Business Intelligence

Final Project Part 2 - MS SQL

Movies- Database Assignment

BY GROUP 10

Moganaviniith Rathinavel (200531668)

Aswin Narayanan (200568974)

Nanditha Babu (200573747)

Srivas Sathyanarayanan (200544678)

Raveen Kumar (200574798)

1.Create a database "Movies".

```
CREATE DATABASE Movies;
```

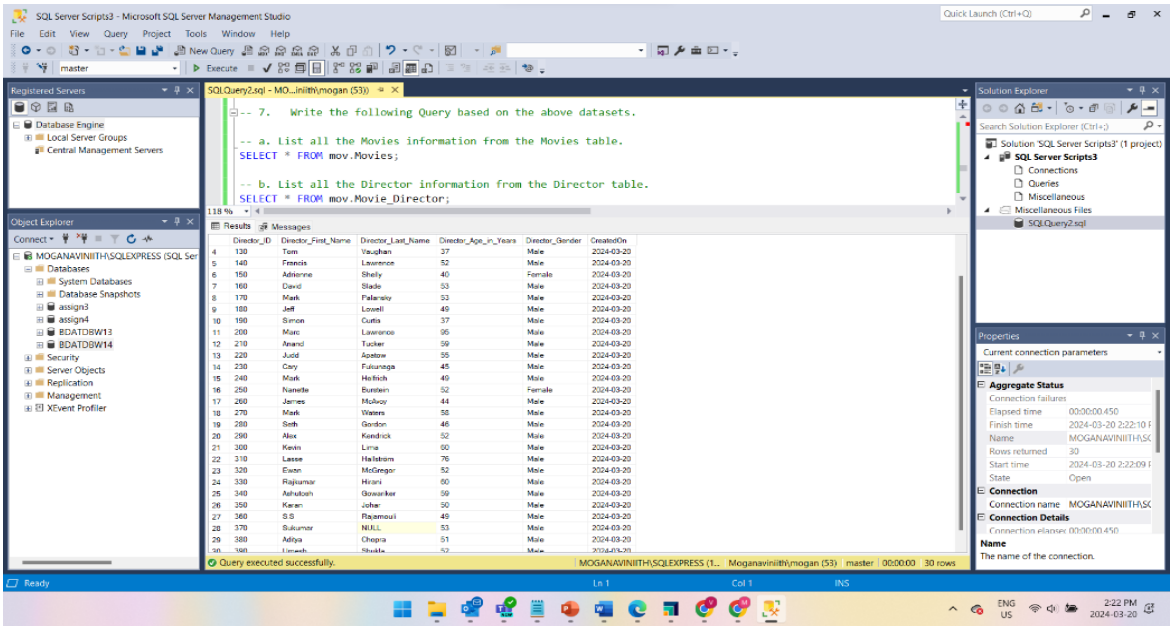
2.Create a "mov" schema under "Movies".

```
CREATE SCHEMA mov;
```

3.Create a table call "Movie_Director "under mov schema with the following specifications-

```
CREATE TABLE mov.Movie_Director (
  Director_ID INT IDENTITY(1,1) NOT NULL PRIMARY KEY CLUSTERED,
  Director_First_Name VARCHAR(255),
  Director_Last_Name VARCHAR(255),
  Director_Age_in_Years INT,
  Director_Gender VARCHAR(255), -- Assuming 'Director_Gender' is the correct column name
and 'Date of Joining' was mistakenly described
  CreatedOn DATE NOT NULL DEFAULT GETDATE()
);

SET IDENTITY_INSERT mov.Movie_Director ON;

INSERT INTO mov.Movie_Director (Director_ID, Director_First_Name, Director_Last_Name,
Director_Age_in_Years, Director_Gender, CreatedOn)
VALUES
(100, 'Kevin', 'Smith', 52, 'Male', DEFAULT),
(110, 'Miguel', 'Arteta', 41, 'Male', DEFAULT),
(120, 'Mark', 'Johnson', 58, 'Male', DEFAULT),
(130, 'Tom', 'Vaughan', 37, 'Male', DEFAULT),
(140, 'Francis', 'Lawrence', 52, 'Male', DEFAULT),
(150, 'Adrienne', 'Shelly', 40, 'Female', DEFAULT),
(160, 'David', 'Slade', 53, 'Male', DEFAULT),
(170, 'Mark', 'Palansky', 53, 'Male', DEFAULT),
(180, 'Jeff', 'Lowell', 49, 'Male', DEFAULT),
(190, 'Simon', 'Curtis', 37, 'Male', DEFAULT),
(200, 'Marc', 'Lawrence', 95, 'Male', DEFAULT),
(210, 'Anand', 'Tucker', 59, 'Male', DEFAULT),
(220, 'Judd', 'Apatow', 55, 'Male', DEFAULT),
```

```
(230, 'Cary', 'Fukunaga', 45, 'Male', DEFAULT),
(240, 'Mark', 'Helfrich', 49, 'Male', DEFAULT),
(250, 'Nanette', 'Burstein', 52, 'Female', DEFAULT),
(260, 'James', 'McAvoy', 44, 'Male', DEFAULT),
(270, 'Mark', 'Waters', 58, 'Male', DEFAULT),
(280, 'Seth', 'Gordon', 46, 'Male', DEFAULT),
(290, 'Alex', 'Kendrick', 52, 'Male', DEFAULT),
(300, 'Kevin', 'Lima', 60, 'Male', DEFAULT),
(310, 'Lasse', 'Hallström', 76, 'Male', DEFAULT),
(320, 'Ewan', 'McGregor', 52, 'Male', DEFAULT),
(330, 'Rajkumar', 'Hirani', 60, 'Male', DEFAULT),
(340, 'Ashutosh', 'Gowariker', 59, 'Male', DEFAULT),
(350, 'Karan', 'Johar', 50, 'Male', DEFAULT),
(360, 'S.S', 'Rajamouli', 49, 'Male', DEFAULT),
(370, 'Sukumar', NULL, 53, 'Male', DEFAULT),
(380, 'Aditya', 'Chopra', 51, 'Male', DEFAULT),
(390, 'Umesh', 'Shukla', 52, 'Male', DEFAULT);

SET IDENTITY_INSERT mov.Movie_Director OFF;
```

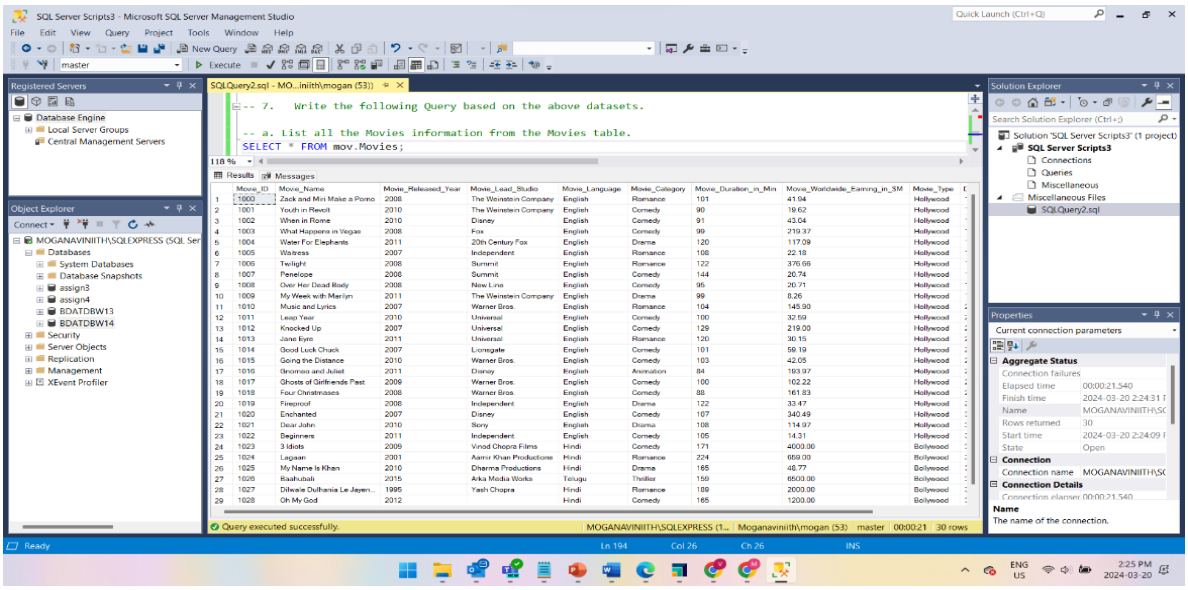


-- 4. Create a Movies table under mov schema with the following specifications-

```
CREATE TABLE mov.Movies (
    Movie_ID INT IDENTITY(1000,1) NOT NULL PRIMARY KEY CLUSTERED,
    Movie_Name VARCHAR(255),
    Movie_Released_Year INT,
    Movie_Lead_Studio VARCHAR(255),
    Movie_Language VARCHAR(255),
    Movie_Category VARCHAR(255),
    Movie_Duration_in_Min INT,
    Movie_Worldwide_Earning_in_$M DECIMAL(15,2),
    Movie_Type VARCHAR(255) NOT NULL CHECK (Movie_Type IN ('Hollywood', 'Bollywood')),
    Director_ID INT,
```

```
    CreatedOn DATE NOT NULL DEFAULT GETDATE(),
    FOREIGN KEY (Director_ID) REFERENCES mov.Movie_Director(Director_ID)
);

SET IDENTITY_INSERT mov.Movies ON;
INSERT INTO mov.Movies (Movie_ID, Movie_Name, Movie_Released_Year, Movie_Lead_Studio,
Movie_Language, Movie_Category, Movie_Duration_in_Min, Movie_Worldwide_Earning_in_$M,
Movie_Type, Director_ID) VALUES
(1000, 'Zack and Miri Make a Porno', 2008, 'The Weinstein Company', 'English', 'Romance',
101, 41.94, 'Hollywood', 100),
(1001, 'Youth in Revolt', 2010, 'The Weinstein Company', 'English', 'Comedy', 90, 19.62,
'Hollywood', 110),
(1002, 'When in Rome', 2010, 'Disney', 'English', 'Comedy', 91, 43.04, 'Hollywood', 120),
(1003, 'What Happens in Vegas', 2008, 'Fox', 'English', 'Comedy', 99, 219.37,
'Hollywood', 130),
(1004, 'Water For Elephants', 2011, '20th Century Fox', 'English', 'Drama', 120, 117.09,
'Hollywood', 140),
(1005, 'Waitress', 2007, 'Independent', 'English', 'Romance', 108, 22.18, 'Hollywood',
150),
(1006, 'Twilight', 2008, 'Summit', 'English', 'Romance', 122, 376.66, 'Hollywood', 160),
(1007, 'Penelope', 2008, 'Summit', 'English', 'Comedy', 144, 20.74, 'Hollywood', 170),
(1008, 'Over Her Dead Body', 2008, 'New Line', 'English', 'Comedy', 95, 20.71,
'Hollywood', 180),
(1009, 'My Week with Marilyn', 2011, 'The Weinstein Company', 'English', 'Drama', 99,
8.26, 'Hollywood', 190),
(1010, 'Music and Lyrics', 2007, 'Warner Bros.', 'English', 'Romance', 104, 145.9,
'Hollywood', 200),
(1011, 'Leap Year', 2010, 'Universal', 'English', 'Comedy', 100, 32.59, 'Hollywood',
210),
(1012, 'Knocked Up', 2007, 'Universal', 'English', 'Comedy', 129, 219, 'Hollywood', 220),
(1013, 'Jane Eyre', 2011, 'Universal', 'English', 'Romance', 120, 30.15, 'Hollywood',
230),
(1014, 'Good Luck Chuck', 2007, 'Lionsgate', 'English', 'Comedy', 101, 59.19,
'Hollywood', 240),
(1015, 'Going the Distance', 2010, 'Warner Bros.', 'English', 'Comedy', 103, 42.05,
'Hollywood', 250),
(1016, 'Gnomeo and Juliet', 2011, 'Disney', 'English', 'Animation', 84, 193.97,
'Hollywood', 260),
(1017, 'Ghosts of Girlfriends Past', 2009, 'Warner Bros.', 'English', 'Comedy', 100,
102.22, 'Hollywood', 270),
(1018, 'Four Christmases', 2008, 'Warner Bros.', 'English', 'Comedy', 88, 161.83,
'Hollywood', 280),
(1019, 'Fireproof', 2008, 'Independent', 'English', 'Drama', 122, 33.47, 'Hollywood',
290),
(1020, 'Enchanted', 2007, 'Disney', 'English', 'Comedy', 107, 340.49, 'Hollywood', 300),
(1021, 'Dear John', 2010, 'Sony', 'English', 'Drama', 108, 114.97, 'Hollywood', 310),
(1022, 'Beginners', 2011, 'Independent', 'English', 'Comedy', 105, 14.31, 'Hollywood',
320),
(1023, '3 Idiots', 2009, 'Vinod Chopra Films', 'Hindi', 'Comedy', 171, 4000, 'Bollywood',
330),
(1024, 'Lagaan', 2001, 'Aamir Khan Productions', 'Hindi', 'Romance', 224, 659,
'Bollywood', 340),
(1025, 'My Name Is Khan', 2010, 'Dharma Productions', 'Hindi', 'Drama', 165, 48.77,
'Bollywood', 350),
(1026, 'Baahubali', 2015, 'Arka Media Works', 'Telugu', 'Thriller', 159, 6500,
'Bollywood', 360),
(1027, 'Dilwale Dulhania Le Jayenge', 1995, 'Yash Chopra', 'Hindi', 'Romance', 189, 2000,
'Bollywood', 380),
```

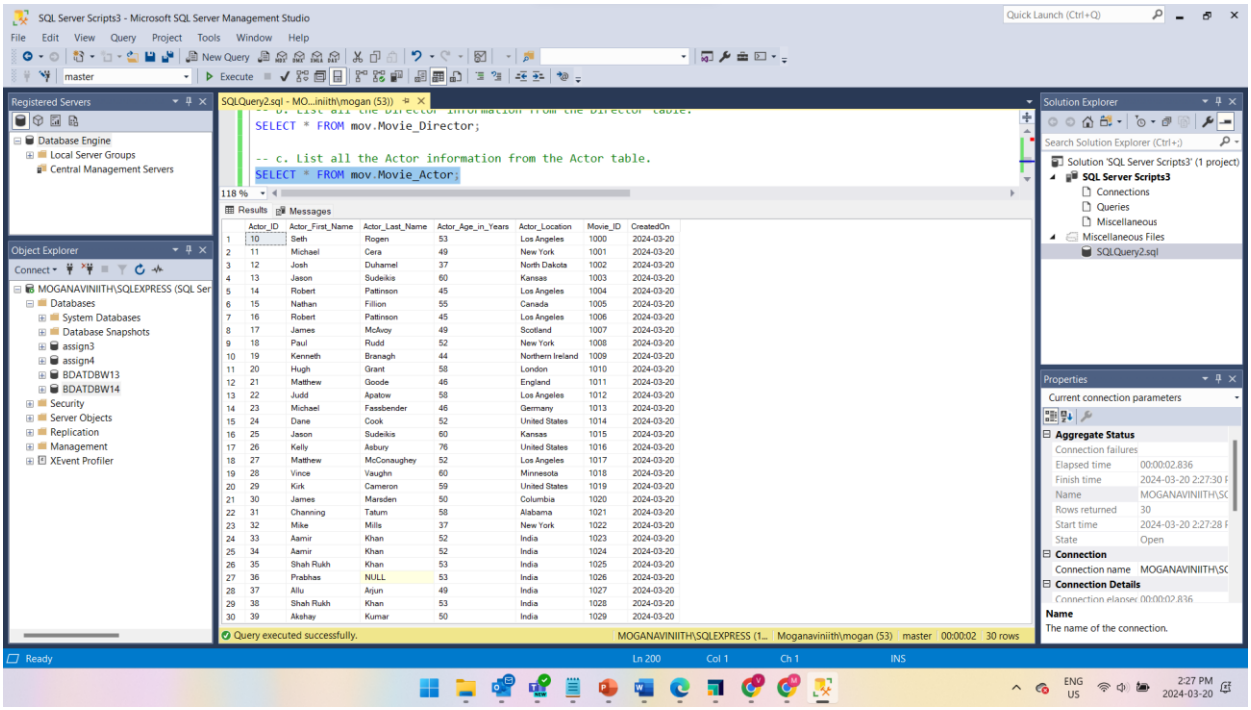
```
(1028, 'Oh My God', 2012, '', 'Hindi', 'Comedy', 165, 1200, 'Bollywood', 390),
(1029, 'Pushpa', 2021, 'Mythri Movie Makers', 'Telugu', 'Drama', 179, 3730, 'Bollywood',
370);
SET IDENTITY_INSERT mov.Movies OFF;
```



