**Information System Management Lab**

**BCOM 307**

**Assignment #24**

***Submitted by:***

**Name:** YASH JAIN

**Enrollment No:** 03914788818

**Semester:** B.Com(H) 5th Semester

**Class:** B.COM(H)

**Section:** B.Com 5A

**Date of Submission:** 01/11/2021

***Submitted to:***

**Praveen Kumar Singh**

**Assistant Professor, MAIMS**

****

**Department of Commerce**

**Maharaja Agrasen Institute of Management Studies**

**Affiliated to Guru Gobind Singh Indraprastha University, Delhi**

**Sector -22, Rohini, Delhi -110086, India; www.maims.ac.in**

Maharaja Agrasen Institute of Management Studies

Affiliated to GGS IP University; Recognized u/s 2(f) of UGC Recognized by Bar Council ofIndia; ISO 9001: 2015

Certified Institution Sector 22, Rohini, Delhi -110086, India; [www.maims.ac.in](http://www.maims.ac.in/)

Department of Commerce

Academic Year: 2020-21

Semester: Vth

# Assignment No. 24

# Unit No:

**Course/Subject Code:** BCOM 307 **Subject Title: Information System Management Lab**

# Issue Date Last Date of Submission:

**Instructions for Students:**

# All Questions are Compulsory.

1. The student should attach proper cover page for each assignment clearly mentioning the Assignment No.
2. Each assignment should be prepared by the student individually with proper explaination and screenshots.
3. A4 size ruled sheets should be used for the assignment.
4. Assignment pages should be serially numbered at the bottom of page.

***During online education mode, upload scanned copy of the complete assignment including cover page latest by due date.***

|  |  |  |
| --- | --- | --- |
| **Question No.** | **Question** | **CO No.** |
| 1 | Create the following tables in the student\_record database:  MEMBERS :   |  |  | | --- | --- | | Column Name | Data Type | | Id (primary key) | int | | First\_Name | varchar | | Last\_Name | Varchar | | Movie\_Id (foreign key) | int |     MOVIES:   |  |  | | --- | --- | | Column Name | Data Type | | Id (primary key) | int | | Title | varchar | | Category | Varchar | | **CO2, CO3, CO4, CO5** |
| 2 | Insert records in both tables, as mentioned below:  MEMBERS :   |  |  |  |  | | --- | --- | --- | --- | | Id | First\_Name | Last\_Name | Movie\_Id | | 1 | Adam | Smith | 1 | | 2 | Ravi | Kumar | 2 | | 3 | Susan | Davidson | 5 | | 4 | Jenny | Adrianna | 4 | | 6 | Lee | Pong | 5 |   `  **CO2,**  **CO3, CO4, CO5**  MOVIES :   |  |  |  | | --- | --- | --- | | Id | Title | Category | | 1 | Assassin’s Creeds: Embers | Animation | | 2 | Real Steel(2012) | Animation | | 3 | Alvin and the Chipmunks | Animation | | 4 | Adventures of Tin Tin | Animation | | 5 | Safe (2012) | Action | | 6 | Safe House (2012) | Action | |
| 3 | Show all member records with all movie records. |
| 4 | Write SQL Command to show list of members with movie name who rented movies. |
| 5 | Show all movies with the names of members who rented them. |

