CREATE DATABASE DA\_Assignment\_2

USE DA\_Assignment\_2

**--Tasks**

**--1. Complec Queries**

**-- A. Write a query to find the top 5 platforms with the highest average user ratings**

SELECT TOP 5 Platform, AVG(User\_Rating) AS Avg\_Rating

FROM dbo.games\_dataset

GROUP BY Platform

ORDER BY Avg\_Rating DESC;

**--B. Use Common Table Expressions (CTEs) to calculate the average user rating for each**

**--genre and identify genres with an average rating above a certain threshold.**

WITH GenreRatings AS (

SELECT Genre, AVG(User\_Rating) AS Avg\_Rating

FROM dbo.games\_dataset

GROUP BY Genre

)

SELECT Genre, Avg\_Rating

FROM GenreRatings

WHERE Avg\_Rating > 4.0;

**--2. Stored Procedures:**

**--Create a stored procedure to categorize games into 'High', 'Medium', and 'Low' ratings**

**--based on their user ratings and update the dataset accordingly.**

**--Creating column named Rating category**

ALTER TABLE dbo.games\_dataset

ADD Rating\_Category NVARCHAR(20);

**-- Create the stored procedure**

CREATE PROCEDURE dbo.Categorize\_Games

AS

BEGIN

SET NOCOUNT ON;

**-- Ensure the column exists before updating**

IF COL\_LENGTH('dbo.games\_dataset', 'Rating\_Category') IS NOT NULL

BEGIN

UPDATE dbo.games\_dataset

SET Rating\_Category =

CASE

WHEN User\_Rating >= 4.5 THEN 'High'

WHEN User\_Rating >= 3.0 THEN 'Medium'

ELSE 'Low'

END;

END

END;

GO

**--Execute stored Procedure**

EXEC dbo.Categorize\_Games;

SELECT \*

FROM

dbo.games\_dataset;

/\*

**Drop the procedure if it exists**

IF OBJECT\_ID('dbo.Categorize\_Games', 'P') IS NOT NULL

DROP PROCEDURE dbo.Categorize\_Games;

GO

\*/

**--3. Triggers:**

**--Write a trigger to automatically update the 'User Rating' column to a default value when**

**--a new game is inserted into the table without a specified rating.**

CREATE TRIGGER dbo.Set\_Default\_Rating

ON dbo.games\_dataset

AFTER INSERT

AS

BEGIN

SET NOCOUNT ON;

-- Update rows where User\_Rating is NULL

UPDATE g

SET g.User\_Rating = 3.5 -- Default value

FROM dbo.games\_dataset g

INNER JOIN inserted i ON g.Game\_Name = i.Game\_Name

WHERE i.User\_Rating IS NULL;

END;

GO

--With User\_Rating=Null Value

INSERT INTO dbo.games\_dataset (Game\_Name, Genre,Platform, Release\_Year, User\_Rating)

VALUES ('New Game', 'Action', 'PC', 2024, NULL);

**--Output:**

**--Cannot insert the value NULL into column 'User\_Rating', table 'DA\_Assignment\_2.dbo.games\_dataset'; column does not allow nulls. INSERT fails.**

--With User\_Rating=Value

INSERT INTO dbo.games\_dataset (Game\_Name, Genre,Platform, Release\_Year, User\_Rating)

VALUES ('New Game', 'Action', 'PC', 2024, 3.5);

SELECT \* FROM dbo.games\_dataset WHERE Game\_Name = 'New Game'

**--Deleting Added Rows**

DELETE FROM dbo.games\_dataset

WHERE Game\_Name = 'New Game' AND Platform = 'PC';

**--Drop the trigger if it already exists to avoid conflicts**

/\*

IF OBJECT\_ID('dbo.Set\_Default\_Rating', 'TR') IS NOT NULL

DROP TRIGGER dbo.Set\_Default\_Rating;

GO

\*/

**--4. Views:**

**--Create a view to display games with complete rating information and filter out games**

**--with missing data.**

CREATE VIEW dbo.Complete\_Ratings AS

SELECT \*

FROM dbo.games\_dataset

WHERE User\_Rating IS NOT NULL;

GO

**--Execute View**

SELECT \* FROM dbo.Complete\_Ratings;

/\*

IF OBJECT\_ID('dbo.Complete\_Ratings', 'V') IS NOT NULL

DROP VIEW dbo.Complete\_Ratings;

GO

\*/