5.Create a Movie_Actor table under mov schema with the following specifications

```
CREATE TABLE mov.Movie_Actor (
    Actor_ID INT IDENTITY(10,1) NOT NULL PRIMARY KEY CLUSTERED,
    Actor_First_Name VARCHAR(255),
    Actor_Last_Name VARCHAR(255),
    Actor_Age_in_Years INT,
    Actor_Location VARCHAR(255),
    Movie_ID INT,
    CreatedOn DATE NOT NULL DEFAULT GETDATE(),
    CONSTRAINT fk_Movie_ID FOREIGN KEY (Movie_ID) REFERENCES mov.Movies(Movie_ID)
);
SET IDENTITY_INSERT mov.Movie_Actor ON;
INSERT INTO mov.Movie_Actor (Actor_ID, Actor_First_Name, Actor_Last_Name,
Actor_Age_in_Years, Actor_Location, Movie_ID) VALUES
(10, 'Seth', 'Rogen', 53, 'Los Angeles', 1000),
(11, 'Michael', 'Cera', 49, 'New York', 1001),
(12, 'Josh', 'Duhamel', 37, 'North Dakota', 1002),
(13, 'Jason', 'Sudeikis', 60, 'Kansas', 1003),
(14, 'Robert', 'Pattinson', 45, 'Los Angeles', 1004),
(15, 'Nathan', 'Fillion', 55, 'Canada', 1005),
(16, 'Robert', 'Pattinson', 45, 'Los Angeles', 1006),
(17, 'James', 'McAvoy', 49, 'Scotland', 1007),
(18, 'Paul', 'Rudd', 52, 'New York', 1008),
(19, 'Kenneth', 'Branagh', 44, 'Northern Ireland', 1009),
(20, 'Hugh', 'Grant', 58, 'London', 1010),
(21, 'Matthew', 'Goode', 46, 'England', 1011),
(22, 'Judd', 'Apatow', 58, 'Los Angeles', 1012),
(23, 'Michael', 'Fassbender', 46, 'Germany', 1013),
```

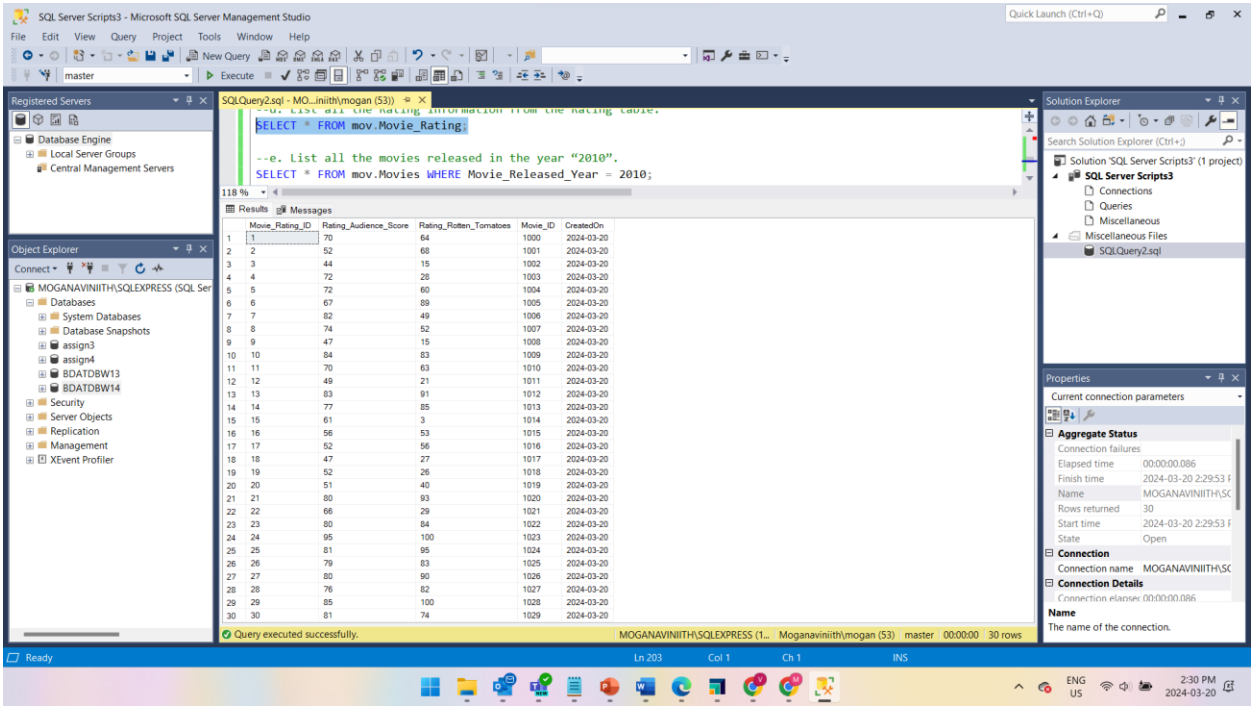
```
(24, 'Dane', 'Cook', 52, 'United States', 1014),
(25, 'Jason', 'Sudeikis', 60, 'Kansas', 1015),
(26, 'Kelly', 'Asbury', 76, 'United States', 1016),
(27, 'Matthew', 'McConaughey', 52, 'Los Angeles', 1017),
(28, 'Vince', 'Vaughn', 60, 'Minnesota', 1018),
(29, 'Kirk', 'Cameron', 59, 'United States', 1019),
(30, 'James', 'Marsden', 50, 'Columbia', 1020),
(31, 'Channing', 'Tatum', 58, 'Alabama', 1021),
(32, 'Mike', 'Mills', 37, 'New York', 1022),
(33, 'Aamir', 'Khan', 52, 'India', 1023),
(34, 'Aamir', 'Khan', 52, 'India', 1024),
(35, 'Shah Rukh', 'Khan', 53, 'India', 1025),
(36, 'Prabhas', NULL, 53, 'India', 1026),
(37, 'Allu', 'Arjun', 49, 'India', 1027),
(38, 'Shah Rukh', 'Khan', 53, 'India', 1028),
(39, 'Akshay', 'Kumar', 50, 'India', 1029);

SET IDENTITY_INSERT mov.Movie_Actor OFF;
```



6.Create a Movie_Rating table under mov schema with the following specifications

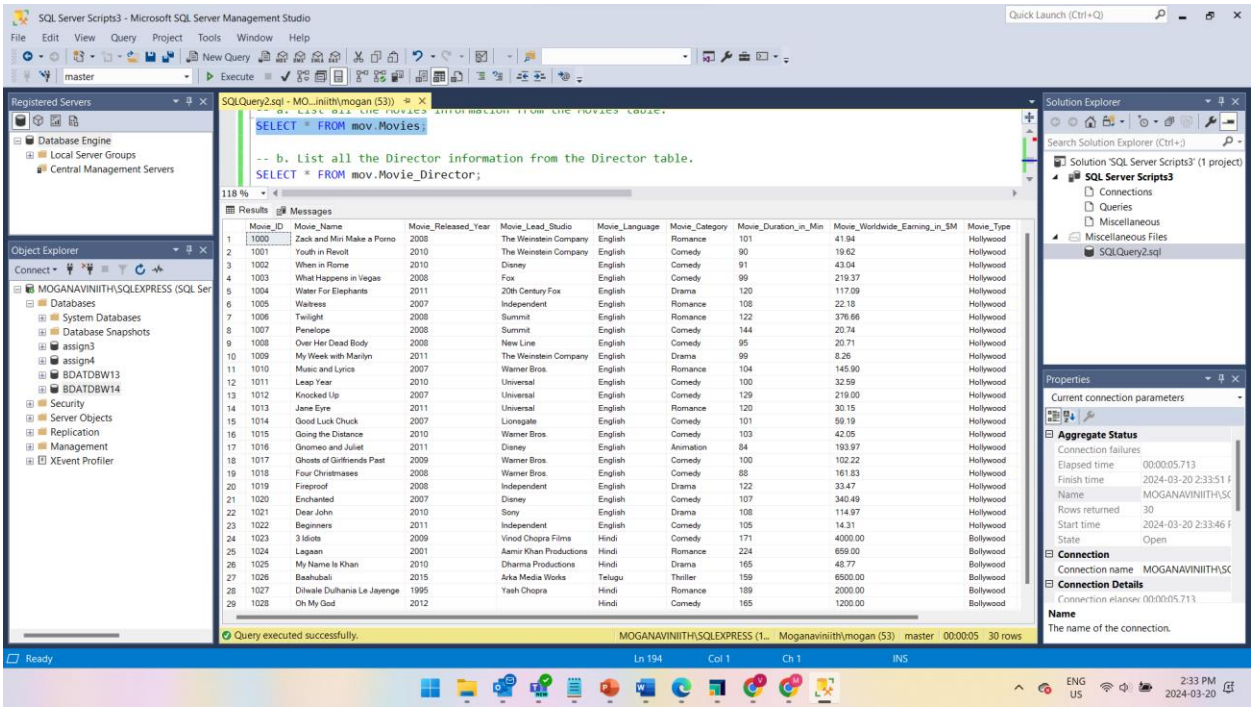
```
CREATE TABLE mov.Movie_Rating (
    Movie_Rating_ID INT IDENTITY(1,1) NOT NULL PRIMARY KEY CLUSTERED,
    Rating_Audience_Score VARCHAR(255),
    Rating_Rotten_Tomatoes VARCHAR(255),
    Movie_ID INT,
    CreatedOn DATE NOT NULL DEFAULT GETDATE(),
    CONSTRAINT fk_MovieRating_Movie_ID FOREIGN KEY (Movie_ID) REFERENCES
mov.Movies(Movie_ID));
INSERT INTO mov.Movie_Rating (Rating_Audience_Score, Rating_Rotten_Tomatoes, Movie_ID)
VALUES
(70, 64, 1000),
```

```
(52, 68, 1001),
(44, 15, 1002),
(72, 28, 1003),
(72, 60, 1004),
(67, 89, 1005),
(82, 49, 1006),
(74, 52, 1007),
(47, 15, 1008),
(84, 83, 1009),
(70, 63, 1010),
(49, 21, 1011),
(83, 91, 1012),
(77, 85, 1013),
(61, 3, 1014),
(56, 53, 1015),
(52, 56, 1016),
(47, 27, 1017),
(52, 26, 1018),
(51, 40, 1019),
(80, 93, 1020),
(66, 29, 1021),
(80, 84, 1022),
(95, 100, 1023),
(81, 95, 1024),
(79, 83, 1025),
(80, 90, 1026),
(76, 82, 1027),
(85, 100, 1028),
(81, 74, 1029);
```

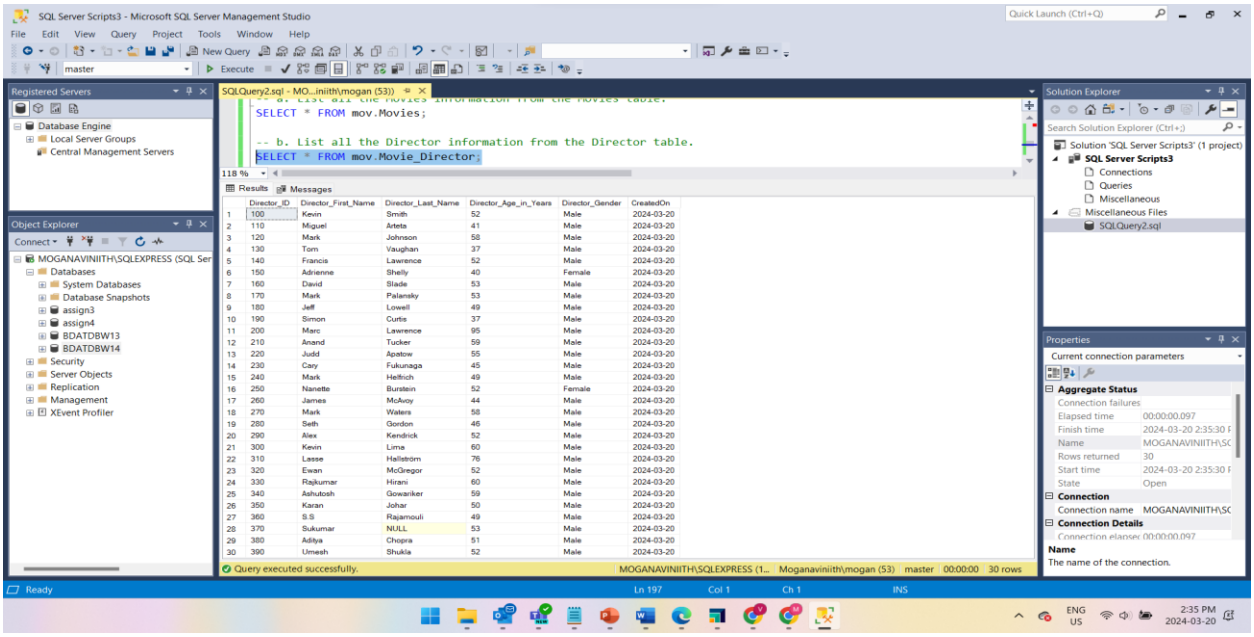


7.Write the following Query based on the above datasets.

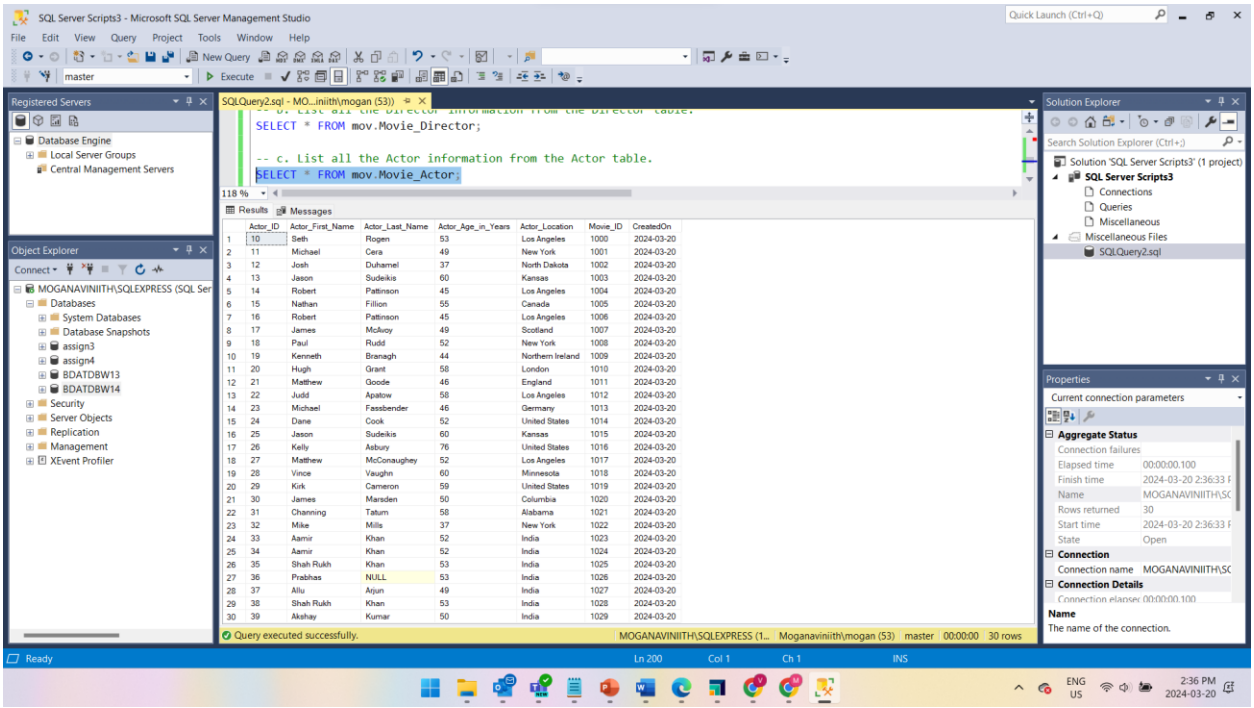
a. List all the Movies information from the Movies table.



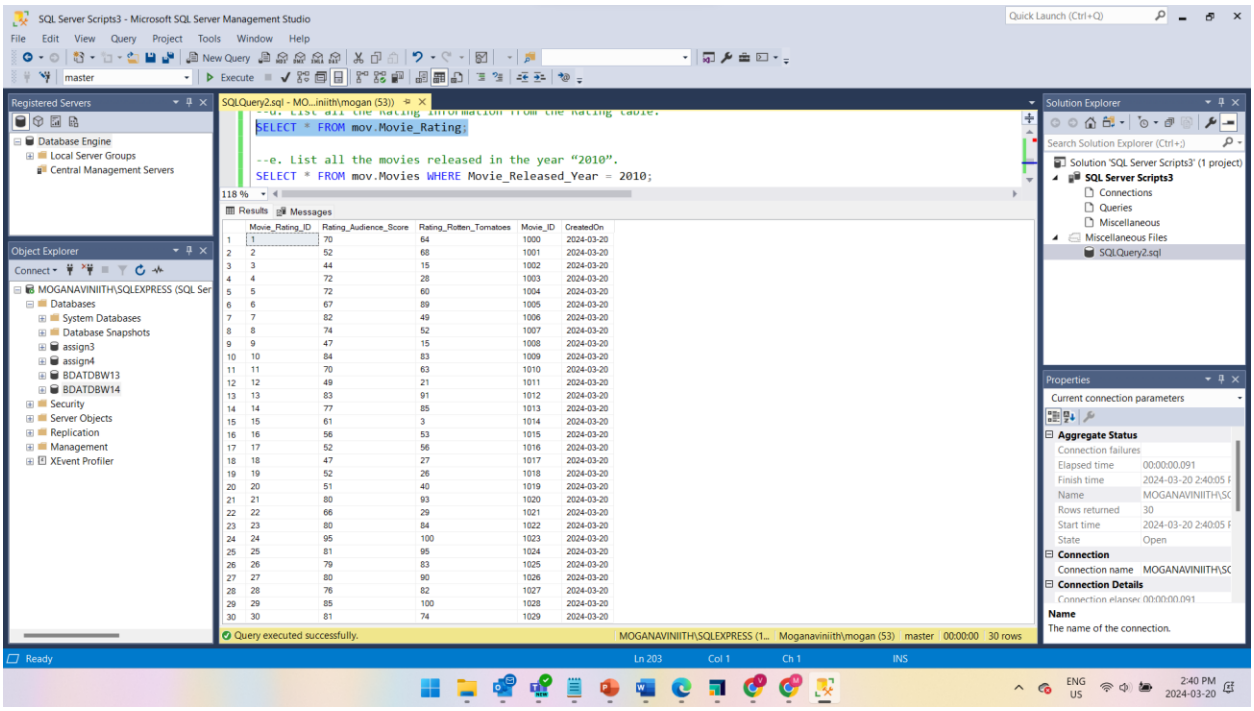
b. List all the Director information from the Director table.



