

Laboratory Assignments Subject: Introduction to Databases Subject code: CSE 3151

**Name – VEDANT
HARSHIT
Reg no. - 1941012746
Sec – I**

Assignment 1: Practicing DDL and DML Commands

Objective of this Assignment:

- To create tables and to insert data into these tables.
1. Create the following Tables with their respective columns, data types and size.

Table Name	Columns	Datatypes	Size
Instructor	ID	Number	5
	Name	Varchar2	30
	Dept_name	Varchar2	20
	Salary	Number	6

Query:

```
CREATE TABLE INSTRUCTOR(  
ID NUMERIC(5),  
NAME VARCHAR(30),  
DEPT_NAME VARCHAR(20),  
SALARY NUMERIC(6)  
)
```

Output:

```

SQLQuery2.sql - LAPTOP-CLORFFAO\SQLEXPRESS.master (LAPTOP-CLORFFAO\Admin (61)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
New Query Execute
Object Explorer
Connect > master
LAPTOP-CLORFFAO\SQLEXPRESS (SQL Server)
System Databases
master
Tables
System Tables
External Tables
Graph Tables
dbo.INSTRUCTOR
Views
Synonyms
Programmability
Service Broker
Storage
Security
model
msdb
tempdb
Database Snapshots
Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler
master
[SQLQuery2.sql] [LAPTOP-CLORFFAO\Admin (61)] [1941012768.sql] [OFFFA0\Admin (5B)]
***** Script for SelectTopNRows command from SSMS *****
SELECT TOP (1000) [ID]
      ,[NAME]
      ,[DEPT_NAME]
      ,[SALARY]
   FROM [master].[dbo].[INSTRUCTOR]

Results Messages
ID | NAME | DEPT_NAME | SALARY
1  | null | null       | null

```

Query executed successfully.

2. Create the following Tables with their respective columns, data types and size.

Table Name	Columns	Datatypes	Size
Course	Course_id	Varchar2	10
	Title	Varchar2	30
	Dept_name	Varchar2	20
	Credits	Number	2

Query:

```

CREATE TABLE COURSE(
COURSE_ID VARCHAR(10),
TITLE VARCHAR(30),
DEPT_NAME VARCHAR(20),
CREDITS NUMERIC(2)
);

```

Output:

SQLQuery3.sql - LAPTOP-CLORFFA0\SQLEXPRESS.master (LAPTOP-CLORFFA0\Admin (61)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

Object Explorer

Connect > LAPTOP-CLORFFA0\SQLEXPRESS (SQL Server) > master

master

Execute

SQLQuery3.sql - L... ORFFA0\Admin (61) 1941012768.sql - ORFFA0\Admin (58)*

```
***** Script for SelectTopNRows command from SSMS *****
SELECT TOP (1000) [COURSE_ID]
      ,[TITLE]
      ,[DEPT_NAME]
      ,[CREDITS]
   FROM [master].[dbo].[COURSE]
```

Results Messages

COURSE_ID	TITLE	DEPT_NAME	CREDITS

Query executed successfully.

LAPTOP-CLORFFA0\SQLEXPRESS ... LAPTOP-CLORFFA0\Admin ... master 00:00:00 0 rows

3. Create the following Tables with their respective columns, data types and size.

Table Name	Columns	Datatypes	Size
Prereq	Course_id	Varchar2	10
	Prereq_id	Varchar2	10

Query:

```
CREATE TABLE PREREQ(
COURSE_ID VARCHAR(10),
PREREQ_ID VARCHAR(10)
);
```

Output:

4. Create the following Tables with their respective columns, data types and size.

Table Name	Columns	Datatypes	Size
Department	Dept_name	Varchar2	20
	Building	Varchar2	20
	Budget	Number	10

Query:

```
CREATE TABLE DEPARTMENT(
    DEPARTMENT_NAME VARCHAR(20),
    BUILDING VARCHAR(20),
    BUDGET NUMERIC(20)
);
```

Output:

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure under 'master'. In the center, a query window titled 'SQLQuery2.sql' contains the script for creating the 'DEPARTMENT' table. Below it, another window titled 'SQLQuery1.sql' contains a SELECT statement to retrieve data from the table. The results pane at the bottom shows the expected output with three columns: DEPARTMENT_NAME, BUILDING, and BUDGET. A status bar at the bottom indicates 'Query executed successfully.'

```
CREATE TABLE DEPARTMENT(
    DEPARTMENT_NAME VARCHAR(20),
    BUILDING VARCHAR(20),
    BUDGET NUMERIC(20)
);

SELECT TOP (1000) [DEPARTMENT_NAME]
      ,[BUILDING]
      ,[BUDGET]
   FROM [master].[dbo].[DEPARTMENT]
```

5. Create the following Tables with their respective columns, data types and size.

Table Name	Columns	Datatypes	Size
Teaches	ID	Number	5
	Course_id	Varchar2	10
	Sec_id	Number	2
	Semester	Varchar2	10
	year	Number	4

Query:

```

CREATE TABLE TEACHES(
    ID NUMERIC(5),
    COURSE_ID VARCHAR(10),
    SEC_ID NUMERIC(2),
    SEMESTAR VARCHAR(10),
    YEAR NUMERIC(4)
)

```

Output:

