**DBMS Lab Record Cycle**

# ER Diagram & Table Design

* + E-R Diagram and table reduction
  + Table descriptions

**Movie database:**

**Table name: Directors**

**Description: Used to store Directors Information**

| **Attribute** | **Data Type** | **Constraints** |
| --- | --- | --- |
| Id | Int | Primary Key/ Not Null |
| Name | Varchar2(40) | Not Null |

**Table name: Stars**

**Description: Used to store Stars Information**

| **Attribute** | **Data Type** | **Constraints** |
| --- | --- | --- |
| Id | Int | Primary Key/ Not Null |
| Name | Varchar2(40) | Unique |
| About | Varchar2(100) |  |

**Table name: Movies**

**Description: Used to store Movies Information**

| **Attribute** | **Data Type** | **Constraints** |
| --- | --- | --- |
| Id | Int | Primary Key/ Not Null |
| Title | Varchar2(40) | Not Null |
| R\_date | Date |  |
| Image\_url | Varchar2(100) |  |
| Certificate | Varchar2(20) |  |
| Runtime | Number(3,2) |  |
| ImdbRating | Number (3,1) | By default 0 |
| Description | Text(100) | By default Null |
| Metascore | Number (3,1) | By default 0 |
| Votes | Int | By default 0 |
| Gross | Number(10,2) | Gross amount should be greater than 10000 |

**Table name: MoviesDirectors**

**Description: Used to store Movie Directors Information**

| **Attribute** | **Data Type** | **Constraints** |  |
| --- | --- | --- | --- |
| MoviesId | Int | Foreign Key references  Id of **Movies** table | Primary Key |
| DirectorsId | Int | Foreign Key references  Id of **Directors** table |

**Table name: MoviesStars**

**Description: Used to store Movie Stars Information**

| **Attribute** | **Data Type** | **Constraints** |  |
| --- | --- | --- | --- |
| MoviesId | Int | Foreign Key references  Id of **Movies** table | Primary Key |
| StarsId | Int | Foreign Key references  Id of **Stars** table |

1. **Practice SQL Data Definition Language(DDL) commands**

* Create the tables based on the above description.
* Add a column ‘DOB’ to **Stars** table.
* Drop the column ‘Gross’ in **Movies** table.
* Add column ‘Language’ in **Movies** table.
* Add column Gross Number(10,2) in **Movies** table.
* Change the name of the column ‘R\_date’ in **Movies table** to Releasedate.
* Add a column ‘Age’ in **Directors** table as Number. Age must be 7 years or above.
* Add a new column ‘Hit’ in **Movies** table with datatype Number(1) and by default 0.
* Add a new column ‘Entry\_date’ in Movies table to record the date on which the movie details are entered in the data base.
* Destroy the table **MoviesStars** and recreate it.
* Change the size of the Director’s name to 30.
* Add the following check constraints:
  + Releasedate should be less than the Entry\_date in the Movies table.
  + Language of movies should be Malayalam, English, Tamil or Hindi.

1. **Practice SQL Data Manipulation Language (DML) commands**
   1. Row insertion, deletion and updating

* Insert the appropriate data (10 rows) for the tables with respect to defined datatypes, size and constraints.
* Change value of Hit to 1 where ‘Votes’ greater than or equal to 90.
* Create table **IndustryHit** with the following columns:

Id

Title

Releasedate

Language

Votes

Gross

The data types and null characteristics for these columns should be

the same as the corresponding columns in the **Movies** table

described at the beginning of the lab exercise.

* New movies hit the box office; their data is as follows:

Id: 1014, 1021, 1032

Title: 2018: Everyone is a Hero, Oppenheimer, Maamannan

Releasedate: 5 May 2023, 21 July 2023, 29 June 2023

Language: Malayalam, English, Tamil

Votes: 97, 96, 95

Gross: 750000000, 500000000, 505000000

Add the new employees to the **IndustryHit** table.

* Insert data into the new **IndustryHit** table.
* Insert data into the **IndustryHit** table by copying the appropriate columns in the **Movies** table for those Movies that have Votes greater than or equal to 95.
* Movie Oppenheimer got a Metascore of 80. Make the appropriate data change.
* Delete all movies whose Metascore is less than 50.
* Movie ‘Voice Of Sathyanathan’ was released.