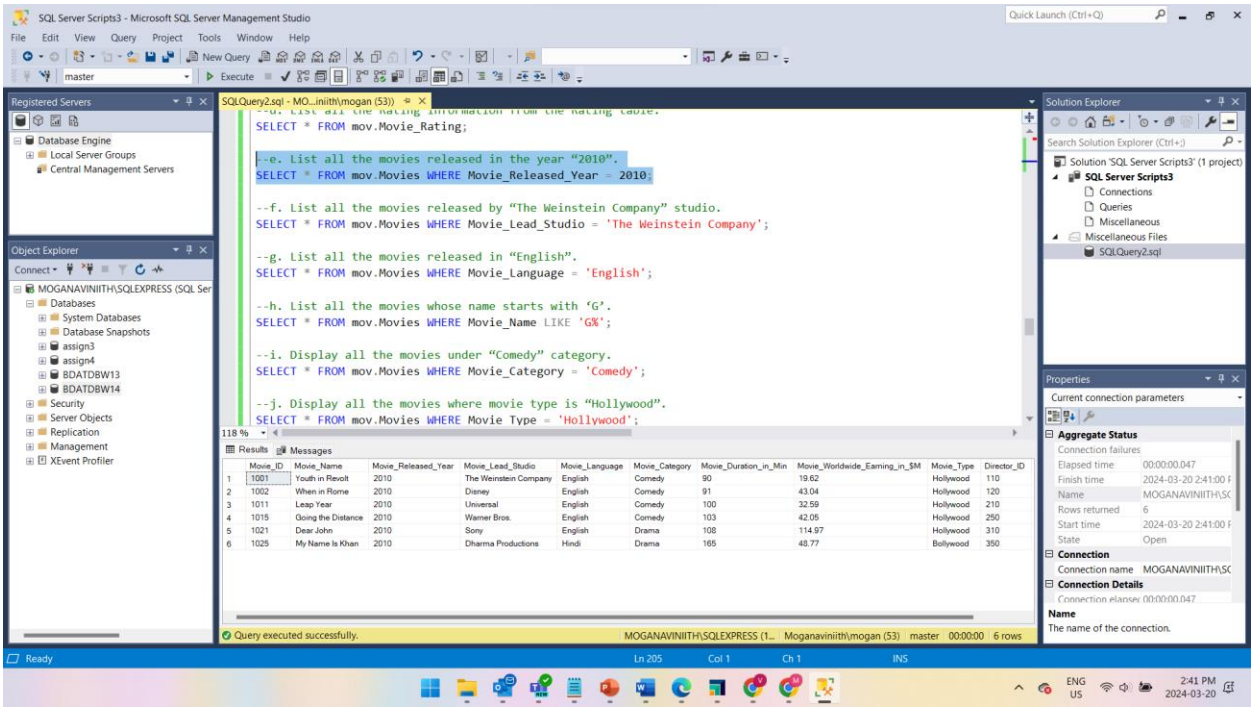
c. List all the Actor information from the Actor table.



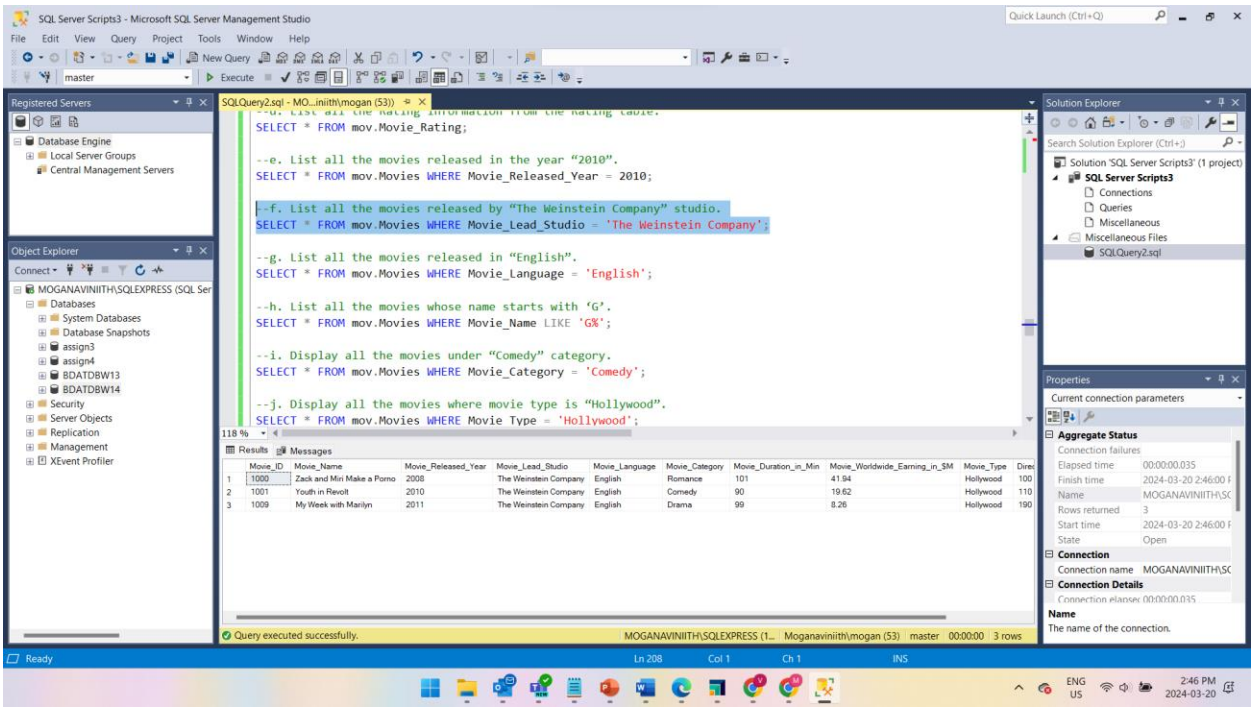
d. List all the Rating information from the Rating table.



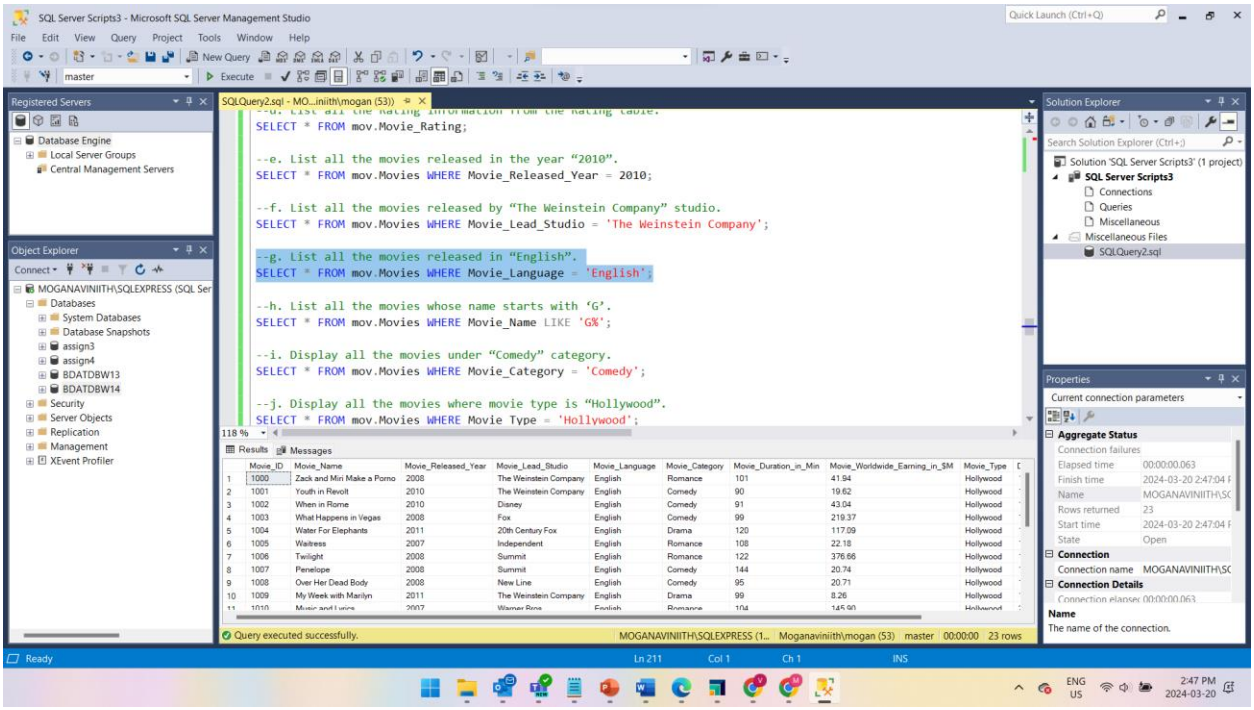
e. List all the movie released in year "2010".



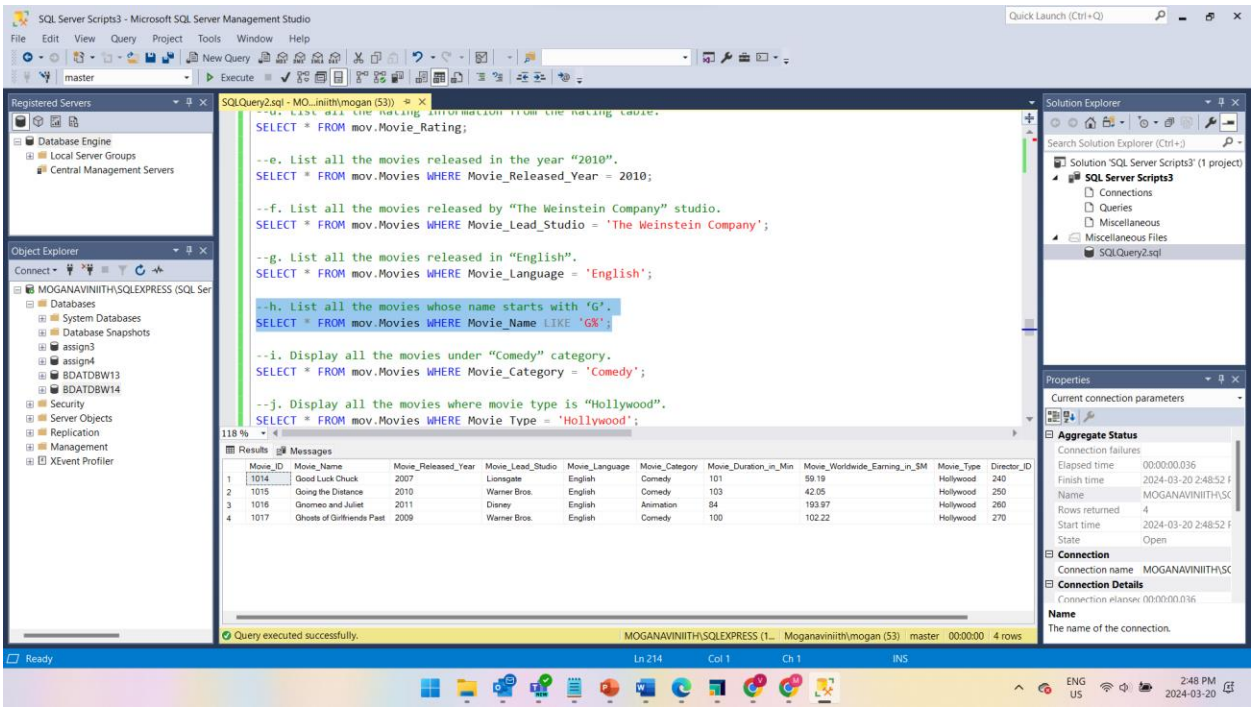
f. List all the movie released by "The Weinstein Company" studio.



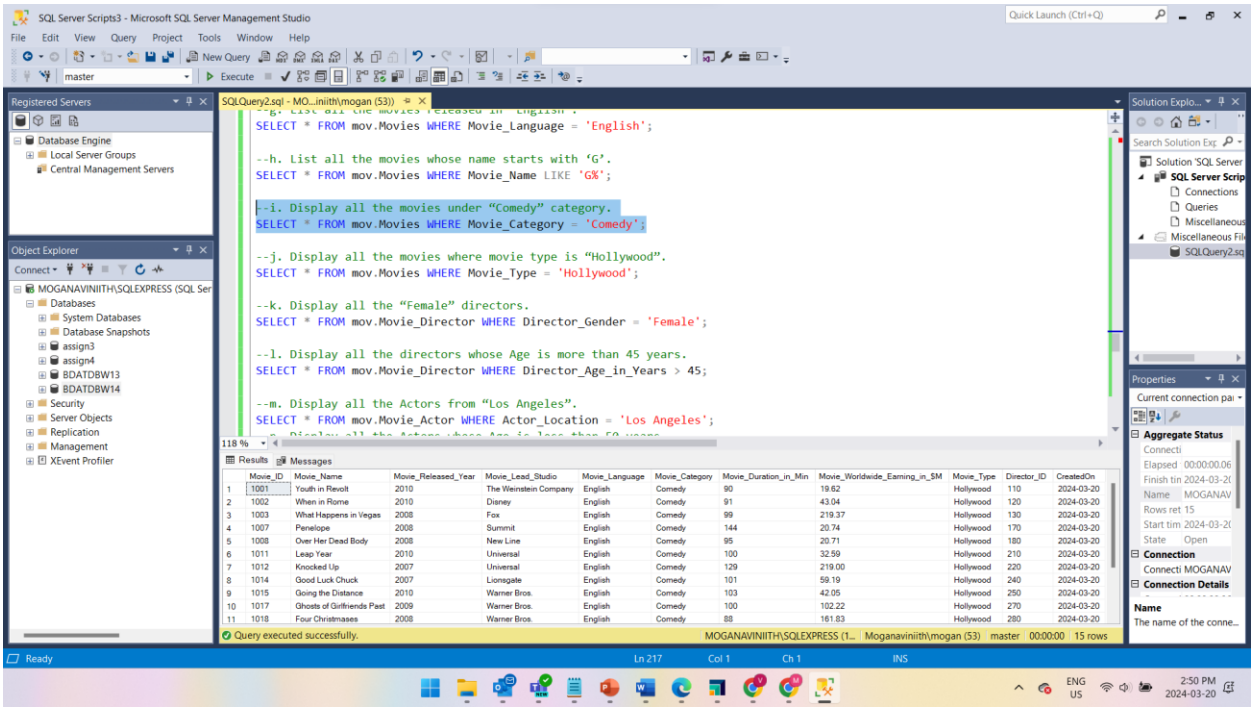
g. List all the movie released in "English".



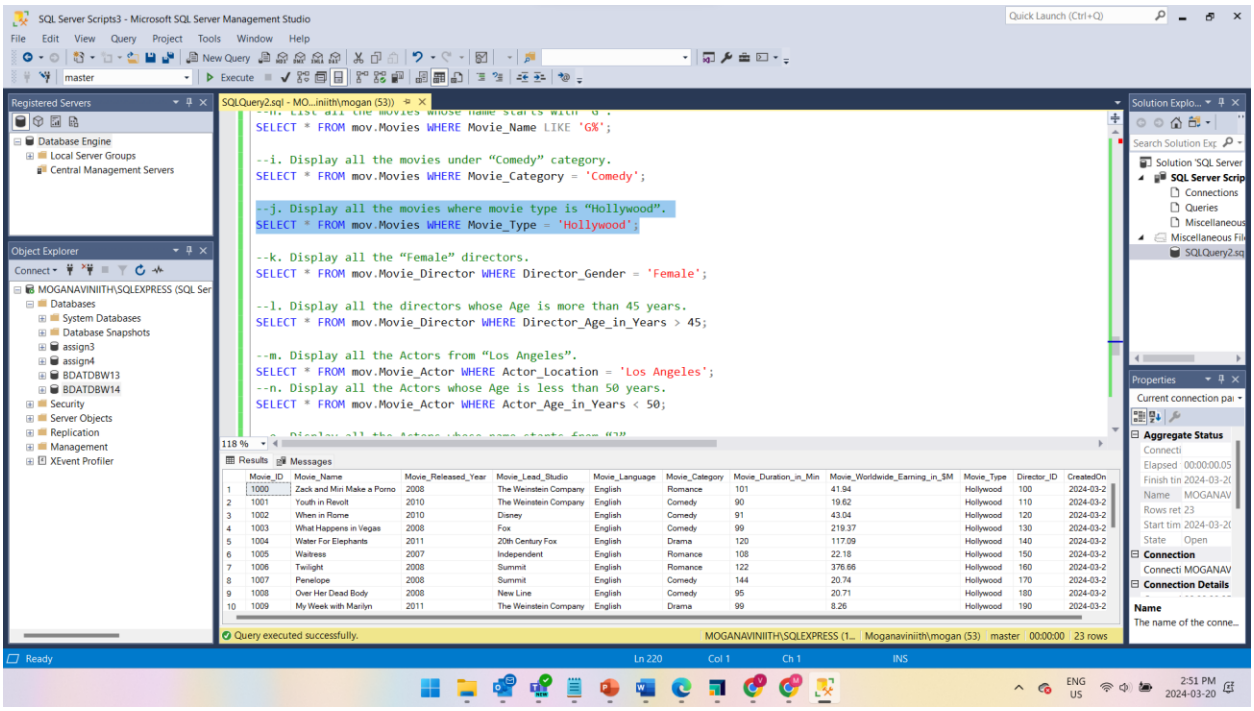
h. List all the movie whose name starts with 'G'.