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure, including the master database and its tables (e.g., COURSE, DEPARTMENT, INSTRUCTOR, PREREQ, TEACHES). In the center, the Query Editor pane contains the SQL code for creating the TEACHES table and a subsequent SELECT query. The Results pane at the bottom shows the output of the SELECT query, which is empty (0 rows). A status bar at the bottom indicates the query was executed successfully.

```

SQLQuery5.sql - LAPTOP-CLORFFA0\SQLEXPRESS.master (LAPTOP-CLORFFA0\Admin (63)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
master Execute
Object Explorer
SQLQuery5.sql - L..ORFFA0\Admin (63) 1041012768.sq ~, ORFFA0\Admin (60)
SELECT TOP (1000) [ID]
    ,[COURSE_ID]
    ,[SEC_ID]
    ,[SEMESTAR]
    ,[YEAR]
FROM [master].[dbo].[TEACHES]

Results Messages
ID | COURSE_ID | SEC_ID | SEMESTAR | YEAR
0  |           |       |          |      |
0 rows
LAPTOP-CLORFFA0\SQLEXPRESS ... LAPTOP-CLORFFA0\Admin ... master 00:00:00 0 rows

```

2. Insert data into **Instructor** table with the data below:

ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Query:

```
INSERT INTO Instructor (ID,Name,Dept_name,Salary)
VALUES (10101,'Srinivasan','Comp.Sci',650000);
select * from instructor
Insert INTO Instructor VALUES (12121,'Wu','Finance',90000);
Insert INTO Instructor VALUES (15151,'Mozart','Music',40000);
Insert INTO Instructor VALUES (22222,'Einstein','Physics',90000);
Insert INTO Instructor VALUES (32343,'El Said','History',60000);
Insert INTO Instructor VALUES (33456,'Gold','Physics',87000);
Insert INTO Instructor VALUES (45565,'Katz','Comp.Sci',75000);
Insert INTO Instructor VALUES (58856,'Califieri','History',62000);
Insert INTO Instructor VALUES (76543,'Singh','Finance',80000);
Insert INTO Instructor VALUES (76766,'Crick','Biology',72000);
Insert INTO Instructor VALUES (83821,'Brandt','Comp.sci',92000);
Insert INTO Instructor VALUES (98345,'Kim','Elec.Eng',80000);
```

Output:

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar indicates the connection is to 'LAPTOP-CLORFFA0\SQLEXPRESS master' (Admin (53)). The Object Explorer on the left shows the database structure, including 'master' and various tables like 'INSTRUCTOR', 'TEACHES', and 'COURSE'. The central pane displays a query window with the following SQL code:

```
1941012768(2).sql - [LAPTOP-CLORFFA0\Admin (53)] - [1941012768.sql] - [LAPTOP-CLORFFA0\Admin (60)]
INSERT INTO Instructor (ID,Name,Dept_name,Salary)
VALUES (10101,'Srinivasan','Comp.Sci',650000);
select * from instructor
Insert INTO Instructor VALUES (12121,'Wu','Finance',90000);
Insert INTO Instructor VALUES (15151,'Mozart','Music',40000);
Insert INTO Instructor VALUES (22222,'Einstein','Physics',90000);
Insert INTO Instructor VALUES (32343,'El Said','History',60000);
Insert INTO Instructor VALUES (33456,'Gold','Physics',87000);
Insert INTO Instructor VALUES (45565,'Katz','Comp.Sci',75000);
Insert INTO Instructor VALUES (58866,'Califeri','History',62000);
Insert INTO Instructor VALUES (76543,'Singh','Finance',80000);
Insert INTO Instructor VALUES (76766,'Crick','Biology',72000);
Insert INTO Instructor VALUES (83821,'Brandit','Comp.sci',92000);
Insert INTO Instructor VALUES (98345,'Kim','Elec.Eng',80000);
select * from instructor
```

The results pane below shows the output of the query, listing 12 rows of data:

ID	NAME	DEPT_NAME	SALARY
10101	Srinivasan	Comp.Sci	650000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	90000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp.Sci	75000
58866	Califeri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandit	Comp.sci	92000
98345	Kim	Elec.Eng	80000

3. Insert data into **Course** table with the data below:

course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

Query:

```
Query:
insert into Course Values('BIO-101', 'Intro. to Biology', 'Biology', 4);
insert into Course Values('BIO-301', 'Genetics', 'Biology', 4);
insert into Course Values('BIO-399', 'Computational Biology', 'Biology', 3);

insert into Course Values('CS-101', 'Intro. to Computer Science', 'Computer Sci.', 4);
insert into Course Values('CS-190', 'Game Design', 'Computer. Sci.', 4);
insert into Course Values('CS-315', 'Robotics', 'Computer. Sci.', 3);
```

```

insert into Course Values('CS-319', 'Image Processing', 'Computer. Sci.',3);
insert into Course Values('CS-347', 'Database System Concepts', 'Computer. Sci.',3);

insert into Course Values('EE-181', 'Intro. to Digital Systems', 'Elec. Eng.',3);
insert into Course Values('FIN-201', 'Investment Banking', 'Finance',3);
insert into Course Values('HIS-351', 'World History', 'History',3);

insert into Course Values('MU-199', 'Music Video Production', 'Music',3);
insert into Course Values('PHY-101', 'Physical Principles', 'Physics',4);

select * from Course;

```

Output:

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure of 'LAPTOP-CLORFFA0\SQLEXPRESS' (master). In the center, a query window titled '19410127680(2).sql - LAPTOP-CLORFFA0\SQLEXPRESS master (LAPTOP-CLORFFA0\Admin (53)) - Microsoft SQL Server Management Studio' contains the SQL script provided above. The 'Results' tab shows the output of the 'select * from Course;' command, displaying 13 rows of course information in a table.

