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

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:

1. **All Questions are Compulsory.**
2. The student should attach proper cover page for each assignment clearly mentioning theAssignment No.
3. Each assignment should be prepared by the student individually with proper explaination andscreenshots.
4. A4 size ruled sheets should be used for the assignment.
5. 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.

<table>
<tr><th>Question No.</th><th>Question</th><th>CO No.</th></tr>
<tr><td>1</td><td>Create the following tables in the student_record database:

MEMBERS :

<table><tr><td>Column Name</td><td>Data Type</td></tr><tr><td>Id (primary key)</td><td>int</td></tr><tr><td>First_Name</td><td>varchar</td></tr><tr><td>Last_Name</td><td>Varchar</td></tr><tr><td>Movie_Id (foreign key)</td><td>int</td></tr></table>
MOVIES:

<table><tr><td>Column Name</td><td>Data Type</td></tr><tr><td>Id (primary key)</td><td>int</td></tr><tr><td>Title</td><td>varchar</td></tr><tr><td>Category</td><td>Varchar</td></tr></table></td><td>CO2, CO3, CO4, CO5</td></tr>
</table>

<table>
<tr>
<td>2</td>
<td>Insert records in both tables, as mentioned below:

MEMBERS :

<table>
<tr><th>Id</th><th>First_Name</th><th>Last_Name</th><th>Movie_Id</th></tr>
<tr><td>1</td><td>Adam</td><td>Smith</td><td>1</td></tr>
<tr><td>2</td><td>Ravi</td><td>Kumar</td><td>2</td></tr>
<tr><td>3</td><td>Susan</td><td>Davidson</td><td>5</td></tr>
<tr><td>4</td><td>Jenny</td><td>Adrianna</td><td>4</td></tr>
<tr><td>6</td><td>Lee</td><td>Pong</td><td>5</td></tr>
</table>
`

MOVIES :

<table>
<tr><th>Id</th><th>Title</th><th>Category</th></tr>
<tr><td>1</td><td>Assassin’s Creeds: Embers</td><td>Animation</td></tr>
<tr><td>2</td><td>Real Steel(2012)</td><td>Animation</td></tr>
<tr><td>3</td><td>Alvin and the Chipmunks</td><td>Animation</td></tr>
<tr><td>4</td><td>Adventures of Tin Tin</td><td>Animation</td></tr>
<tr><td>5</td><td>Safe (2012)</td><td>Action</td></tr>
<tr><td>6</td><td>Safe House (2012)</td><td>Action</td></tr>
</table>
</td>
<td rowspan="3">CO2, CO3, CO4, CO5</td>
</tr>
<tr><td>3</td><td>Show all member records with all movie records.</td></tr>
<tr><td>4</td><td>Write SQL Command to show list of members with movie name who rented movies.</td></tr>
<tr><td>5</td><td>Show all movies with the names of members who rented them.</td></tr>
</table>

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.

ISM_Lab_Assignment_24-YashJain_BCom5A

Limit to 5000 rows

```
use student_record;

#creating both tables
create table movies (
id int primary key,
title varchar(40),
category varchar(25)
);

desc movies;
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	Field	Type	Null	Key	Default	Extra
1	id	int	NO	PRI	NULL	
2	title	varchar(40)	YES		NULL	
3	category	varchar(25)	YES		NULL	

ISM_Lab_Assignment_24-YashJain_BCom5A

Limit to 5000 rows

```
create table members (
id int primary key,
first_name varchar(15),
last_name varchar(15),
movie_id int,
foreign key (movie_id) references movies(id)
);

desc members;
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	Field	Type	Null	Key	Default	Extra
1	id	int	NO	PRI	NULL	
2	first_name	varchar(15)	YES		NULL	
3	last_name	varchar(15)	YES		NULL	
4	movie_id	int	YES	MUL	NULL	

Task 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

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

ISM_Lab_Assignment_24-YashJain_BCom5A

```
#inserting rows in both tables
insert into movies(id,title,category) values (1,'Assassins Creed: Embers','Animation');
insert into movies(id,title,category) values (2,'Real Steel(2012)','Animation');
insert into movies(id,title,category) values (3,'Alvin and the Chipmunks','Animation');
insert into movies(id,title,category) values (4,'Adventures of Tin Tin','Animation');
insert into movies(id,title,category) values (5,'Safe(2012)','Action');
insert into movies(id,title,category) values (6,'Safe House(2012)','Action');

select * from movies;
```

#	id	title	category
1	1	Assassins Creed: Embers	Animation
2	2	Real Steel(2012)	Animation
3	3	Alvin and the Chipmunks	Animation
4	4	Adventures of Tin Tin	Animation
5	5	Safe(2012)	Action
6	6	Safe House(2012)	Action