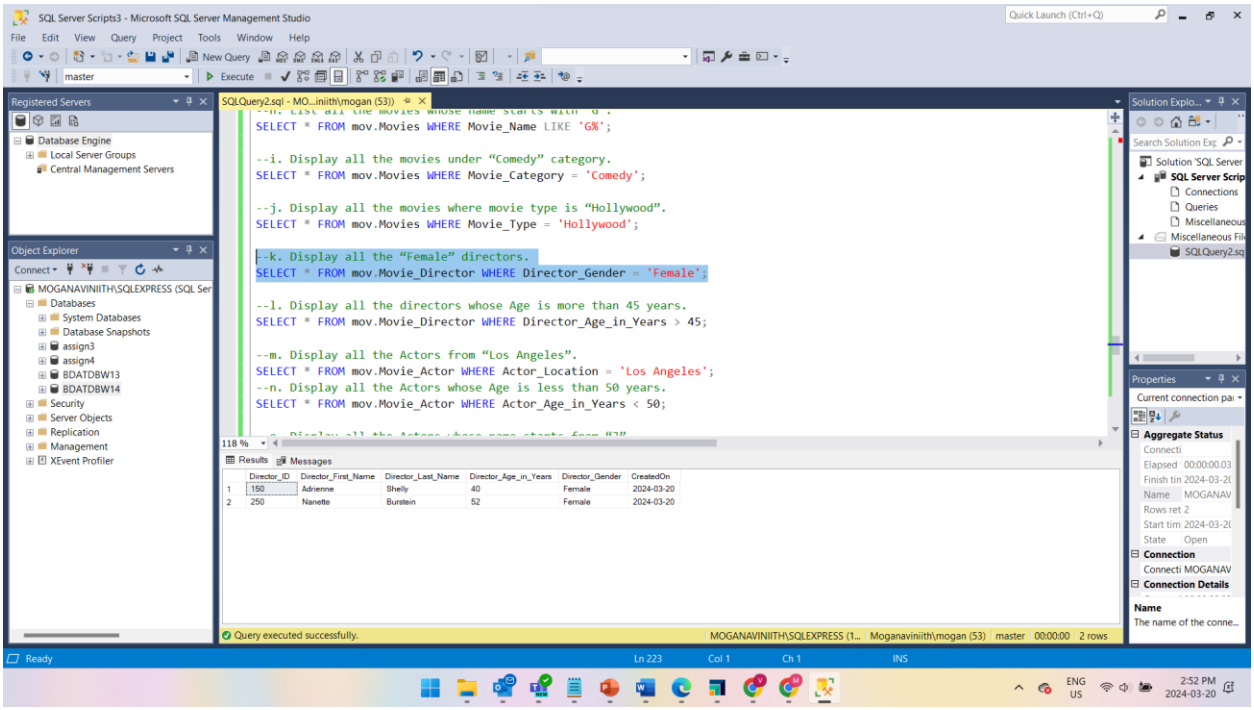
i. Display all the movie under "Comedy" category.



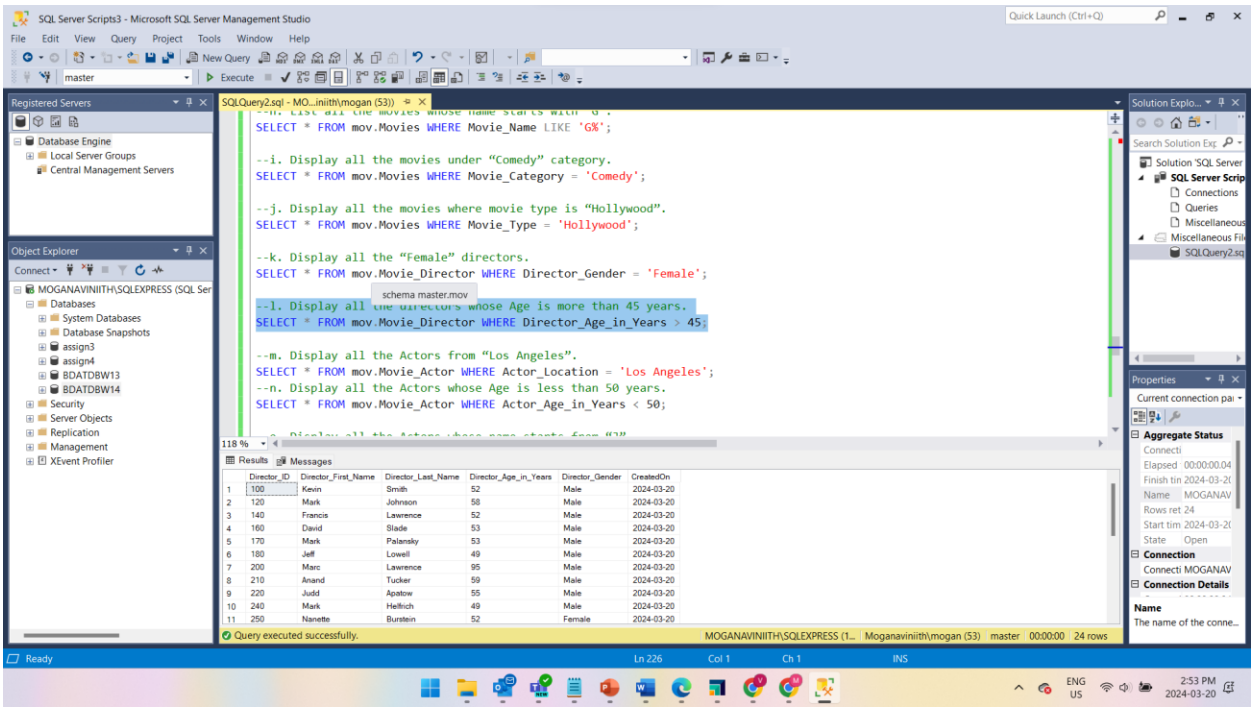
j. Display all the movie where movie type is "Hollywood".



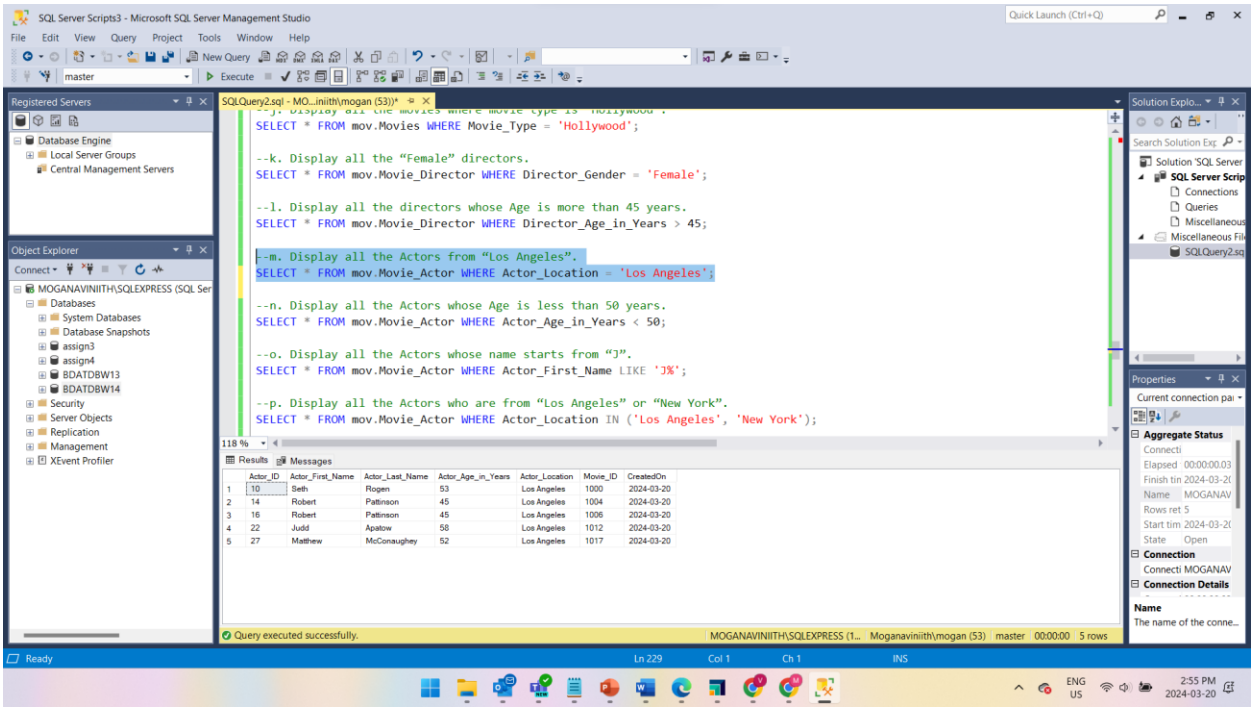
k. Display all the "Female" directors.



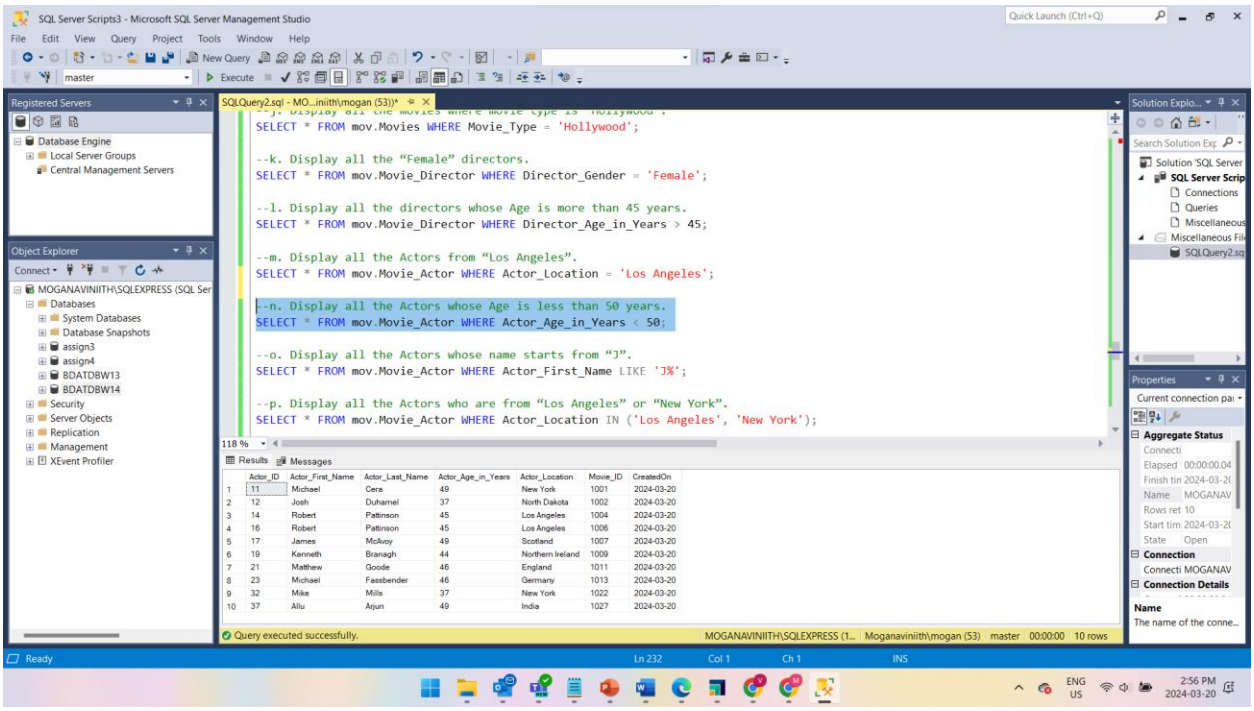
l. Display all the director whose Age is more than 45 years.



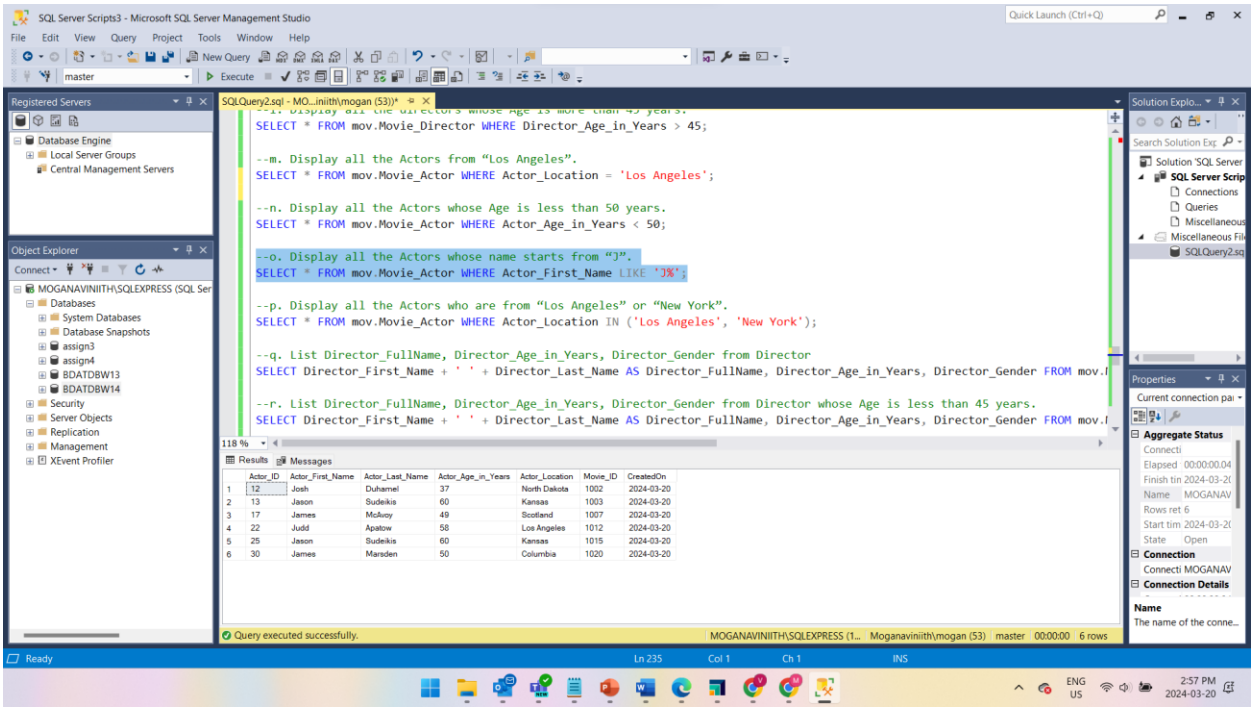
m. Display all the Actors from "Los Angeles".



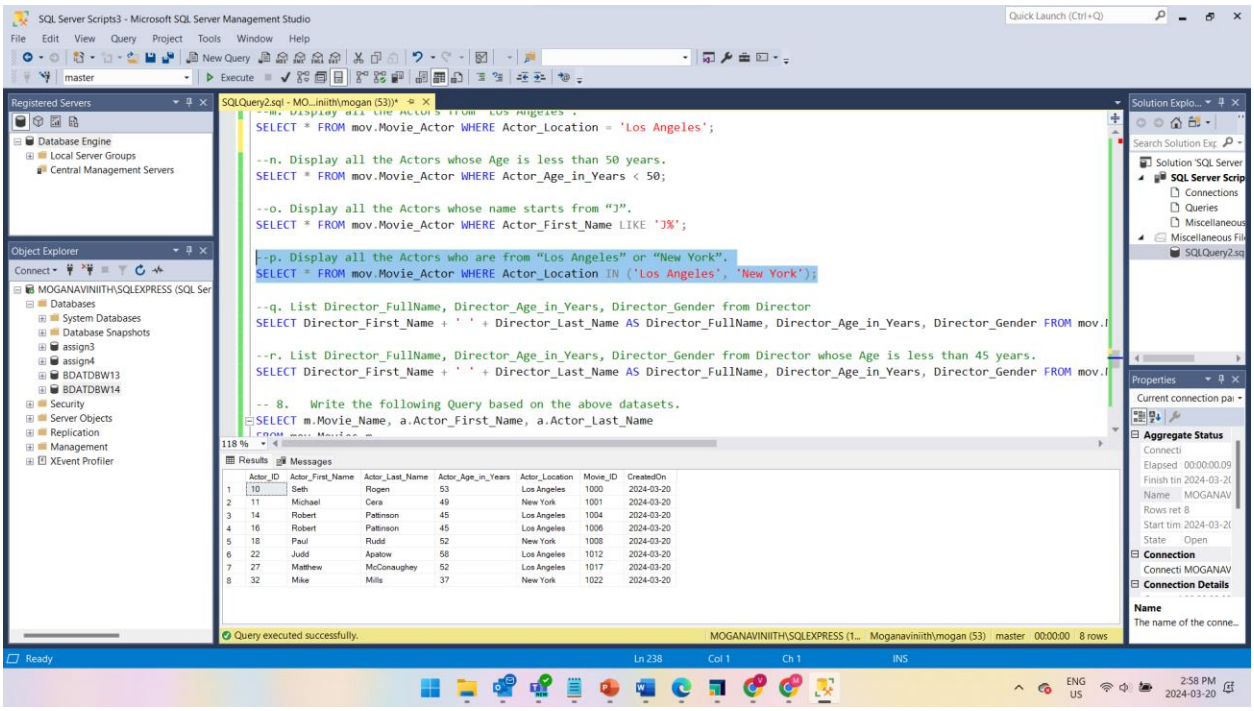
n. Display all the Actor whose Age is less than 50 years.



o. Display all the Actor whose name is starts from "J".