COURSE_ID	TITLE	DEPT_NAME	CREDITS
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Computer. Sci.	4
CS-190	Game Design	Computer. Sci.	4
CS-315	Robotics	Computer. Sci.	3
CS-319	Image Processing	Computer. Sci.	3
CS-347	Database System Concepts	Computer. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

At the bottom of the screen, a status bar indicates: 'Query executed successfully.' followed by the connection details: 'LAPTOP-CLORFFA0\SQLEXPRESS ... | LAPTOP-CLORFFA0\Admin ... | master | 00:00:01 | 13 rows'.

3. Insert data into **Prereq** table with the data below:

<i>course_id</i>	<i>prereq_id</i>
BIO-301	BIO-101
BIO-399	BIO-101
CS-190	CS-101
CS-315	CS-101
CS-319	CS-101
CS-347	CS-101
EE-181	PHY-101

Query:

```
INSERT INTO PREREQ  
VALUES('BIO-301', 'BIO-101'),  
      ('BIO-399', 'BIO-101'),  
      ('CS-190', 'CS-101'),  
      ('CS-315', 'CS-101'),  
      ('CS-319', 'CS-101'),  
      ('CS-347', 'CS-101'),  
      ('EE-181', 'PHY-101');
```

```
select * from prereq;
```

Output:

1941012768(2).sql - LAPTOP-CLORFFAO\SQLEXPRESS.master (LAPTOP-CLORFFAO\Admin (53)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

New Query Save All Open All Close All Refresh

master

Execute

Object Explorer

Connect ▾ master

LAPTOP-CLORFFAO\SQLEXPRESS (SQL Server)

Databases System Databases master

Tables System Tables External Tables Graph Tables

Course COURSE DEPARTMENT INSTRUCTOR PREREQ TEACHES

Views Synonyms Programmability Service Broker Storage Security

model msdb tempdb

Database Snapshots

Security Server Objects Replication PolyBase Management XEvent Profiler

1941012768(2).sql - ORFFAO\Admin (53) - 1941012768.sql - ORFFAO\Admin (60)

```
insert into Course Values('FIN-401', 'Investment Banking', 'Finance', 3);
insert into Course Values('HIS-351', 'World History', 'History', 3);

insert into Course Values('MU-199', 'Music Video Production', 'Music', 3);
insert into Course Values('PHY-101', 'Physical Principles', 'Physics', 4);

--4
INSERT INTO PREREQ
VALUES('BIO-301', 'BIO-101'),
('BIO-399', 'BIO-101'),
('CS-190', 'CS-101'),
('CS-315', 'CS-101'),
('CS-319', 'CS-101'),
('CS-347', 'CS-101'),
('EE-181', 'PHY-101');

select * from prereq;
```

Results Messages

COURSE_ID	PREREQ_ID
BIO-301	BIO-101
BIO-399	BIO-101
CS-190	CS-101
CS-315	CS-101
CS-319	CS-101
CS-347	CS-101
EE-181	PHY-101

Query executed successfully.

4. Insert data into **Department** table with the data below:

<i>dept_name</i>	<i>building</i>	<i>budget</i>
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Query:

```
INSERT INTO DEPARTMENT
VALUES('BIOLOGY', 'WATSON', 90000),
      ('COMP. SCI.', 'TAYLOR', 100000),
      ('ELEC. ENG.', 'TAYLOR', 85000),
      ('FINANCE', 'PAINTER', 120000),
      ('HISTORY', 'PAINTER', 80000),
      ('MUSIC', 'PACKARD', 80000),
      ('PHYSICS', 'WATSON', 70000);
```

```
select * from DEPARTMENT
```

Output:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the master database is selected. In the center, two queries are running in separate tabs:

- Tab 1 (1941012768(2).sql) contains the insertion script:

```
INSERT INTO DEPARTMENT
VALUES('BIOLOGY', 'WATSON', 90000),
      ('COMP. SCI.', 'TAYLOR', 100000),
      ('ELEC. ENG.', 'TAYLOR', 85000),
      ('FINANCE', 'PAINTER', 120000),
      ('HISTORY', 'PAINTER', 80000),
      ('MUSIC', 'PACKARD', 80000),
      ('PHYSICS', 'WATSON', 70000);
```

- Tab 2 (1941012768.sql) contains the select statement:

```
select * from DEPARTMENT
```

The results pane at the bottom shows the output of the select statement:

DEPARTMENT_NAME	BUILDING	BUDGET
BIOLOGY	WATSON	90000
COMP. SCI.	TAYLOR	100000
ELEC. ENG.	TAYLOR	85000
FINANCE	PAINTER	120000
HISTORY	PAINTER	80000
MUSIC	PACKARD	80000
PHYSICS	WATSON	70000

A status bar at the bottom indicates "Query executed successfully." and "LAPTOP-CLORFFA0\SQLEXPRESS ... LAPTOP-CLORFFA0\Admin... master 00:00:00 | 7 rows".