**ASSIGNMENT 24 - SQL CROSS JOIN AND LEFT JOIN CLAUSE**

**Task 1 : Create the following tables in the student\_record database:**

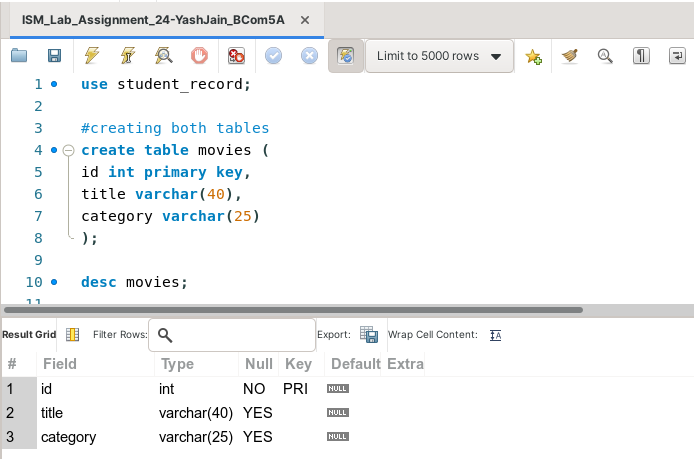
**MEMBERS :**

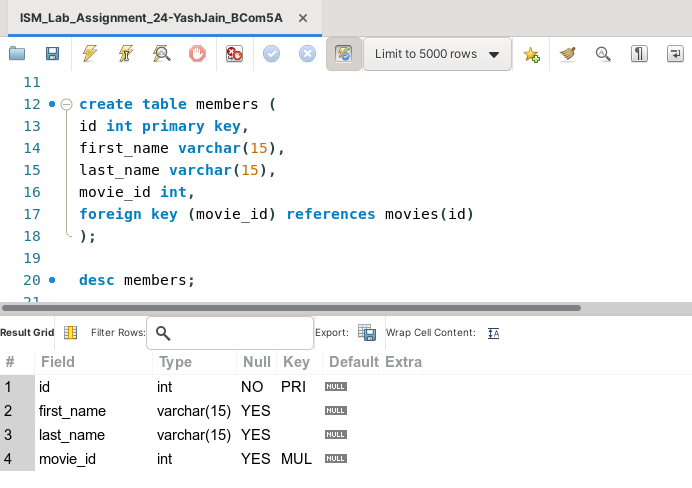
|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| Id (primary key) | int |
| First\_Name | varchar |
| Last\_Name | Varchar |
| Movie\_Id (foreign key) | int |

**MOVIES:**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| Id (primary key) | int |
| Title | varchar |
| Category | Varchar |

This task can be completed using the **CREATE TABLE** statement, along with the **DESC** clause to describe the table.





**Task 2: Insert records in both tables, as mentioned below:**

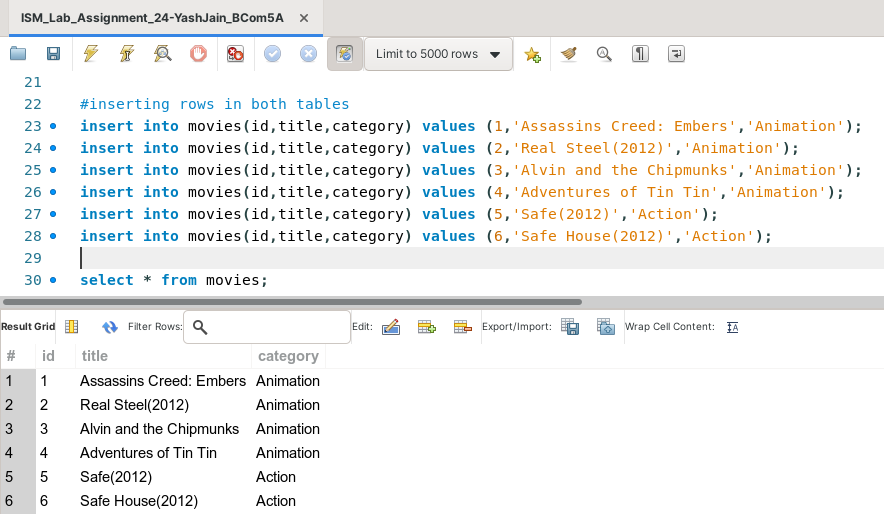
|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **First\_Name** | **Last\_Name** | **Movie\_Id** |
| **1** | **Adam** | **Smith** | **1** |
| **2** | **Ravi** | **Kumar** | **2** |
| **3** | **Susan** | **Davidson** | **5** |
| **4** | **Jenny** | **Adrianna** | **4** |
| **6** | **Lee** | **Pong** | **5** |

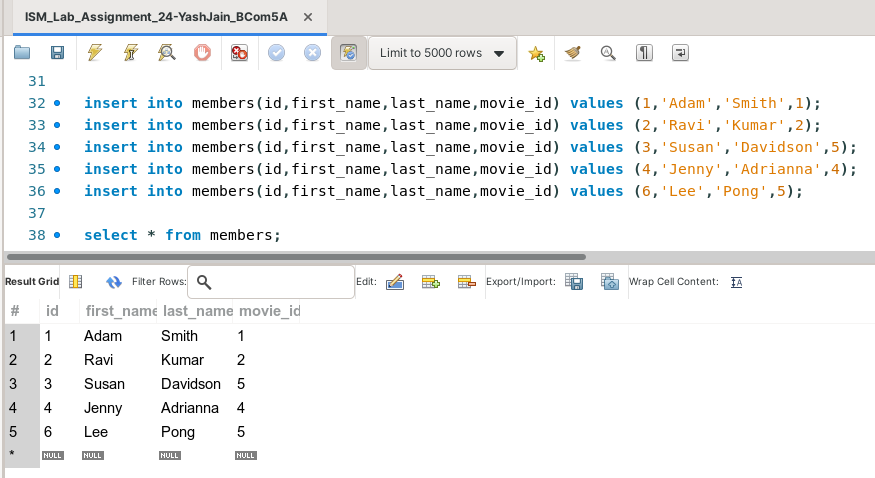
**MEMBERS :**

**`**

**MOVIES :**

|  |  |  |
| --- | --- | --- |
| **Id** | **Title** | **Category** |
| **1** | **Assassin’s Creeds: Embers** | **Animation** |
| **2** | **Real Steel(2012)** | **Animation** |
| **3** | **Alvin and the Chipmunks** | **Animation** |
| **4** | **Adventures of Tin Tin** | **Animation** |
| **5** | **Safe (2012)** | **Action** |
| **6** | **Safe House (2012)** | **Action** |