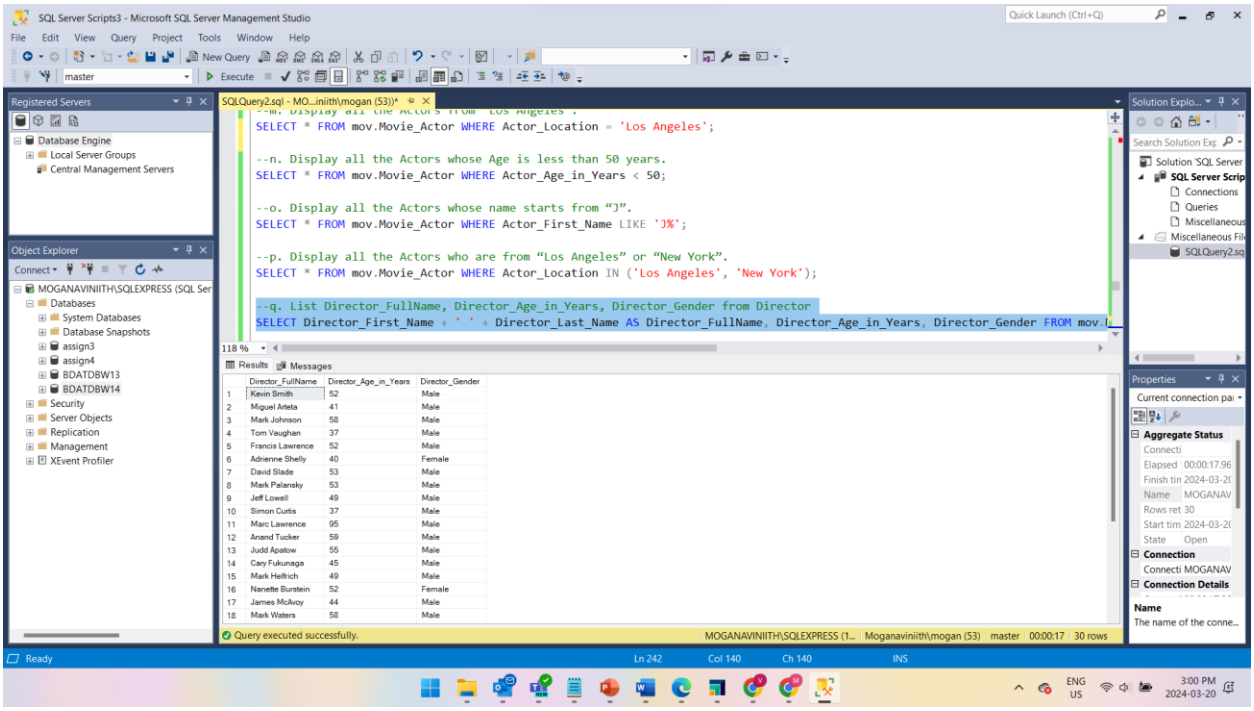


p. Display all the Actor who is from "Los Angeles" or "New York".

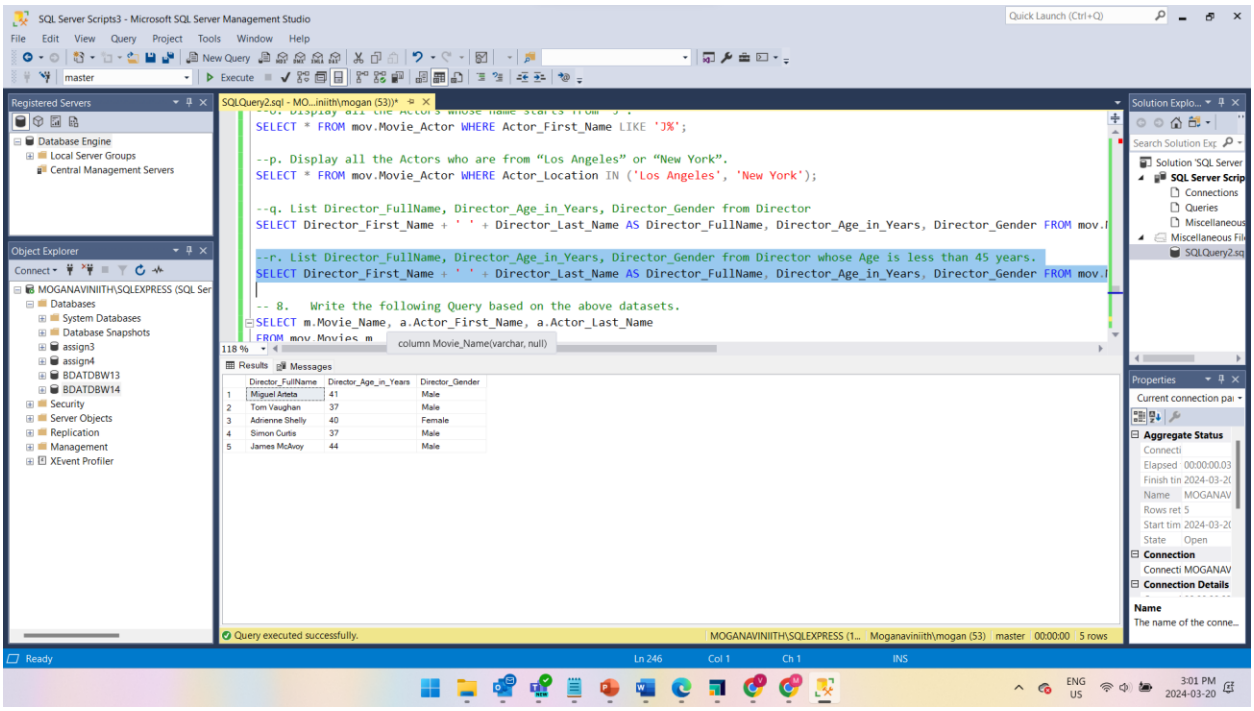


q. List Director_FullName, Director_Age_in_Years, Director_Gender from Director

[Director_FullName =Director_First_Name + " " + Director_Last_Name]

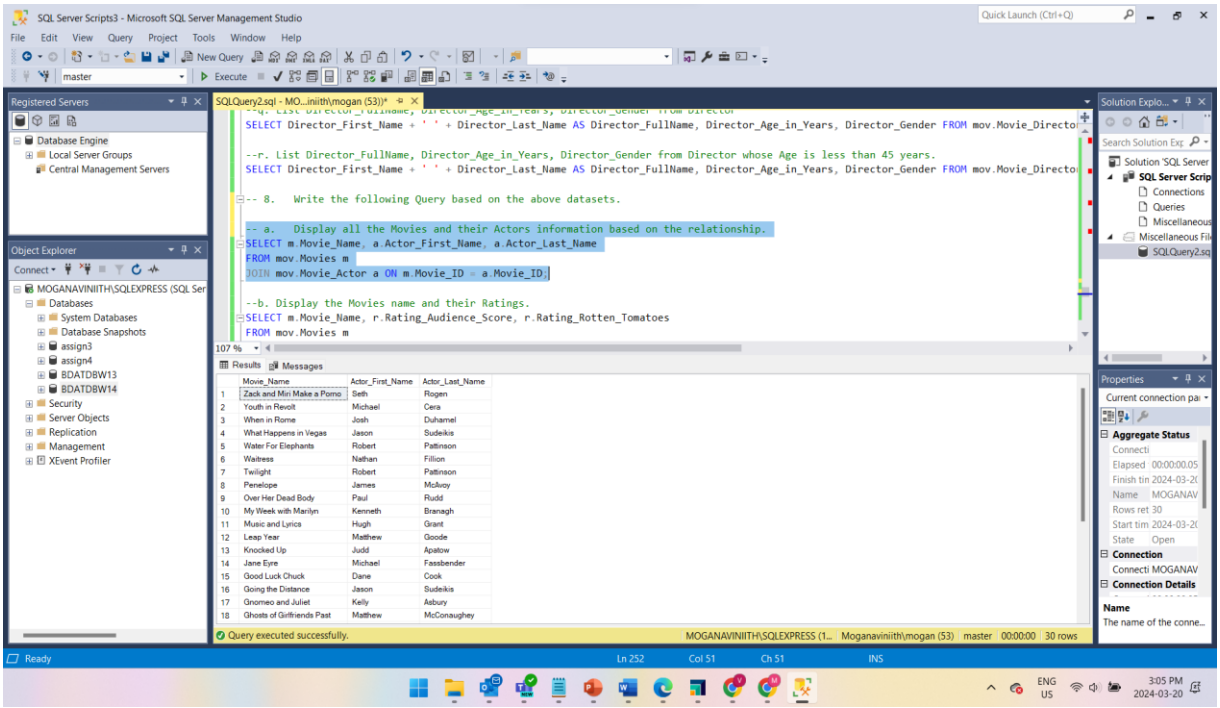


r. List Director_FullName, Director_Age_in_Years, Director_Gender from Director whose Age is less than 45 years. [Director_FullName =Director_First_Name + " " + Director_Last_Name]

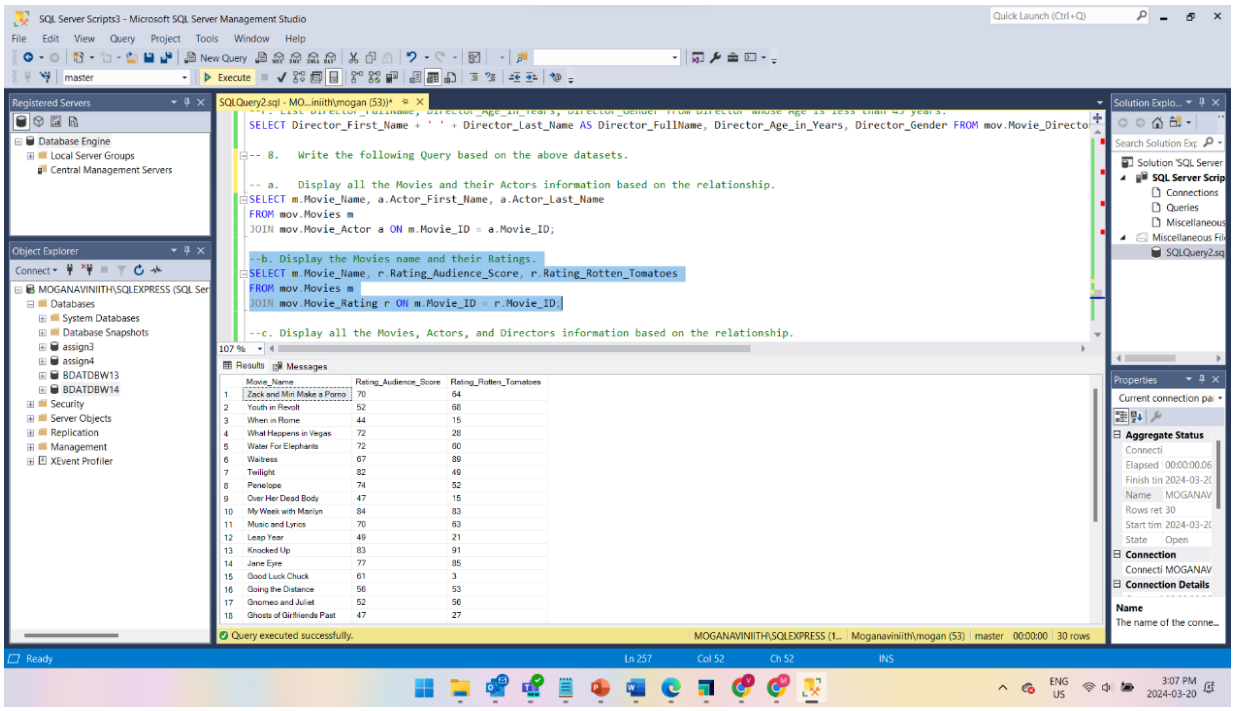


8.Write the following Query based on the above datasets.

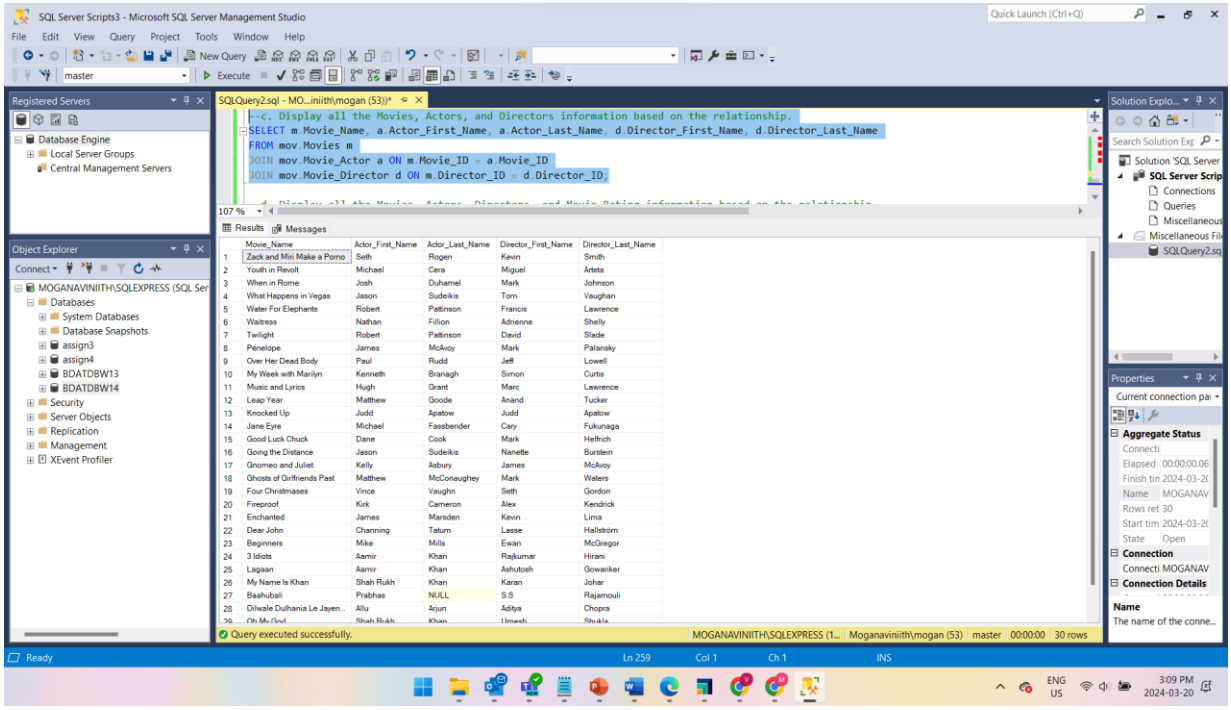
a. Display all the Movies and their Actors information based on the relationship.



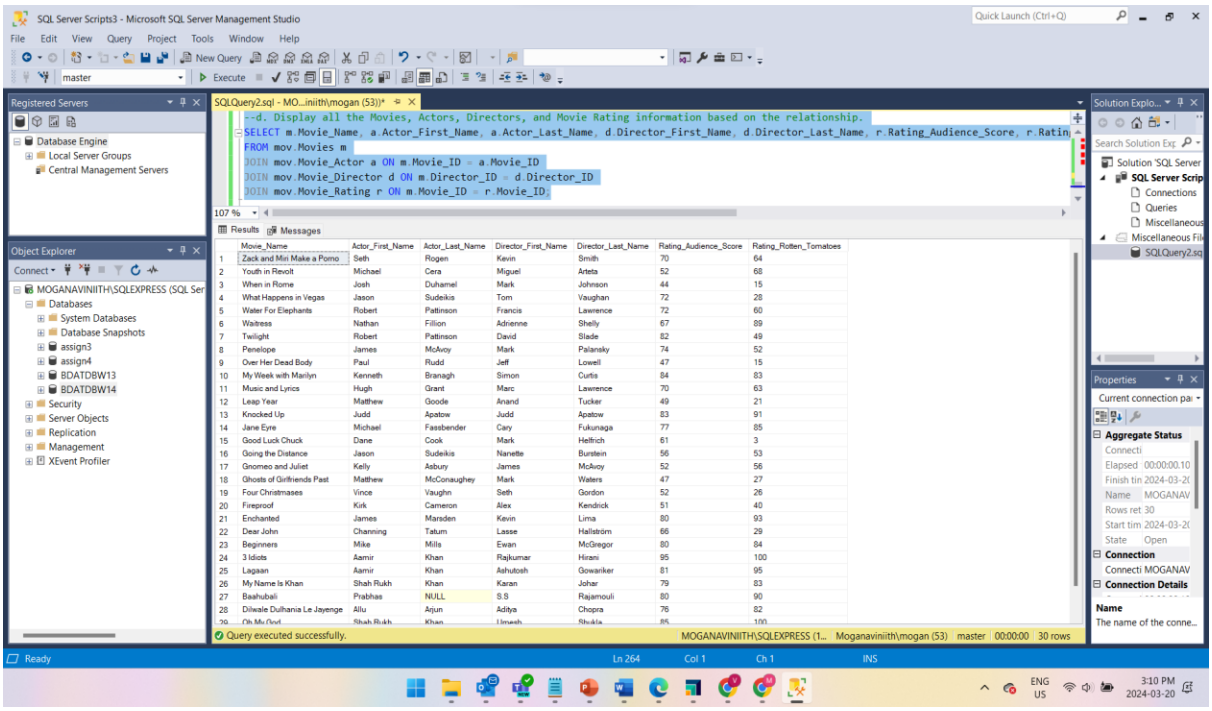
b. Display the Movies name and their Ratings.