5. Insert data into Teaches table with the data below:

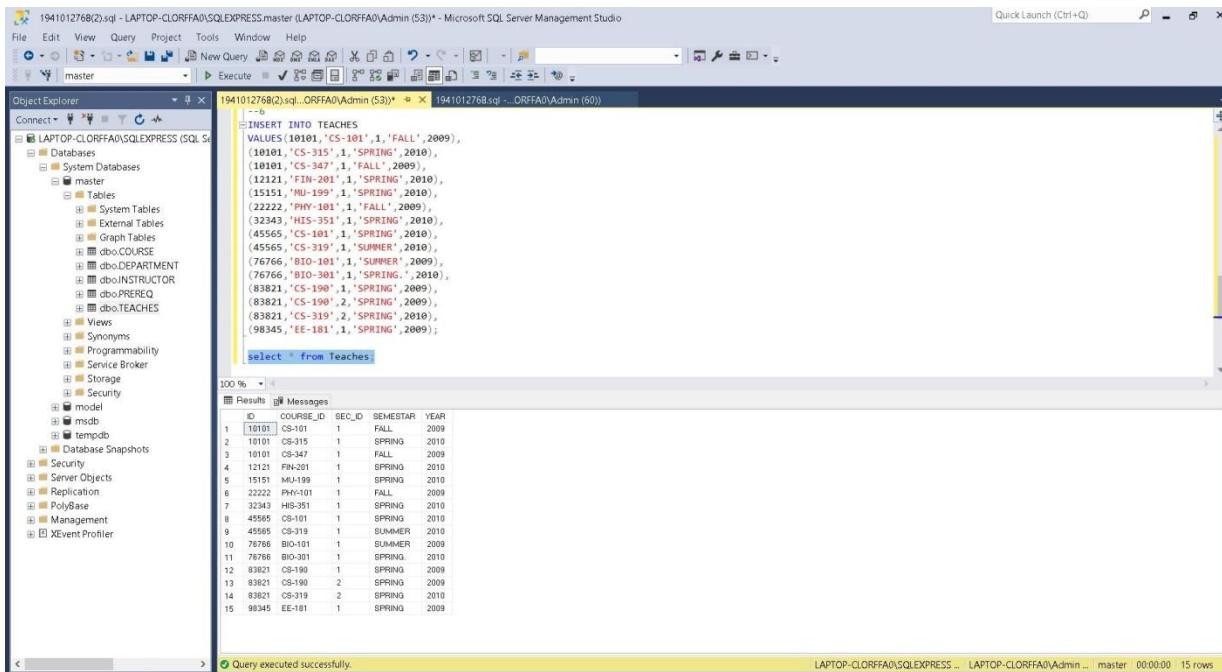
<i>ID</i>	<i>course_id</i>	<i>sec_id</i>	<i>semester</i>	<i>year</i>
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

Query:

```
INSERT INTO TEACHES
VALUES(10101, 'CS-101', 1, 'FALL', 2009),
(10101, 'CS-315', 1, 'SPRING', 2010),
(10101, 'CS-347', 1, 'FALL', 2009),
(12121, 'FIN-201', 1, 'SPRING', 2010),
(15151, 'MU-199', 1, 'SPRING', 2010),
(22222, 'PHY-101', 1, 'FALL', 2009),
(32343, 'HIS-351', 1, 'SPRING', 2010),
(45565, 'CS-101', 1, 'SPRING', 2010),
(45565, 'CS-319', 1, 'SUMMER', 2010),
(76766, 'BIO-101', 1, 'SUMMER', 2009),
(76766, 'BIO-301', 1, 'SPRING', 2010),
(83821, 'CS-190', 1, 'SPRING', 2009),
(83821, 'CS-190', 2, 'SPRING', 2009),
(83821, 'CS-319', 2, 'SPRING', 2010),
(98345, 'EE-181', 1, 'SPRING', 2009);
```

```
select * from Teaches;
```

Output:



ASSIGNMENT –2

1. Write the SQL Expressions for the following SQL Queries and verify it in your PC by executing it.

- a. Display the structure of all the tables you have created in assignment (1).

Ans.

EXEC SP_TABLES Course;

EXEC SP_TABLES Department;

EXEC SP_TABLESINSTRUCTOR;

EXEC SP_TABLESPRE REQUIRED;

EXEC SP_TABLESTeaches;

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the master database is selected. In the center pane, there are five result sets from the executed queries:

- Result set 1 (Course):

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	REMARKS
master	dbo	COURSE	TABLE	NULL
- Result set 2 (Department):

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	REMARKS
master	dbo	DEPARTMENT	TABLE	NULL
- Result set 3 (Instructor):

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	REMARKS
master	dbo	INSTRUCTOR	TABLE	NULL
- Result set 4 (Prereq):

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	REMARKS
master	dbo	PREREQ	TABLE	NULL
- Result set 5 (Teaches):

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	REMARKS
master	dbo	TEACHES	TABLE	NULL

At the bottom of the results pane, a message states "Query executed successfully." and "LAPTOP-CLORFFA\SQLEXPRESS | LAPTOP-CLORFFA\Admin | master | 00:00:00 | 5 rows".

- b. Display the contents of all the tables you have created in assignment (1).

Ans.