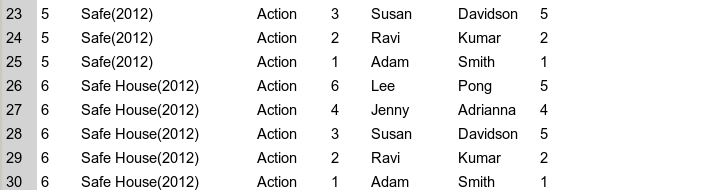
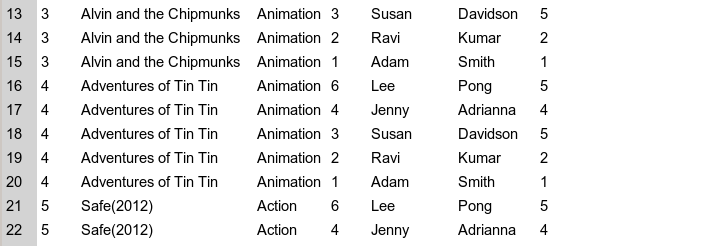
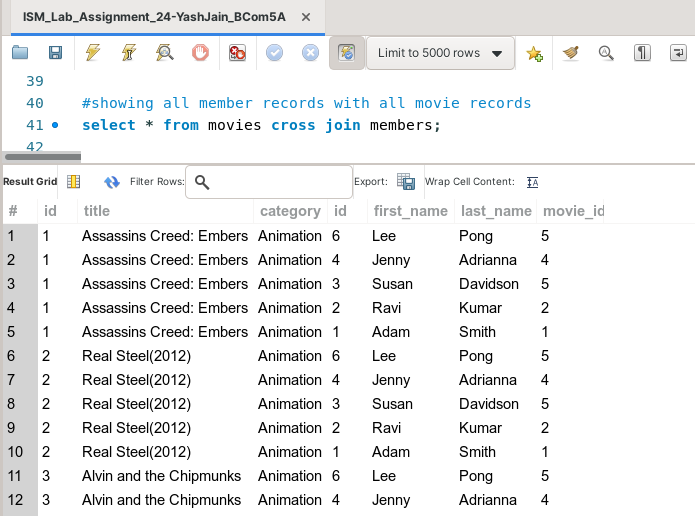
****

****

**Task 3: Show all member records with all movie records.**

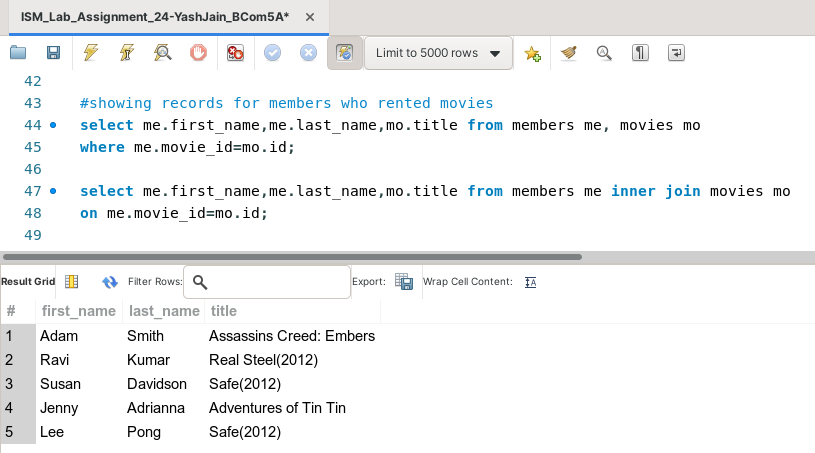
This task can be completed using the **SQL CROSS JOIN Clause.** The SQL CROSS JOIN Statement matches each record of one table to all the rows from another table. It is the simplest form of JOIN statement. In other words, it gives us combination of each row of first table with all the records in second table. It can be seen as a 'Cartesian Product' of 2 tables considered as Sets. The syntax for this is -

SELECT \* from table2 CROSS JOIN table1;

****

**Task 4: Write SQL Command to show list of members with movie name who rented movies.**

This task can be completed using the **INNER JOIN** Clause. Here, we don’t have to necessarily use the keyword INNER JOIN for the operation.

****

**Task 5: Show all movies with the names of members who rented them.**

This task can be completed using the **SQL LEFT JOIN** Clause. The **SQL OUTER JOIN** returns all records matching from both tables. It shows null values for records of joined table if no match is found. **LEFT JOIN** is a type of OUTER JOIN.

**LEFT JOIN** fetches all records from the first table even if there is no record in the other table. It displays the entries in the right table as NULL Values. The Syntax for this is -

SELECT column1,colum2 FROM table1 LEFT JOIN table2 ON table1.common\_column=table2.common\_column;