ISM_Lab_Assignment_24-YashJain_BCom5A

```
insert into members(id,first_name,last_name,movie_id) values (1,'Adam','Smith',1);
insert into members(id,first_name,last_name,movie_id) values (2,'Ravi','Kumar',2);
insert into members(id,first_name,last_name,movie_id) values (3,'Susan','Davidson',5);
insert into members(id,first_name,last_name,movie_id) values (4,'Jenny','Adrianna',4);
insert into members(id,first_name,last_name,movie_id) values (6,'Lee','Pong',5);

select * from members;
```

#	id	first_name	last_name	movie_id
1	1	Adam	Smith	1
2	2	Ravi	Kumar	2
3	3	Susan	Davidson	5
4	4	Jenny	Adrianna	4
5	6	Lee	Pong	5
*	NULL	NULL	NULL	NULL

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;
```

ISM_Lab_Assignment_24-YashJain_BCom5A

Limit to 5000 rows

```
#showing all member records with all movie records
select * from movies cross join members;
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	id	title	category	id	first_name	last_name	movie_id
1	1	Assassins Creed: Embers	Animation	6	Lee	Pong	5
2	1	Assassins Creed: Embers	Animation	4	Jenny	Adrianna	4
3	1	Assassins Creed: Embers	Animation	3	Susan	Davidson	5
4	1	Assassins Creed: Embers	Animation	2	Ravi	Kumar	2
5	1	Assassins Creed: Embers	Animation	1	Adam	Smith	1
6	2	Real Steel(2012)	Animation	6	Lee	Pong	5
7	2	Real Steel(2012)	Animation	4	Jenny	Adrianna	4
8	2	Real Steel(2012)	Animation	3	Susan	Davidson	5
9	2	Real Steel(2012)	Animation	2	Ravi	Kumar	2
10	2	Real Steel(2012)	Animation	1	Adam	Smith	1
11	3	Alvin and the Chipmunks	Animation	6	Lee	Pong	5
12	3	Alvin and the Chipmunks	Animation	4	Jenny	Adrianna	4
13	3	Alvin and the Chipmunks	Animation	3	Susan	Davidson	5
14	3	Alvin and the Chipmunks	Animation	2	Ravi	Kumar	2
15	3	Alvin and the Chipmunks	Animation	1	Adam	Smith	1
16	4	Adventures of Tin Tin	Animation	6	Lee	Pong	5
17	4	Adventures of Tin Tin	Animation	4	Jenny	Adrianna	4
18	4	Adventures of Tin Tin	Animation	3	Susan	Davidson	5
19	4	Adventures of Tin Tin	Animation	2	Ravi	Kumar	2
20	4	Adventures of Tin Tin	Animation	1	Adam	Smith	1
21	5	Safe(2012)	Action	6	Lee	Pong	5
22	5	Safe(2012)	Action	4	Jenny	Adrianna	4
23	5	Safe(2012)	Action	3	Susan	Davidson	5
24	5	Safe(2012)	Action	2	Ravi	Kumar	2
25	5	Safe(2012)	Action	1	Adam	Smith	1
26	6	Safe House(2012)	Action	6	Lee	Pong	5
27	6	Safe House(2012)	Action	4	Jenny	Adrianna	4
28	6	Safe House(2012)	Action	3	Susan	Davidson	5
29	6	Safe House(2012)	Action	2	Ravi	Kumar	2
30	6	Safe House(2012)	Action	1	Adam	Smith	1

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.

ISM_Lab_Assignment_24-YashJain_BCom5A*

```
#showing records for members who rented movies
select me.first_name,me.last_name,mo.title from members me, movies mo
where me.movie_id=mo.id;

select me.first_name,me.last_name,mo.title from members me inner join movies mo
on me.movie_id=mo.id;
```

Result Grid

#	first_name	last_name	title
1	Adam	Smith	Assassins Creed: Embers
2	Ravi	Kumar	Real Steel(2012)
3	Susan	Davidson	Safe(2012)
4	Jenny	Adrianna	Adventures of Tin Tin
5	Lee	Pong	Safe(2012)

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;
```

ISM_Lab_Assignment_24-YashJain_BCom5A

```
#showing records for all movies along with members who rented them
select mo.title,me.first_name,me.last_name from movies mo left join
members me on me.movie_id=mo.id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

#	title	first_name	last_name
1	Assassins Creed: Embers	Adam	Smith
2	Real Steel(2012)	Ravi	Kumar
3	Alvin and the Chipmunks	NULL	NULL
4	Adventures of Tin Tin	Jenny	Adrianna
5	Safe(2012)	Susan	Davidson
6	Safe(2012)	Lee	Pong
7	Safe House(2012)	NULL	NULL