EXEC SP_COLUMNS Course;

EXEC SP_COLUMNS Department;

EXEC SP_COLUMNS INSTRUCTOR;

EXEC SP_COLUMNS Prereq;

EXEC SP_COLUMNS Teaches;

1941012768(4).sql - LAPTOP-CLORFFA0\SQLEXPRESS.master (LAPTOP-CLORFFA0\Admin (54)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

New Query Execute

master 1941012768(4).sql... ORFFAO\Admin (54) 1941012768(2).sql... ORFFAO\Admin (52)

Object Explorer

Connect Databases master System Tables External Tables dbo Tables dbo.COURSE dbo.DEPARTMENT dbo.INSTRUCTOR dbo.PREQRTEACHES Views Synonyms Programmability Service Broker Storage Security model msdb tempdb Database Snapshots Server Objects Replication PolyBase Management XEvent Profiler

1941012768(4).sql... ORFFAO\Admin (54)* 1941012768(2).sql... ORFFAO\Admin (52)

```

--2(a)
EXEC SP_TABLES Course;
EXEC SP_TABLES Department;
EXEC SP_TABLES Instructor;
EXEC SP_TABLES Prereq;
EXEC SP_TABLES Teaches;
--2(b)
EXEC SP_COLUMNS Course;
EXEC SP_COLUMNS Department;
EXEC SP_COLUMNS Instructor;
EXEC SP_COLUMNS Prereq;
EXEC SP_COLUMNS Teaches;

```

Results Messages

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH	SCALE	RADIX	NULLABLE	REMARKS	COLUMN_DEF	SOL_DATA_TYPE	SOL_DATETIME_SUB	CHAR_OCTET_LENGTH
1	master	dbo	COURSE	COLNAME_ID	varchar	10	10	0	1	NULL	NULL	12	NULL	10	
2	master	dbo	COURSE	TITLE	varchar	30	30	0	1	NULL	NULL	12	NULL	30	
3	master	dbo	COURSE	DEPT_NAME	varchar	12	20	0	1	NULL	NULL	12	NULL	20	
4	master	dbo	COURSE	CREDITS	numeric	2	4	0	1	NULL	NULL	2	NULL	NULL	

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH	SCALE	RADIX	NULLABLE	REMARKS	COLUMN_DEF	SOL_DATA_TYPE	SOL_DATETIME_SUB	CHAR_OCTET_LENGTH	
1	master	dbo	DEPARTMENT	DEPARTMENT_NAME	varchar	20	20	0	1	NULL	NULL	12	NULL	20		
2	master	dbo	DEPARTMENT	BUILDING	varchar	12	20	0	1	NULL	NULL	12	NULL	20		
3	master	dbo	DEPARTMENT	BUDGET	numeric	2	22	0	10	1	NULL	NULL	2	NULL	NULL	

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH	SCALE	RADIX	NULLABLE	REMARKS	COLUMN_DEF	SOL_DATA_TYPE	SOL_DATETIME_SUB	CHAR_OCTET_LENGTH	
1	master	dbo	INSTRUCTOR	ID	numeric	5	7	0	10	1	NULL	NULL	2	NULL	NULL	
2	master	dbo	INSTRUCTOR	NAME	varchar	30	30	0	1	NULL	NULL	12	NULL	30		
3	master	dbo	INSTRUCTOR	DEPT_NAME	varchar	12	20	0	1	NULL	NULL	12	NULL	20		
4	master	dbo	INSTRUCTOR	SALARY	numeric	6	8	0	10	1	NULL	NULL	2	NULL	NULL	

Query executed successfully.

LAPTOP-CLORFFA0\SQLEXPRESS... LAPTOP-CLORFFA0\Admin... master 00:00:00 18 rows

c. Display the name and department of each instructor.

Ans.

SELECT Name,DEPT_NAME FROM INSTRUCTOR;

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure. The main pane contains a query window with the following content:

```
--2(a)
EXEC SP_TABLES Course;
EXEC SP_TABLES Department;
EXEC SP_TABLES Instructor;
EXEC SP_TABLES Prereq;
EXEC SP_TABLES Teaches;
--2(b)
EXEC SP_COLUMNS Course;
EXEC SP_COLUMNS Department;
EXEC SP_COLUMNS Instructor;
EXEC SP_COLUMNS Prereq;
EXEC SP_COLUMNS Teaches;
--2(c)
SELECT Name,DEPT_NAME FROM INSTRUCTOR;
```

The results pane shows the output of the query:

Name	DEPT_NAME
Binnisan	Comp Sci
Wu	Finance
Mazak	Music
ElGald	History
Gold	Physics
Katz	Comp Sci
Califeri	History
Singh	Finance
Oncik	Biology
Brandt	Comp Sci
Kim	Elec Eng

A status bar at the bottom indicates "Query executed successfully." and "LAPTOP-CLORFFA\SQLEXPRESS .. LAPTOP-CLORFFA\Admin .. master 00:00:00 12 rows".

d. Display the name and salary of Comp. Sci. Instructors.

Ans.