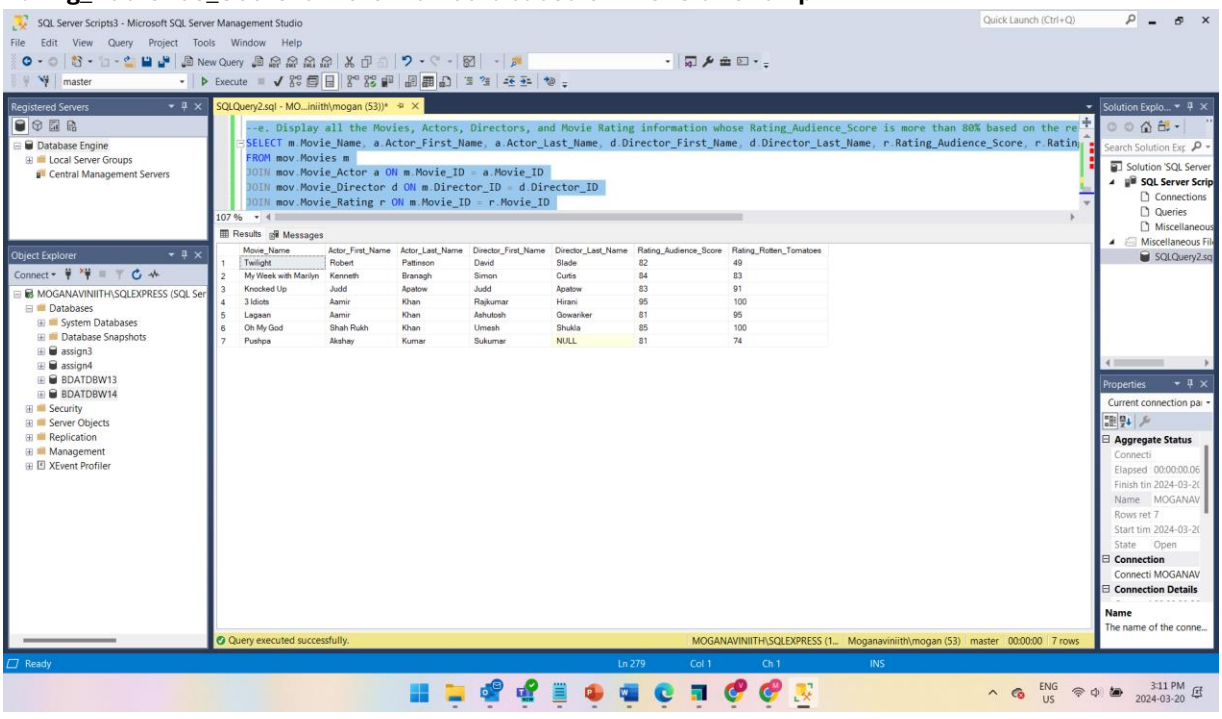
c. Display all the Movies, Actors, and Directors information based on the relationship.



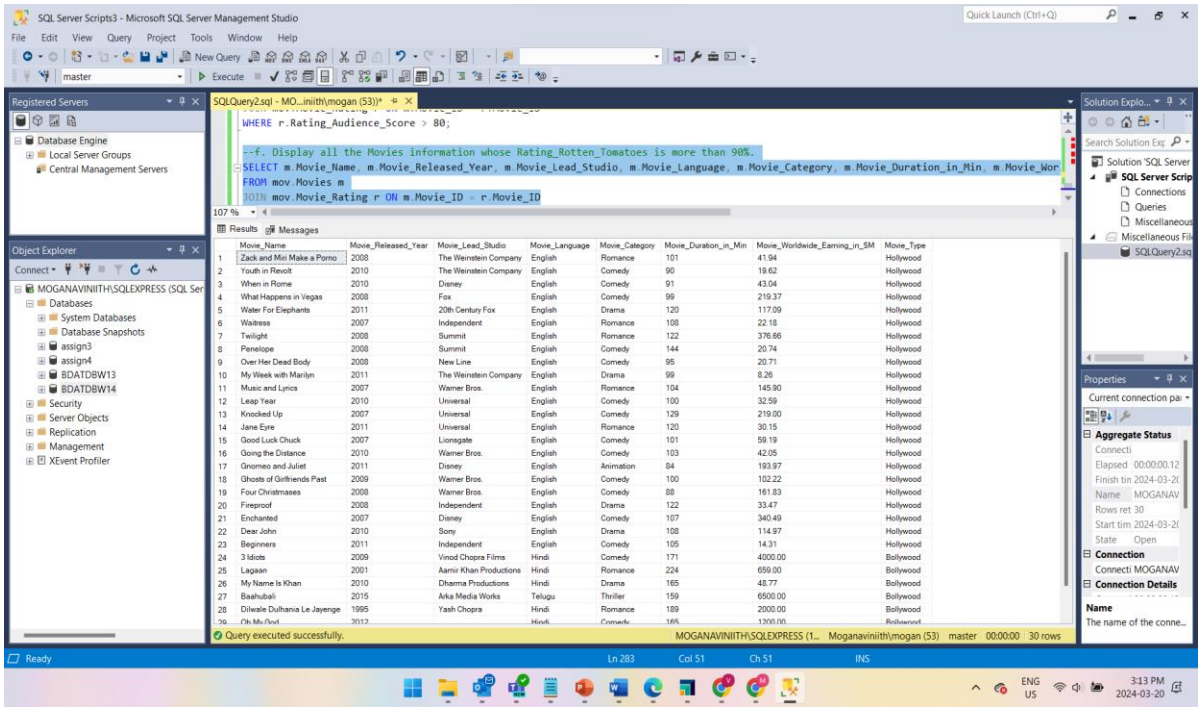
d. Display all the Movies, Actors, Directors, and Movie Rating information based on the relationship.



e. **Display all the Movies, Actors, Directors, and Movie Rating information whose Rating_Audience_Score is more than 80% based on the relationship.**



f. **Display all the Movies information whose Rating_Rotten_Tomatoes is more than 90%.**



9. Write the following Query based on the above datasets.

a. Create new table MovieCopy and copy all records of Movie table.

```
SELECT *
INTO mov.MovieCopy
FROM mov.Movies;
```

b. Create a new table DirectorCopy and copy only the schema of director table.

```
SELECT *
INTO mov.DirectorCopy
FROM mov.Movie_Director
WHERE 1=0; -- No data is copied, only the schema
```

c. Create new table ActorCopy and copy all records of Actor table.

```
SELECT *
INTO mov.ActorCopy
FROM mov.Movie_Actor;
```

d. Create new table RatingCopy and copy all records of Rating table.

```
SELECT *
INTO mov.RatingCopy
FROM mov.Movie_Rating;
```

e. Create new table RatingCopies and copy only the schema from Rating table.

```
SELECT *
INTO mov.RatingCopies
FROM mov.Movie_Rating
WHERE 1=0; -- No data is copied, only the schema
```

10.Write the following Query based on the above datasets.

a. Delete all the records from the RatingCopy table.

```
DELETE FROM mov.RatingCopy;
```

b. Delete all the movies from MovieCopy whose released in year "2010".

```
DELETE FROM mov.MovieCopy
WHERE Movie_Released_Year = 2010;
```

c. Delete all the movies from MovieCopy where language is 'Hindi'.

```
DELETE FROM mov.MovieCopy
WHERE Movie_Language = 'Hindi';
```

d. Delete all the movies from MovieCopy where Rating_Audience_Score is less than 80%.

```
DELETE m
FROM mov.MovieCopy m
JOIN mov.RatingCopy r ON m.Movie_ID = r.Movie_ID
WHERE r.Rating_Audience_Score < 80;
```

e. Delete all the movies from MovieCopy where Rating_Rotten_Tomatoes is less than 70%.

```
DELETE m
FROM mov.MovieCopy m
JOIN mov.RatingCopy r ON m.Movie_ID = r.Movie_ID
WHERE r.Rating_Rotten_Tomatoes < 70;
```

12.Write the following Query based on the above datasets.

a. Create a view to display all the movie information.

```
CREATE VIEW mov.View_AllMovies AS
SELECT *
FROM mov.Movies;
```

b. Create a view to display all the movies and their rating information.

```
CREATE VIEW mov.View_MoviesWithRatings AS
SELECT m.Movie_Name, m.Movie_Released_Year, m.Movie_Lead_Studio, r.Rating_Audience_Score,
r.Rating_Rotten_Tomatoes
FROM mov.Movies m
JOIN mov.Movie_Rating r ON m.Movie_ID = r.Movie_ID;
```

c. Create a view to display all the movies and their actor information.

```
CREATE VIEW mov.View_MoviesWithActors AS
SELECT m.Movie_Name, a.Actor_First_Name, a.Actor_Last_Name, a.Actor_Age_in_Years,
a.Actor_Location
FROM mov.Movies m
JOIN mov.Movie_Actor a ON m.Movie_ID = a.Movie_ID;
```

d. Create a view to display all the movies, rating, actor along with director information.

```
CREATE VIEW mov.View_MoviesRatingsActorsDirectors AS
SELECT m.Movie_Name, r.Rating_Audience_Score, r.Rating_Rotten_Tomatoes,
a.Actor_First_Name, a.Actor_Last_Name, d.Director_First_Name, d.Director_Last_Name
FROM mov.Movies m
JOIN mov.Movie_Rating r ON m.Movie_ID = r.Movie_ID
JOIN mov.Movie_Actor a ON m.Movie_ID = a.Movie_ID
JOIN mov.Movie_Director d ON m.Director_ID = d.Director_ID;
```

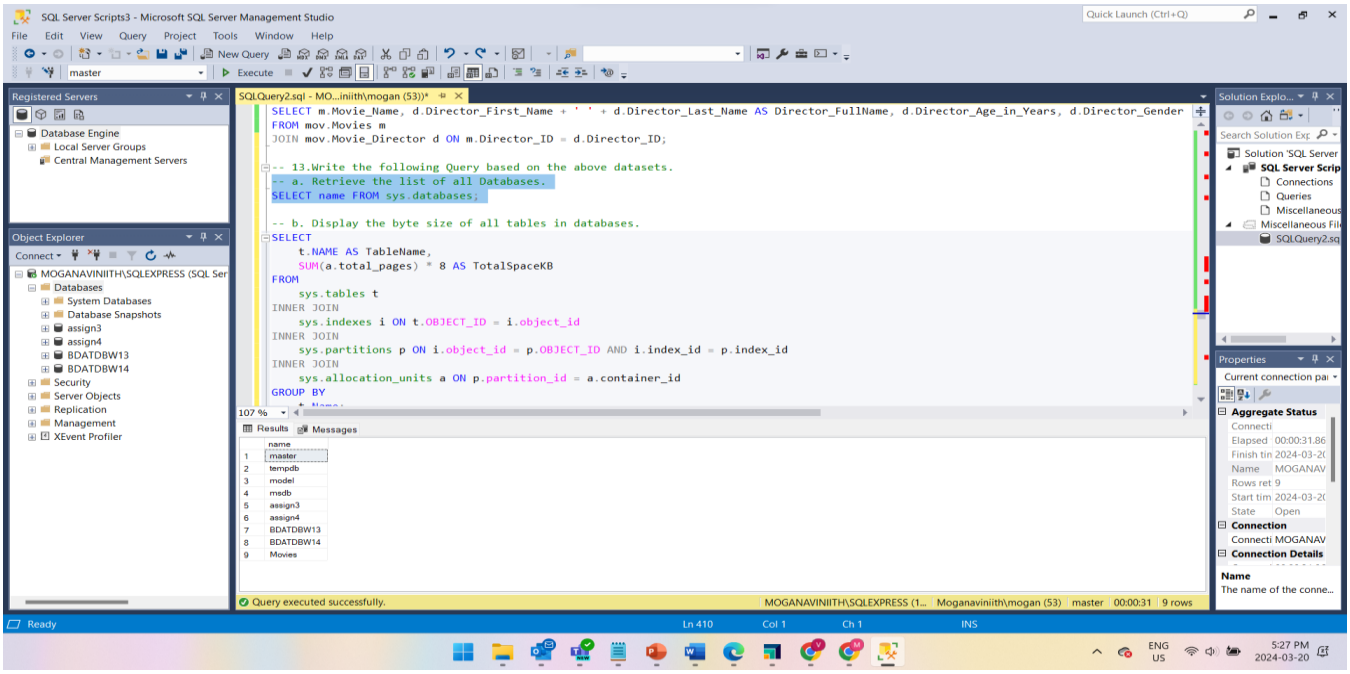
e. Create a view to display all the information based on the result set returned by the specified query.

```
CREATE VIEW mov.View_MoviesDirectors AS
```

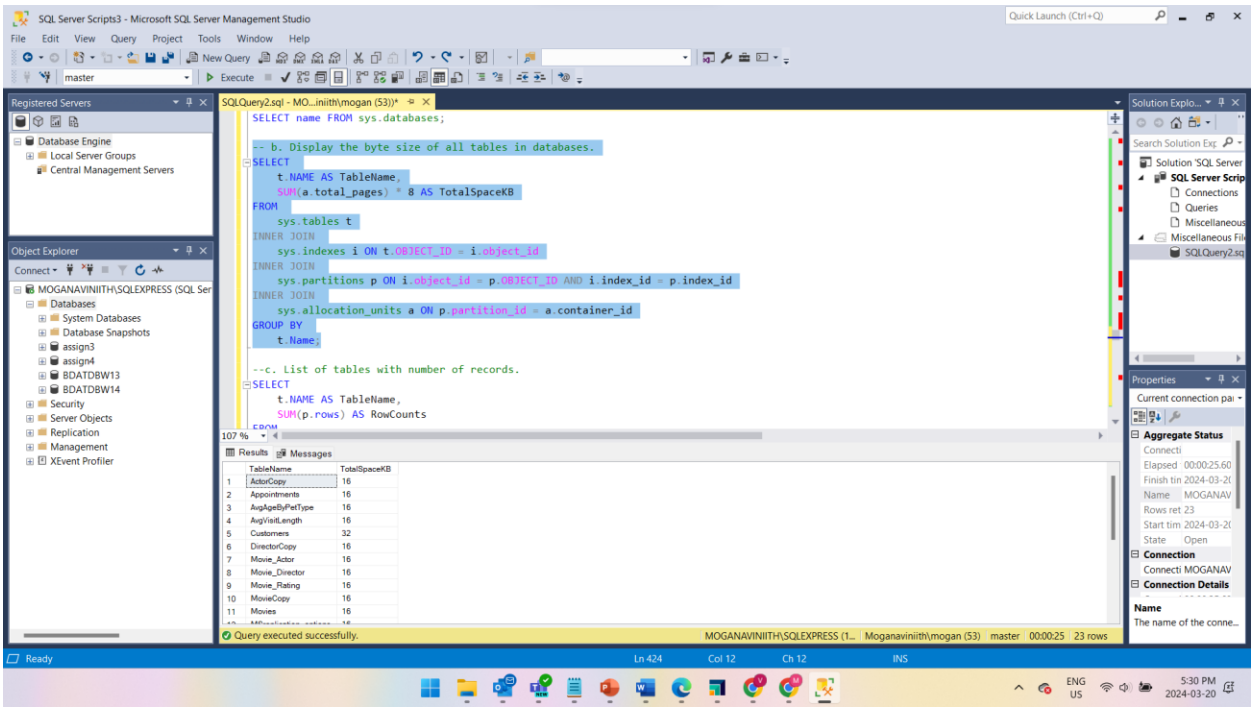
```
SELECT m.Movie_Name, d.Director_First_Name + ' ' + d.Director_Last_Name AS
Director_FullName, d.Director_Age_in_Years, d.Director_Gender
FROM mov.Movies m
JOIN mov.Movie_Director d ON m.Director_ID = d.Director_ID;
```

13.Write the following Query based on the above datasets.

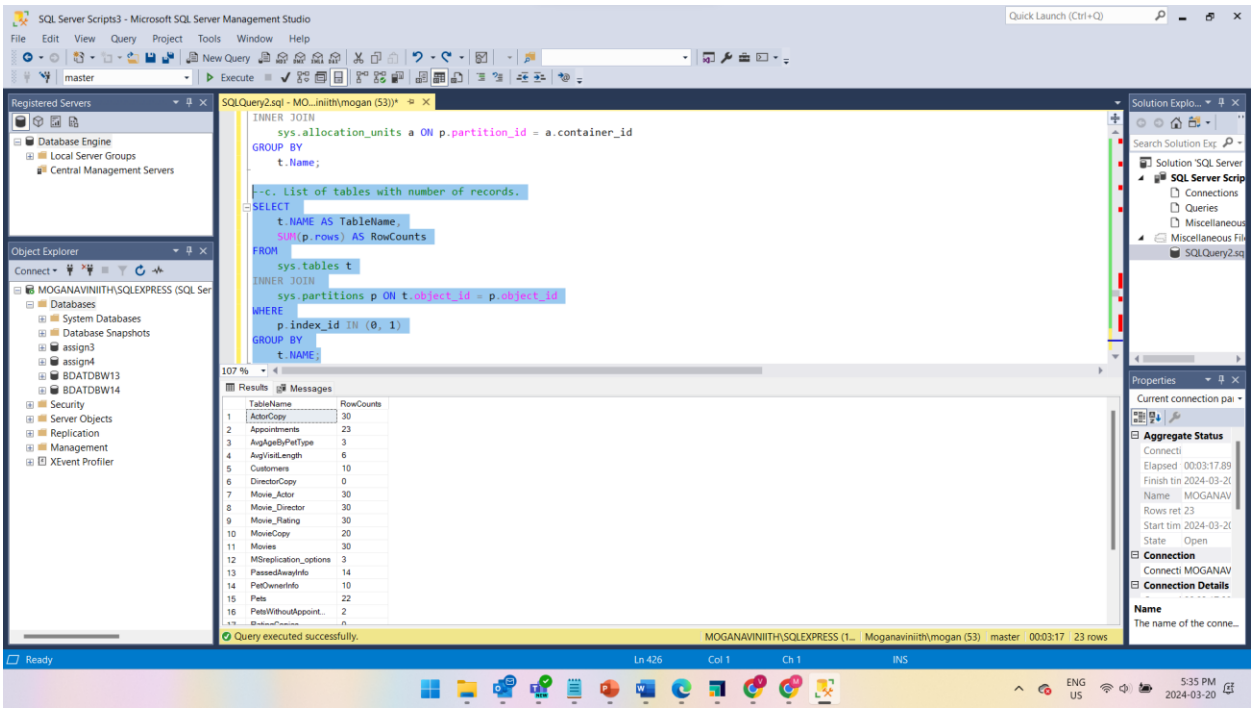
a. Retrieve the list of all Databases.