For ‘Voice Of Sathyanathan’ enter the following data:

Id: 1015

Title: Voice Of Sathyanathan

Releasedate: 28 July 2023

Image\_url: https://m.media-amazon.com/imak2M\_.jpg

Certificate: U

Runtime: 2.10

ImdbRating: 7.4

Description: A man's life becomes increasingly complicated after his neighbor is injured in a dispute over a fence.

Metascore: 60

Votes: 90

Gross: 109500000

* Delete all rows from **IndustryHit and drop the IndustryHit table.**
  1. Retrieval of data (Simple select query and select with ‘where’

options (include all relational and logical operators)

* List details of all movies
* List Title, Votes, Releasedate, Gross where Gross collection greater than 5000,000,00. Sequence the results in descending order by Gross.
* Retrieve the titles and years of Tamil movies released in 2022.
* Get the titles, years, and meta scores of movies sorted in descending order of meta scores.
* List titles, years, languages, dates and votes of all Malayalam and English movies released before 2022 and ImdbRating less than 7. The list should be ordered by Title.
* List all the movies whose title starts with ‘Open’. Order the result by descending order of their id.
* List Hit movies released in 2022 and 2023. Order the result by ascending order of their Titles.
* Retrieve movies with a runtime between 1.5 and 2.5 hours.
* Retrieve movies with Metascore ratings below 50 and IMDb ratings above 6.0.
* Retrieve movies with no description provided.
  1. Functions: Numeric Data, Character Conversion and Group functions
* Illustrate the different numeric functions using dual table (power,

round, ceil, floor, abs, exp, greatest, least, mod, trunc, round,

sign, sqrt etc.)

* Illustrate the character functions (upper, lower, initcap, length,

concat, ascii, substr, ltrim, rtrim, trim, translate, instr,

chr,Lpad,Rpadetc) using the table **Movies.**

* Illustration of conversion functions- to\_number,

to\_char(numberconversion), to\_char(dateconversion)

* Count the total no. of Movies
* Calculate the average votes of movies.
* Determine the maximum and minimum collection of movies. Rename the output as Max\_Coll and Min\_Coll respectively.
* Count the number of movies crossed the collection 50,00,00,000.
* Count the hit movies of 2021.
  1. Data manipulations using date functions
* Provide a list of all movies which were released on June 16, 2020.

Display the year and month of the released date and the Id. Sort the result by Id. Name the derived columns YEAR and MONTH.

* List the number of months between release date and entry date of

each movie.

* List the Entry\_date in the format ‘DD-Month-YY’.
* List the date, 8 days after today’s date.
* List all the movies which were released in the month of February.
* Illustrate the different date functions using dual table (to\_date,

Add\_months, last\_day, months\_between, next\_day, round etc.)

* Illustration of special date formats using to\_char function (use of

th,sp,spth)

* Calculate the total gross earnings for movies released after June 16, 2020.
  1. Set Operations
* Create a new table **IndustryHit** (Id, title, genre, Certificate, Gross, Releasedate)**.** Insert some movies from **Movies** table and some new movies in the new table **IndustryHit.**
* Retrieve the titles of all movies and industry hits which are in the action thriller genre.
* Retrieve the titles of all movies including industry hits.
* Retrieve the titles of all movies which are not industry hits.
  1. Illustration of Group By having clause
* For all genres, display genre type and the sum of all Gross for each genre. Name the derived column SUM\_COLL.
* For all genres, display the genre type and the number of titles. Name the derived column TITLE\_COUNT.
* Display the genres which have more than 3 titles.
* Retrieve the total number of movies released in each year, only for years with at least 5 movies.
* List the certificates along with the number of movies for each certificate, but only show certificates with more than 3 movies.
* Show the total gross earnings for each certificate, but only for certificates with total gross greater than $1 million.
* List the release years with the highest number of movies and the corresponding movie count, limited to the top 3 years.
  1. Sub queries
* Retrieve the titles and runtime of movies with the highest Metascore.
* List the titles of movies with a Gross amount greater than the average Gross amount of all movies.
* Retrieve the titles and descriptions of movies with a Metascore lower than the average Metascore.
* List the movie titles and their IMDb ratings for movies released in the year with the highest average IMDb rating.
* Retrieve the movie titles and their IMDb ratings for movies that have a Metascore greater than twice their IMDb rating.
* Find the title and gross amount of the top 3 highest-grossing movies.
* Calculate the total number of votes received by movies released in the year 2022.
* List the titles and certificate ratings of movies that have an IMDb rating below the average IMDb rating.