SELECT Name, Salary FROM INSTRUCTOR WHERE Departments='Comp. Sci.';

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure. The main pane contains a query window with the following content:

```
--2(a)
EXEC SP_TABLES Course;
EXEC SP_TABLES Department;
EXEC SP_TABLES Instructor;
EXEC SP_TABLES Prereq;
EXEC SP_TABLES Teaches;
--2(b)
EXEC SP_COLUMNS Course;
EXEC SP_COLUMNS Department;
EXEC SP_COLUMNS Instructor;
EXEC SP_COLUMNS Prereq;
EXEC SP_COLUMNS Teaches;
--2(c)
SELECT Name,DEPT_NAME FROM INSTRUCTOR;
--2(d)
SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Comp. Sci.';
```

The results pane shows the output of the query:

Name	Salary
Binnisan	65000
Katz	75000
Brandt	92000

A status bar at the bottom indicates "Query executed successfully." and "LAPTOP-CLORFFA\SQLEXPRESS .. LAPTOP-CLORFFA\Admin .. master 00:00:00 3 rows".

e. Display the records of instructor who belongs to Physics department and getting salary less than 90000.

Ans.

SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Physics' AND Salary<90000;

1941012768(4).sql - LAPTOP-CLORFFA\SQLEXPRESS.master (LAPTOP-CLORFFA\Admin (54)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

master

Object Explorer

SQLQuery2.sql - L..._ORFFA\Admin (54) 1941012768(4).sql ..._ORFFA\Admin (54) 1941012768(2).sql ..._ORFFA\Admin (52)

```
--2(a)
EXEC SP_TABLES Course;
EXEC SP_TABLES Department;
EXEC SP_TABLES Instructor;
EXEC SP_TABLES Prereq;
EXEC SP_TABLES Teaches;
--2(b)
EXEC SP_COLUMNS Course;
EXEC SP_COLUMNS Department;
EXEC SP_COLUMNS Instructor;
EXEC SP_COLUMNS Prereq;
EXEC SP_COLUMNS Teaches;
--2(c)
SELECT Name,DEPT_NAME FROM INSTRUCTOR;
--2(d)
SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Comp.Sci';
--2(e)
SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Physics' AND Salary > 90000;
```

Results Messages

Name	Salary
Gold	87000

100 % 1 row(s) returned.

Query executed successfully.

LAPTOP-CLORFFA\SQLEXPRESS ... LAPTOP-CLORFFA\Admin ... master 00:00:00 | 1 rows

- f. Display the name of the instructors who do not belong to Comp. Sci. Department.**

Ans.

SELECT Name FROM INSTRUCTOR WHERE DEPT_NAME != 'Comp. Sci.';

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'master' database is selected. In the center pane, a query window contains the following T-SQL code:

```

EXEC SP_TABLES Prereq;
EXEC SP_TABLES Teaches;
--2(b)
EXEC SP_COLUMNS Course;
EXEC SP_COLUMNS Department;
EXEC SP_COLUMNS Instructor;
EXEC SP_COLUMNS Prereq;
EXEC SP_COLUMNS Teaches;
--2(c)
SELECT Name,DEPT_NAME FROM INSTRUCTOR;
--2(d)
SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Comp.Sci';
--2(e)
SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Physics' AND Salary>90000;
--2(f)
SELECT Name FROM INSTRUCTOR WHERE DEPT_NAME != 'Comp. Sci.'
--2(g)

```

The results pane shows a table with 12 rows, each containing a name from the INSTRUCTOR table. The names listed are: Srikeshan, Wu, Mozart, Einstein, El Saad, Qud, Kafe, Colleen, Singh, Creek, Brundit, and Kim.

- g. Display the names of the different department distinctly available in Instructor table.**

Ans.

SELECT DISTINCT Departments FROM INSTRUCTOR;

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'master' database is selected. In the center pane, a query window contains the following T-SQL code:

```

EXEC SP_TABLES Prereq;
EXEC SP_TABLES Teaches;
--2(b)
EXEC SP_COLUMNS Course;
EXEC SP_COLUMNS Department;
EXEC SP_COLUMNS Instructor;
EXEC SP_COLUMNS Prereq;
EXEC SP_COLUMNS Teaches;
--2(c)
SELECT Name,DEPT_NAME FROM INSTRUCTOR;
--2(d)
SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Comp.Sci';
--2(e)
SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Physics' AND Salary>90000;
--2(f)
SELECT Name FROM INSTRUCTOR WHERE DEPT_NAME != 'Comp. Sci.'
--2(g)
SELECT DISTINCT DEPT_NAME FROM INSTRUCTOR

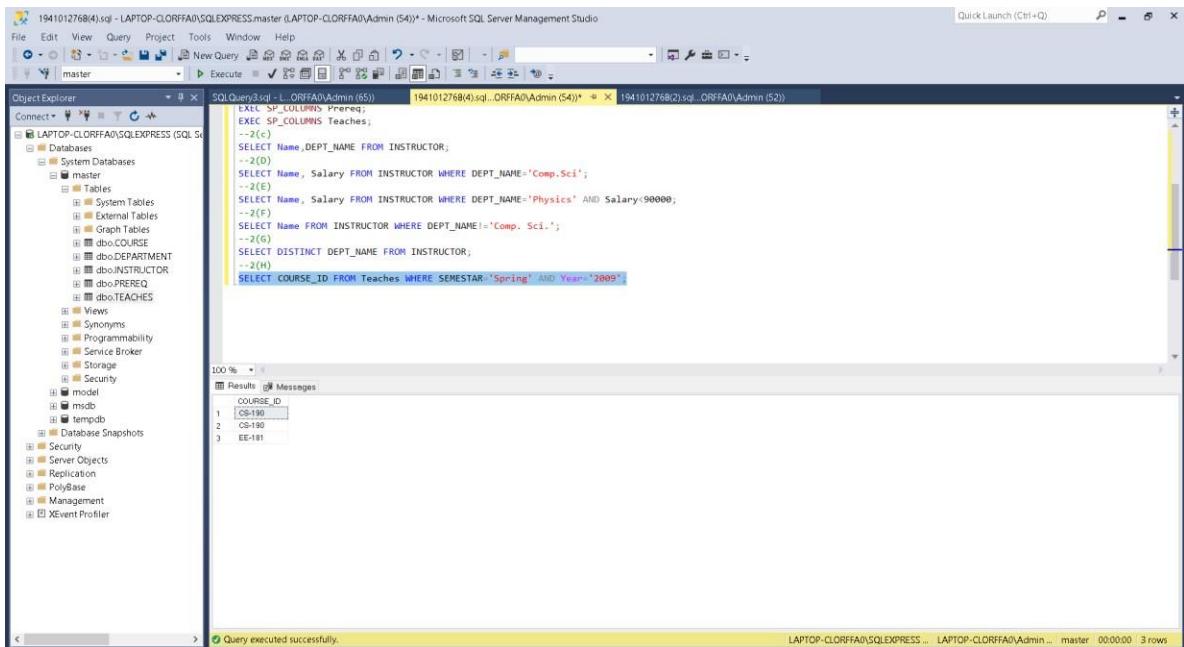
```