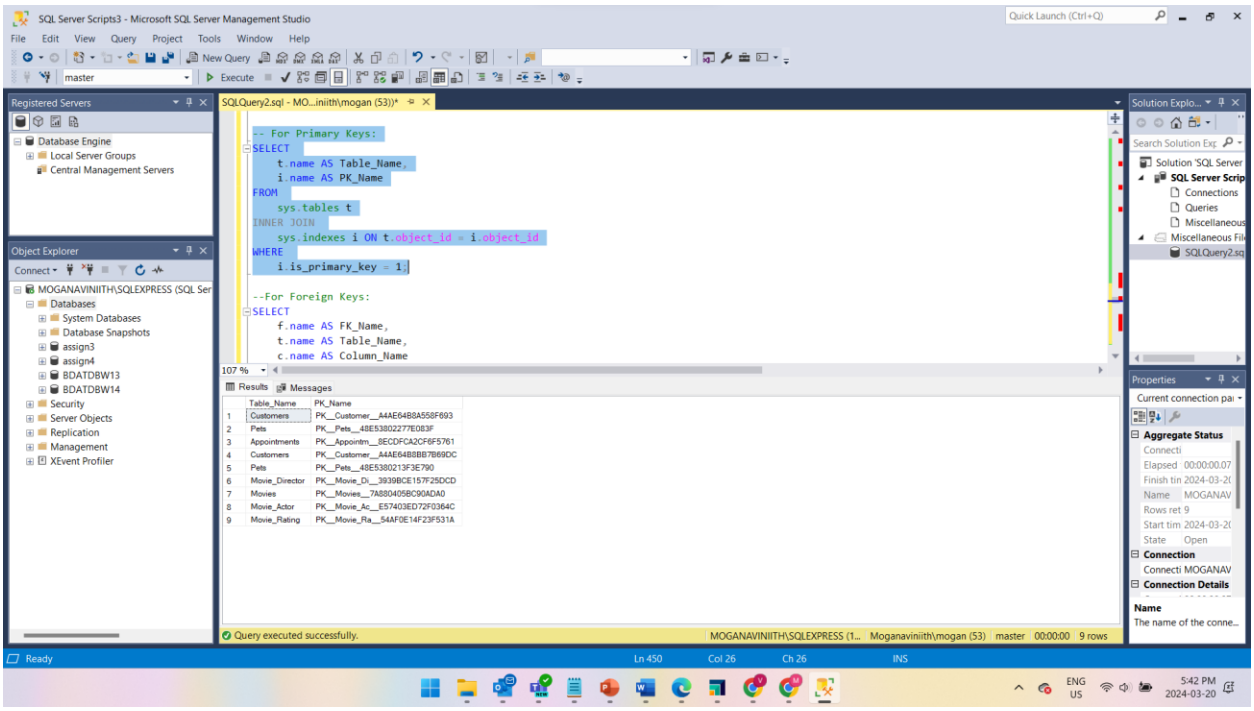
b. Display the byte size of all tables in databases.

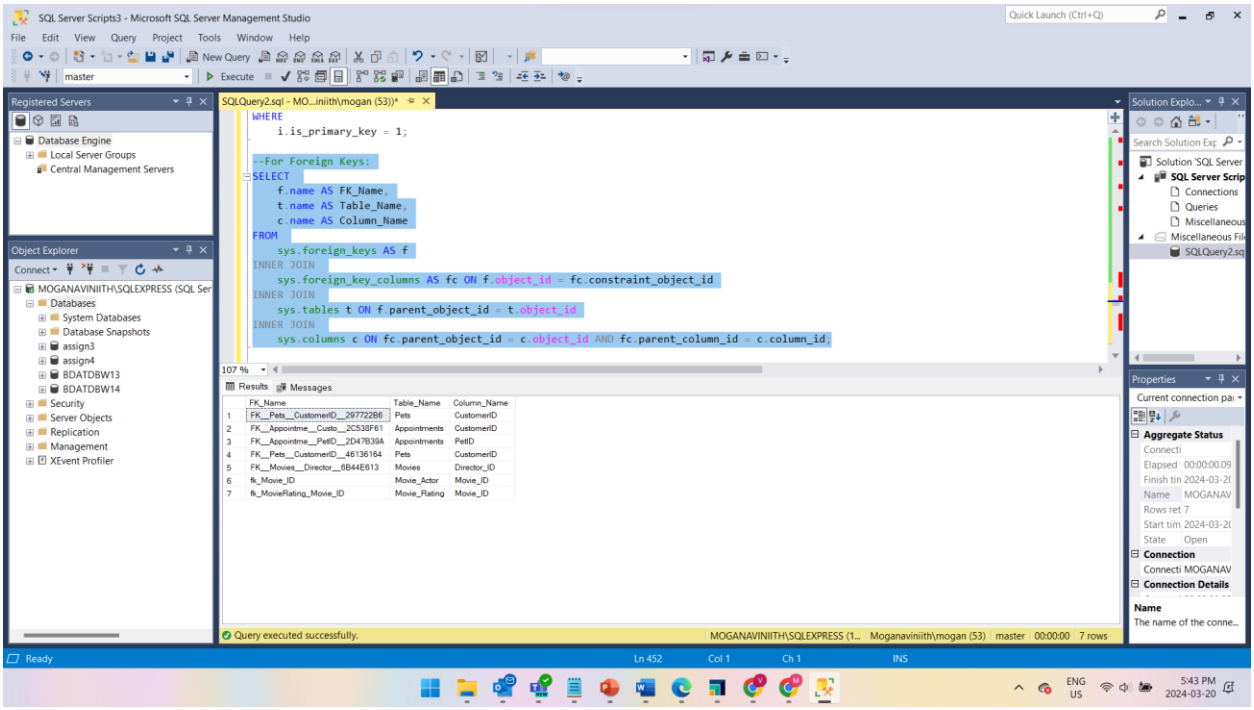


c. List of tables with number of records.

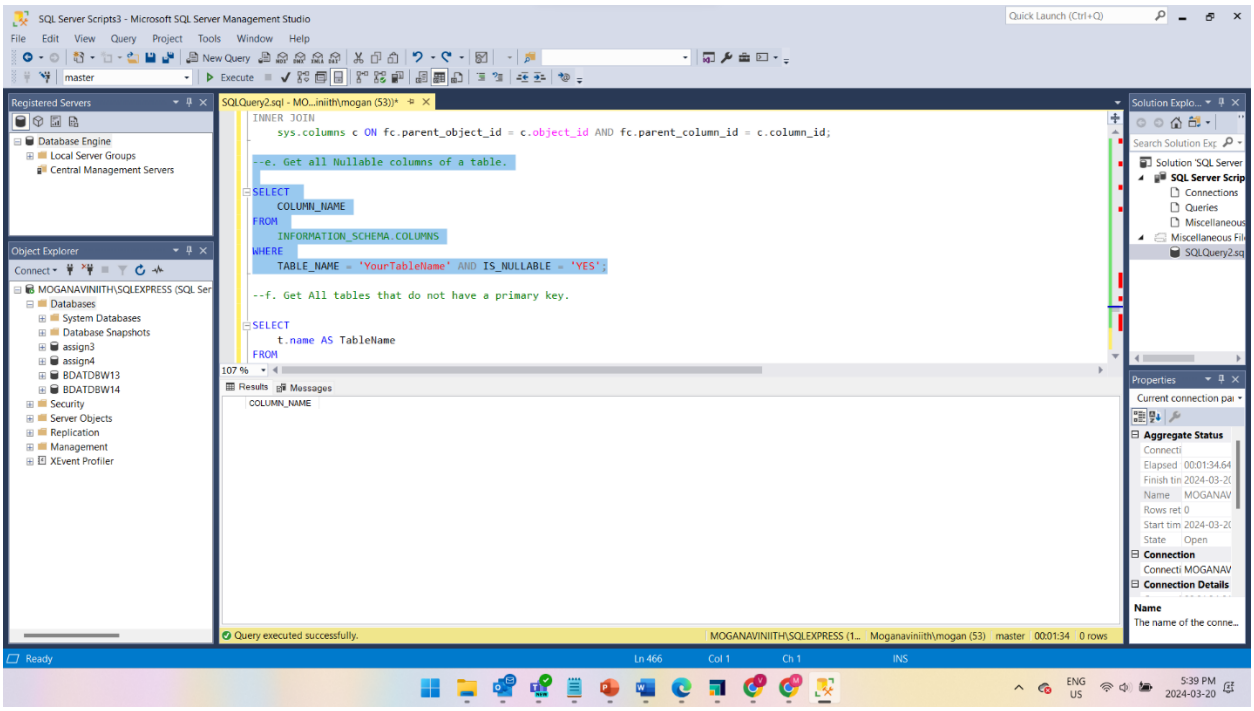


d. List of Primary Key and Foreign Key for Whole Database.

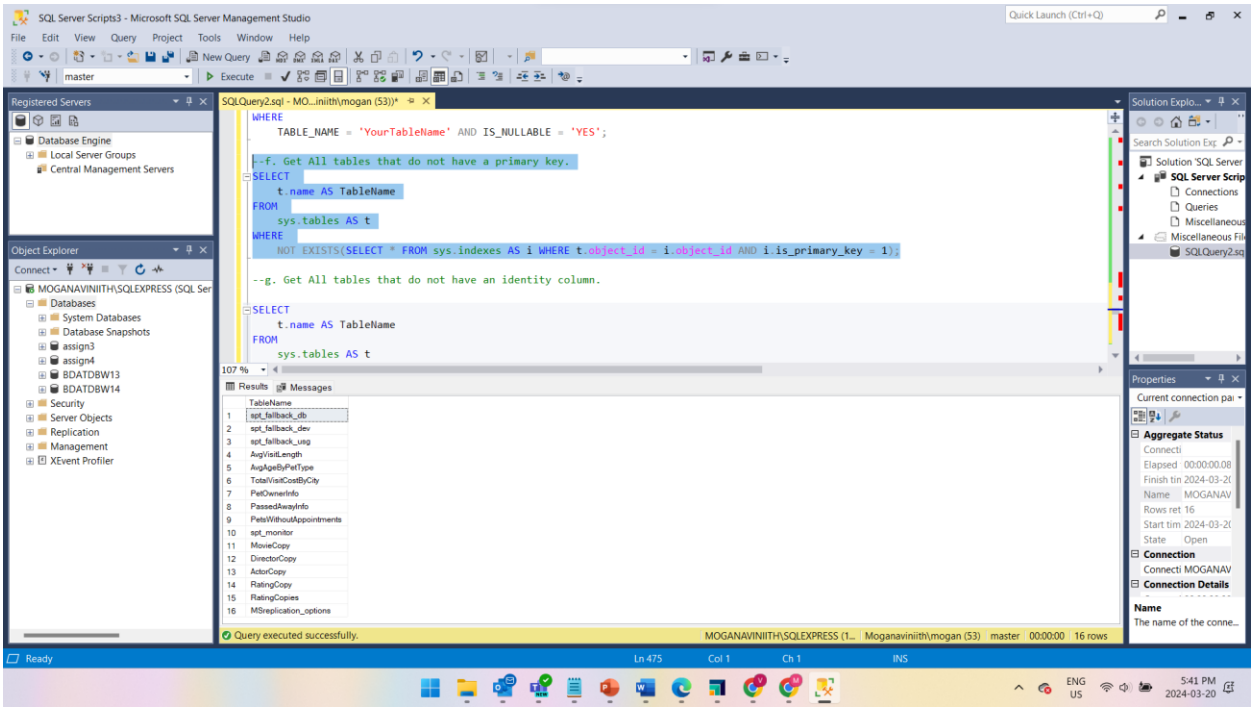




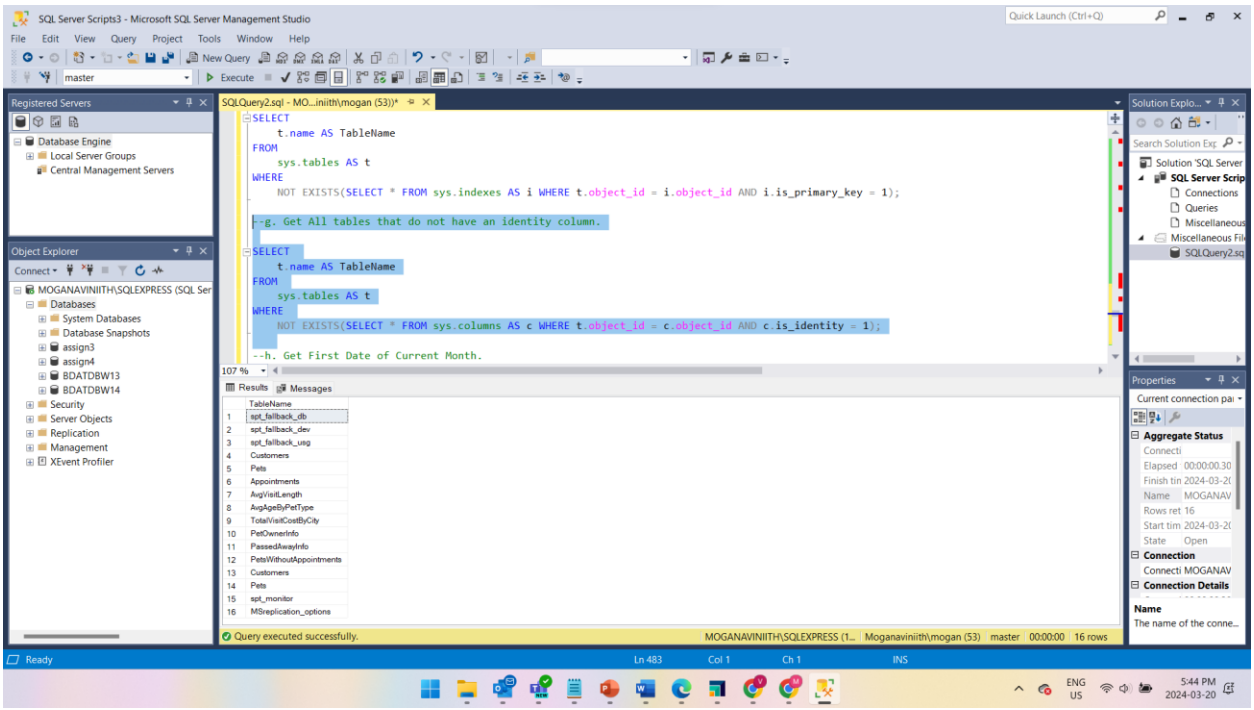
e. Get all Nullable columns of a table.



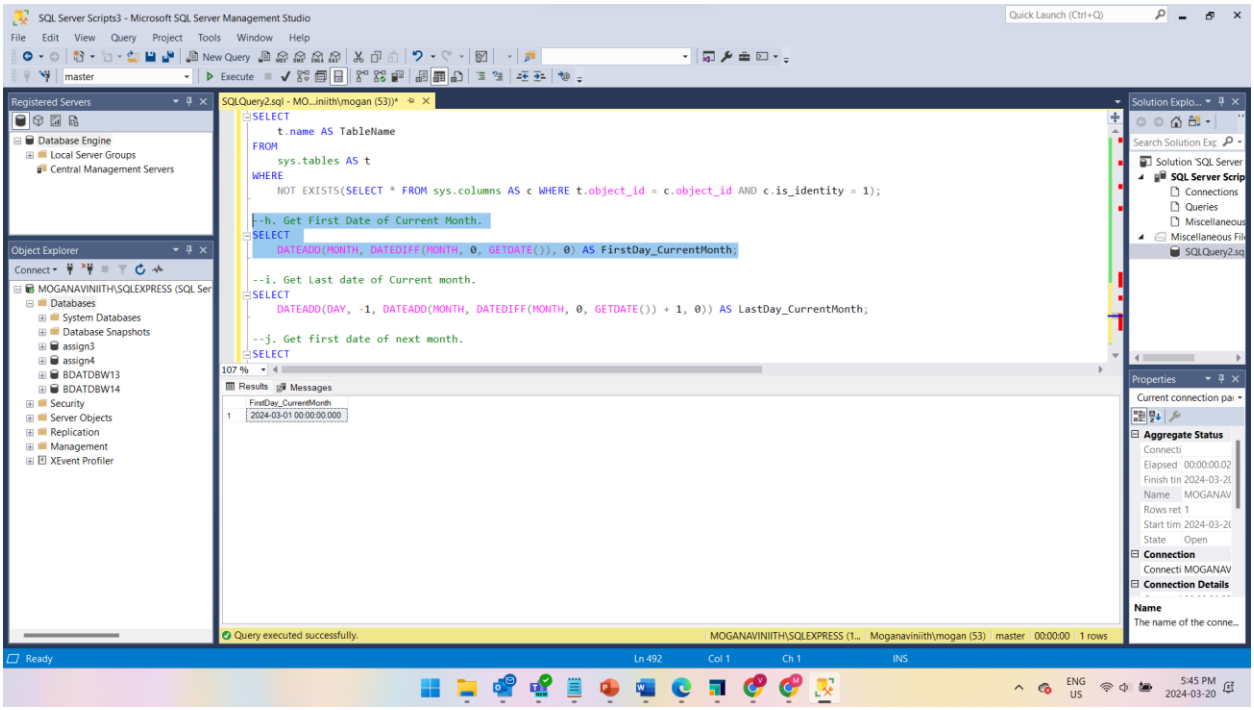
f. Get All table that do not have primary key.