3.8 Views

* Create a view called MovieDetails that combines information from the Movies, Directors, and Stars tables to display movie titles, directors' names, and the names of stars who acted in those movies.
* Create a view called HighlyRatedMovies that displays movies with IMDb ratings greater than 8.0, including their titles and ratings.
* Create a view called DirectorMovies that provides a list of directors along with the number of movies they have directed.
* Create a view called StarMovies that displays stars' names and the titles of movies they have acted in.
* Create a view called LongestMovies that lists the titles of movies with the longest runtimes (duration).
* Create a view called LanguageDistribution that shows the distribution of movies based on the languages they were released in, including the count of movies for each language.
* Create a view called GrossEarnings that displays movies with their titles and gross earnings, sorted by earnings in descending order.
* Create a view called IndustryHitMovies that shows the titles and release dates of industry hit movies.
* Create a view called MovieVotes that displays movies along with their titles and the number of votes they have received.
* Create a view called CertifiedMovies that lists movies with their titles and certificates (e.g., U/A, U).

# 4.0 Practice PL/SQL

**4.1** **Introduction to PL/SQL**

* Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 3 to 7. Store the radius and corresponding values of calculated area in an empty table named Areas, consisting of two columns Radius and Area.
* Write a PL/SQL block of code for inverting a number accepted from the console.
* Write a PL/SQL code block that will accept an account number from the user and debit an amount of Rs.2000 from the account if the account has a minimum balance of 500 after the amount is debited. The process is fired on the Accounts table.
* Write a PL/SQL block of code that updates the salaries of Maria Jacob and Albert by Rs. 2000/- and Rs.2500/- respectively. Then check to see that the total salary does not exceed 75000. If the total salary is greater than 75000, then undo the updates made to salaries of both. (Use savepoint, rollback and commit).

**4.2 Illustration of Cursors**

**Illustration of Implicit cursor**.

* Write a PL/SQL block to accept an employee number and update the salary of that employee to raise the salary by 0.15. Display appropriate message based on the existence of the record in the employee table.
* The HRD manager decides to raise the salary of employees working as ‘analyst’ by 0.15. Write a cursor to update the salary of the employees. Display the no. of employee records that has been modified.

**Illustration of explicit cursor.**

* Write an explicit cursor to display the name,department, salary of the first 5 employees getting the highest salary.
* The HRD manager decides to raise the salary of employees working as ‘analyst’ by 0.15. Whenever any such raise is given to the employees, a record for the same is maintained in the emp\_raise table. It includes the employee number, the date when the raise was given and actual raise. Write a PL/SQL block to update the salary of the employees and insert a record in the emp\_raise table. **Emp\_raise(empcode, raisedate,raise\_amt)**

**4.3**  **Illustration of Procedures**

* Write a PL/SQL block which makes use of a stored procedure Proj\_emp ( emp\_name varchar2(50) ) which finds all the details of the projects involved by the given employee.
* Write a procedure to check whether a string is a palindrome . Call the procedure to list all the palindrome names in the employee table.
* Write a PL/SQL block which retrieve all the employee into a cursor and display the details of all assigned projects for each employee using a stored procedure Proj\_emp ( emp\_name varchar2(50).

# 4.4 Illustration of functions

* Write a function to find the reverse of EmpNo in Employee table and display the EmpNo and Reversed(Emp No) of the first 5 employees using an SQL Query.
* Write a function that would check for the existence of an employee in the employee table given an EmpNo. If existing employee, check whether he is the manager of any department and display messages accordingly.

**4.5 Illustration of Triggers**

* Consider the table Employee. Write PL/SQL statements to create a trigger when fired checks the operation performed on a table and based on the operation, a variable is assigned the value ‘update’ or ‘delete’. Previous values of the modified record of the table Employee are stored into the appropriate variables declared and inserted to the audit table AuditEmployee.
* Write PL/SQL statements to create a trigger which generates an error messages if the salary is below or beyond the valid range 0-5000 on the employee table. The triggering events are update and insert.
* Write PL/SQL statements to create a trigger that limits the DML actions to the Employee table to weekdays from 8.30am to 6.30pm. If a user tries to insert/update/delete a row in the Employee table, a warning message will be prompted.