The results pane shows a table with 7 rows, each containing a department name from the INSTRUCTOR table. The departments listed are: Biology, Comp.sci, Elec Eng, Finance, History, Music, and Physics.

- h. Display all Course_id's that are taught in Spring semester of 2009.**

Ans.

SELECT COURSE_ID FROM Teaches WHERE Semester='Spring' AND Year='2009';



- i. Display course titles that are taught in Comp. Sci. Department and do not have credit equals to 3.

Ans.

```
SELECT TITLE FROM COURSE WHERE DEPT_NAME='Comp. Sci.' AND Credits!=3' ;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure for 'master'. The 'Results' tab on the right displays the output of the following query:

```
EXEC SP_COLUMNS Prereq;
EXEC SP_COLUMNS Teaches;
--2(c)
SELECT Name ,DEPT_NAME FROM INSTRUCTOR;
--2(D)
SELECT Name , Salary FROM INSTRUCTOR WHERE DEPT_NAME='Comp. Sci.';
--2(E)
SELECT Name , Salary FROM INSTRUCTOR WHERE DEPT_NAME='Physics' AND Salary>90000;
--2(F)
SELECT Name FROM INSTRUCTOR WHERE DEPT_NAME!= 'Comp. Sci.' ;
--2(G)
SELECT DISTINCT DEPT_NAME FROM INSTRUCTOR;
--2(H)
SELECT COURSE_ID FROM Teaches WHERE SEMESTER='Spring' AND Year='2009';
--2(I)
SELECT TITLE FROM COURSE WHERE DEPT_NAME='Comp. Sci.' AND Credits!=3' ;
```

The results table shows one row with the title 'TITLE'.

- j. Display all columns of course table sorted in descending order of department names.

Ans.

```
SELECT * FROM COURSE ORDER BY DEPT_NAME DESC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure for 'master'. The 'Results' tab on the right displays the output of the following query:

```
EXEC SP_COLUMNS Prereq;
EXEC SP_COLUMNS Teaches;
--2(c)
SELECT Name ,DEPT_NAME FROM INSTRUCTOR;
--2(D)
SELECT Name , Salary FROM INSTRUCTOR WHERE DEPT_NAME='Comp. Sci.';
--2(E)
SELECT Name , Salary FROM INSTRUCTOR WHERE DEPT_NAME='Physics' AND Salary>90000;
--2(F)
SELECT Name FROM INSTRUCTOR WHERE DEPT_NAME!= 'Comp. Sci.' ;
--2(G)
SELECT DISTINCT DEPT_NAME FROM INSTRUCTOR;
--2(H)
SELECT COURSE_ID FROM Teaches WHERE SEMESTER='Spring' AND Year='2009';
--2(I)
SELECT TITLE FROM COURSE WHERE DEPT_NAME='Comp. Sci.' AND Credits!=3' ;
--2(J)
SELECT * FROM COURSE ORDER BY DEPT_NAME DESC;
```

The results table shows 13 rows of course data, sorted by DEPT_NAME in descending order. The columns are COURSE_ID, TITLE, DEPT_NAME, and CREDITS.

COURSE_ID	TITLE	DEPT_NAME	CREDITS
PHY-101	Physical Principles	Physics	4
MU-199	Music Video Production	Music	3
HIS-351	World History	History	3
EE-101	Intro. to Computing	Finance	3
EE-161	Intro. to Digital Systems	Elect. Eng.	3
CS-101	Intro. to Computer Science	Computer Sci.	4
CG-199	Game Design	Computer Sci.	4
CS-315	Robotics	Computer Sci.	3
CS-319	Image Processing	Computer Sci.	3
CS-347	Database System Concepts	Computer Sci.	3
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3

- k. Add a date_of_join column to instructor table.