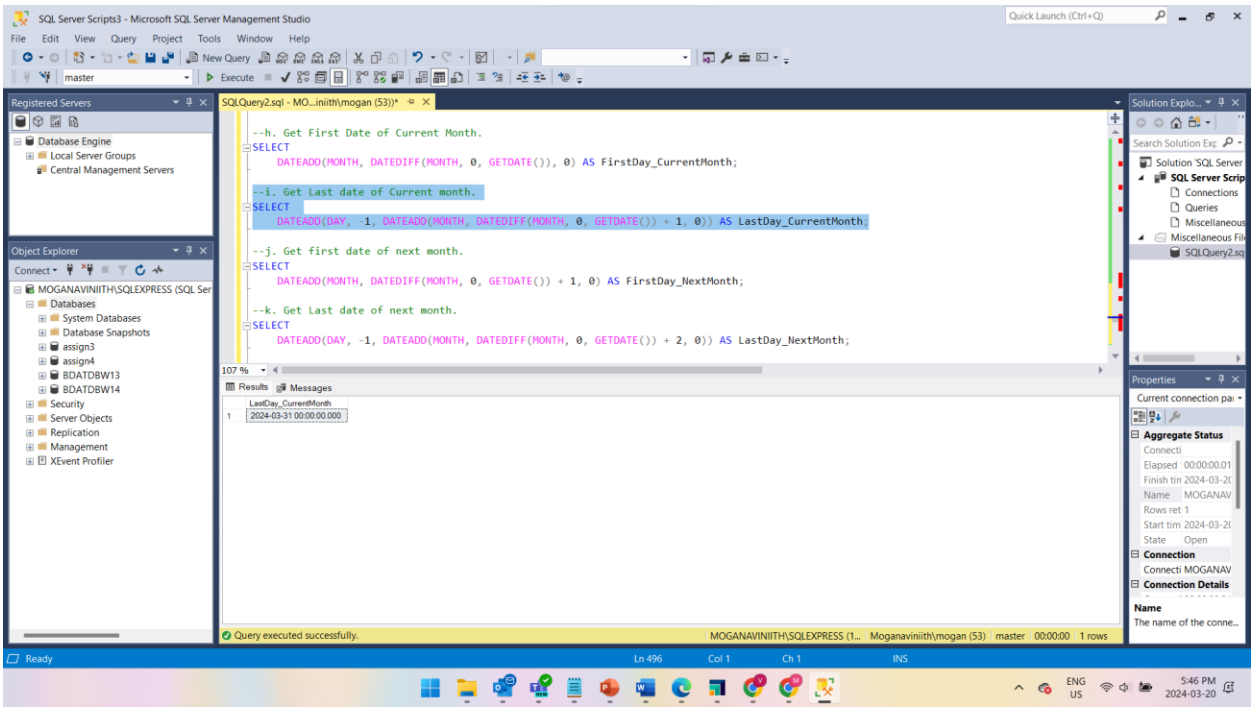
g. Get All table that do not have identity column.



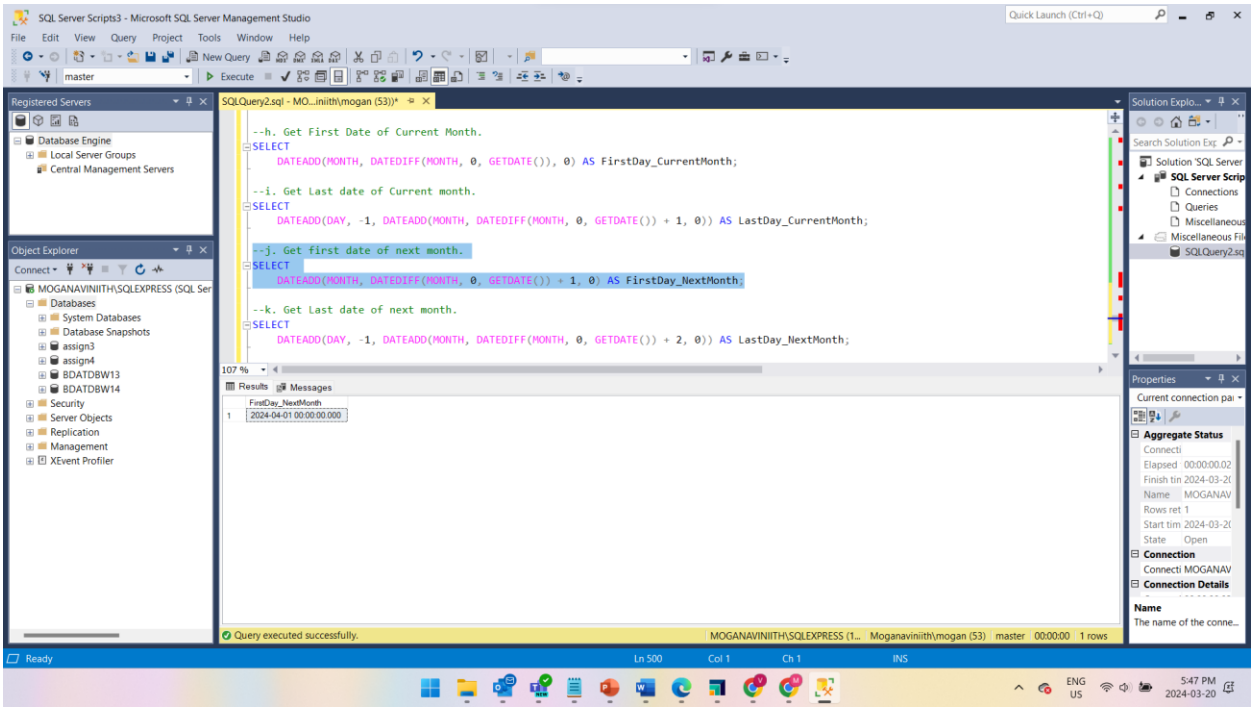
h. Get First Date of Current Month.



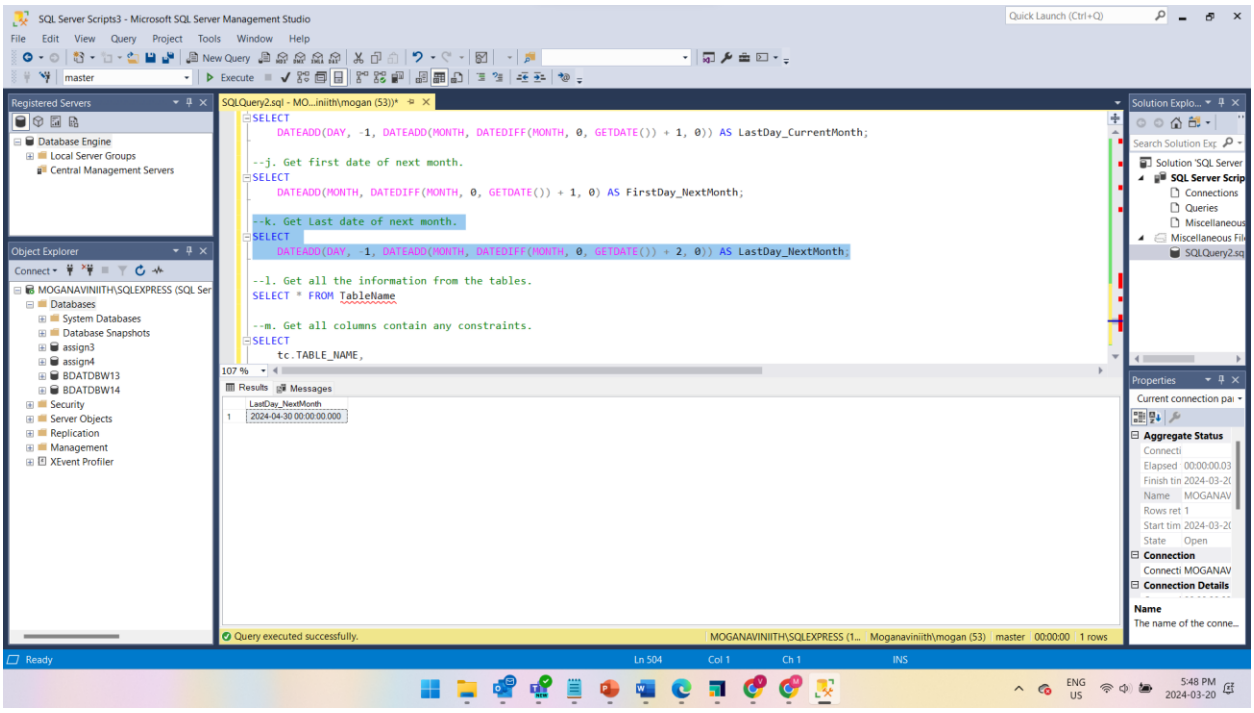
i. Get Last date of Current month.



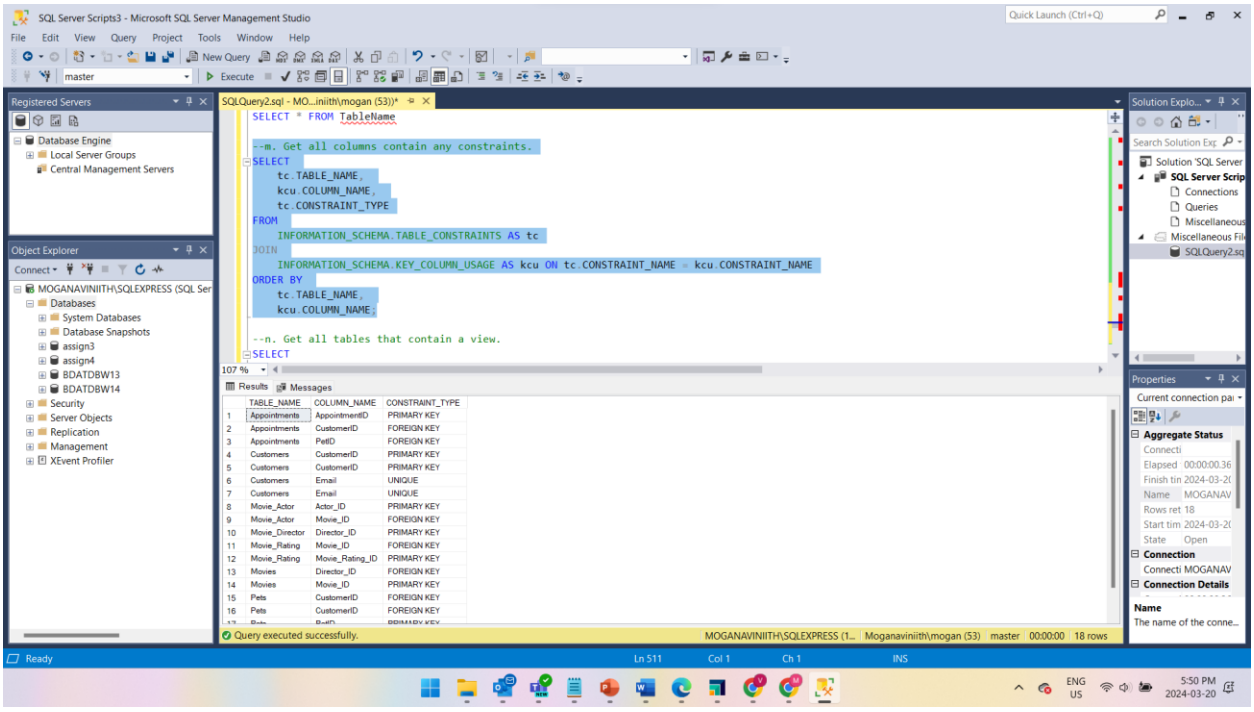
j. Get first date of next month.



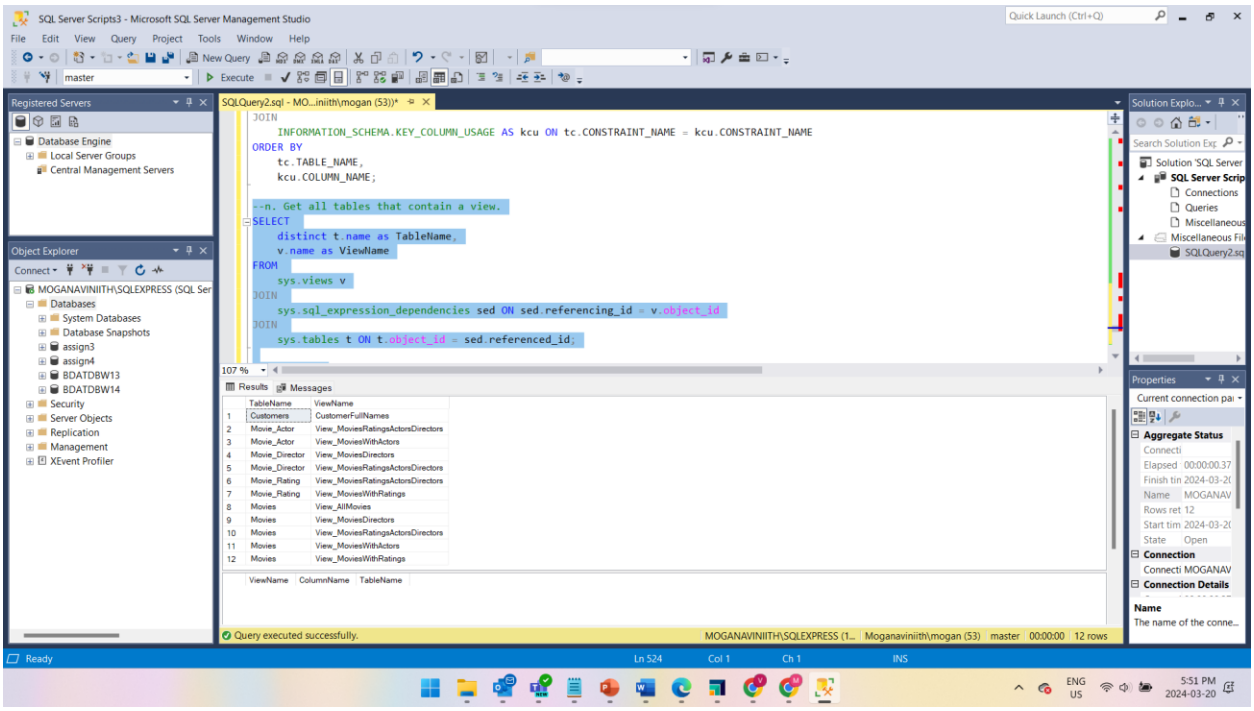
k. Get Last date of next month.



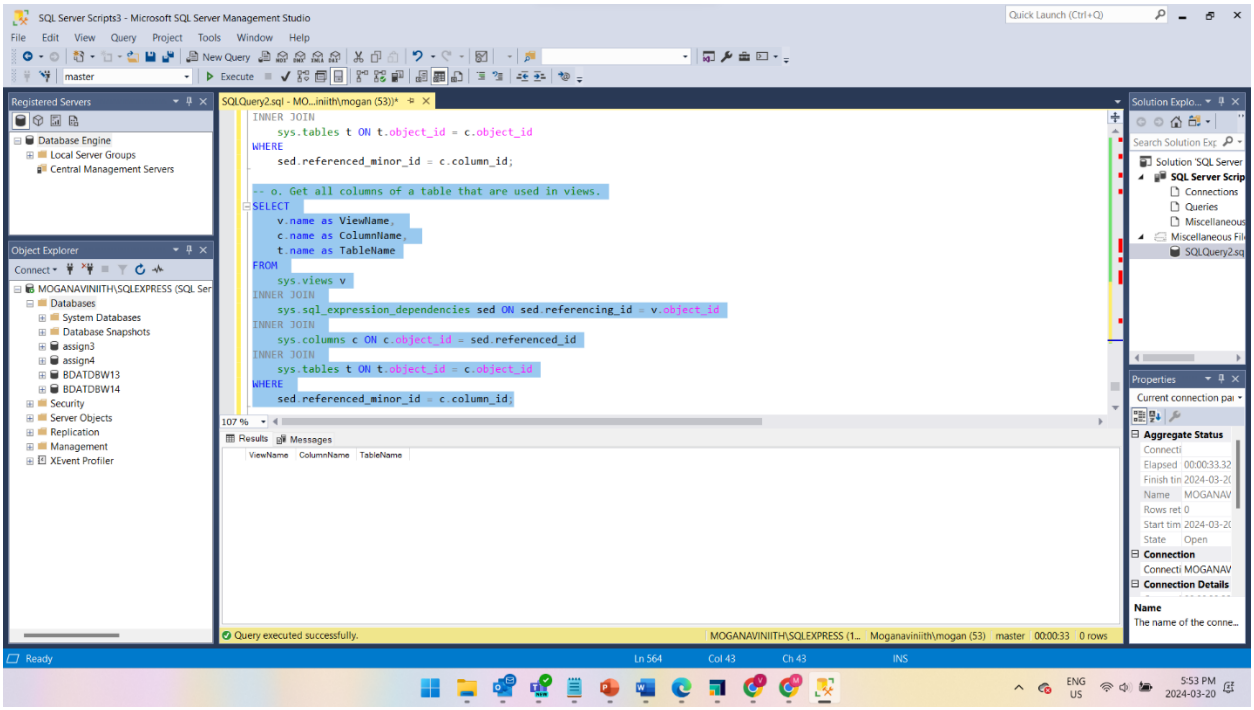
m. Get all columns contain any constraints.



n. Get all tables that contain a view.



o. Get all columns of table that using in views.



Name of students who has participated in the assignment.
Student name: Moganaviniith Rathinavel (200531668)
Student name: Nanditha Babu (200573747)
Student name: Aswin Narayanan (200568974)
Student name: Srivas Sathyanarayanan (200544678)
Student name: Raveen Kumar (200574798)