to 'Exists'. If not, the status is set to 'Not Exists'.

Then, the function calculates the count of movie reviews and the average rating for movies of the genre represented by the input g\_id by joining the MOVIE, MTYPE, and RATING tables. It returns genre status, review count and average rating of that genre.

5)

The function starts by selecting the GEN\_TITLE (genre title) and the count of movies for that genre from the MOVIE, MTYPE, and GENRES tables. The join is done on the MOV\_ID column in all three tables to obtain the data from all three tables.

Then, the query filters the results to only include the movies that were released between the start\_date and end\_date.

Finally, the query groups the data by GEN\_TITLE, orders the count of movies in each genre in descending order, and fetches only the first row of the result set. This first row represents the most frequently occurring genre in the given date range. In the end the function returns most frequent genre along with the count of movies under that genre.

**Problems Faced:**

Question was a bit complex to understand like the third one.