Ans.

```
ALTER TABLE INSTRUCTOR ADD Date_of_join DATE;
SELECT * FROM INSTRUCTOR;
```

1941012768(4).sql - LAPTOP-CLORFFA\SQLEXPRESS.master (LAPTOP-CLORFFA\Admin (54))- Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

master Execute

Object Explorer

SELECT Name, Salary FROM INSTRUCTOR WHERE DEPT_NAME='Physics' AND Salary<90000;
--2(F)
SELECT Name FROM INSTRUCTOR WHERE DEPT_NAME='Comp. Sci.';
--2(G)
SELECT DISTINCT DEPT_NAME FROM INSTRUCTOR;
--2(H)
SELECT COURSE_ID FROM Teaches WHERE SEMESTER='Spring' AND Year='2009';
--2(I)
SELECT TITLE FROM COURSE WHERE DEPT_NAME='Comp. Sci.' AND Credits!=3';
--2(J)
SELECT * FROM COURSE ORDER BY DEPT_NAME DESC;
--2(K)
ALTER TABLE INSTRUCTOR ADD Date_of_join DATE;
SELECT * FROM INSTRUCTOR

100 %

Results Messages

	NAME	DEPT_NAME	SALARY	Date_of_join	
1	10101	Srinivasan	Comp.Sci	85000	NULL
2	12121	Wu	Finance	90000	NULL
3	15151	Mozart	Music	40000	NULL
4	22222	Einstein	Physics	80000	NULL
5	22343	El Said	History	80000	NULL
6	33456	Gold	Physics	97000	NULL
7	44555	Cohen	Comp.Sci	75000	NULL
8	68856	Collier	History	62000	NULL
9	78543	Singh	Finance	80000	NULL
10	79766	Cristk	Biology	72000	NULL
11	83821	Brandt	Comp.sci	92000	NULL
12	98345	Kim	Elec.Eng	80000	NULL

Query executed successfully.

LAPTOP-CLORFFA\SQLEXPRESS ... LAPTOP-CLORFFA\Admin ... master 00:00:00 12 rows

I. Insert date values to existing rows.

Ans.

```
UPDATE INSTRUCTOR SET Date_of_join = '2022-05-23';
SELECT * FROM INSTRUCTOR;
```

The screenshot shows the SQL Server Management Studio interface. In the Object Explorer, the master database is selected. In the center pane, a query window contains the following code:

```
--1
SELECT * FROM COURSE ORDER BY DEPT_NAME DESC;
ALTER TABLE INSTRUCTOR ADD Date_of_join DATE;
SELECT * FROM INSTRUCTOR;
--2(L)
UPDATE INSTRUCTOR SET Date_of_join = '2022-05-23';
SELECT * FROM INSTRUCTOR;
--2(M)
--SP_RENAME 'INSTRUCTOR.Dept_name','Departments';
--SP_RENAME 'COURSE.Dept_name','Departments';
--SP_RENAME 'DEPARTMENT.Dept_name','Departments';
```

The results pane shows the output of the SELECT statement after the update:

ID	NAME	DEPT_NAME	SALARY	Date_of_join
1	10101	Srinivasan	Comp Sci	650000 2022-05-23
2	12121	Vu	Finance	90000 2022-05-23
3	15151	Mozart	Music	40000 2022-05-23
4	22221	Wu	Physics	90000 2022-05-23
5	32324	Eiseld	History	80000 2022-05-23
6	33456	Gold	Physics	87000 2022-05-23
7	45565	Katz	Comp Sci	75000 2022-05-23
8	59858	Calhoun	History	62000 2022-05-23
9	76543	Gupta	Finance	80000 2022-05-23
10	76766	Orcis	Biology	72000 2022-05-23
11	83821	Brandt	Comp Sci	92000 2022-05-23
12	98345	Kim	Elect Eng	80000 2022-05-23

A message at the bottom indicates "Query executed successfully."

m. Change the name of dept_name to department in all the tables.

Ans.

```
SP_RENAME 'INSTRUCTOR.Dept_name','Departments';
SP_RENAME 'COURSE.Dept_name','Departments';
SP_RENAME 'DEPARTMENT.Dept_name','Departments';
```

The screenshot shows the SQL Server Management Studio interface. In the Object Explorer, the master database is selected. In the center pane, a query window contains the following code:

```
--1
--SP_RENAME 'Instructor.Dept_name','Departments';
--SP_RENAME 'Course.Dept_name','Departments';
--SP_RENAME 'Department.Dept_name','Departments';
EXEC SP_COLUMNS Faculty_Info;
EXEC SP_COLUMNS Course;
```

The results pane shows the output of the EXEC_SP_COLUMNS command for the Faculty_Info and Course tables:

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH	SCALE	RADIX	NULLABLE	REMARKS	COLUMN_DEF	SQL_DA
1	S1941012089	dbo	Faculty_Info	ID	numeric	5	7	0	10	1	NULL	NULL	2
2	S1941012089	dbo	Faculty_Info	Name	varchar	50	50	NULL	NULL	1	NULL	NULL	12
3	S1941012089	dbo	Faculty_Info	Departments	varchar	20	20	NULL	NULL	1	NULL	NULL	12
4	S1941012089	dbo	Faculty_Info	Salary	numeric	6	8	0	10	1	NULL	NULL	2
5	S1941012089	dbo	Faculty_Info	Date_of_join	date	10	20	NULL	NULL	1	NULL	NULL	9

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH	SCALE	RADIX	NULLABLE	REMARKS	COLUMN_DEF	SQL_DA
1	S1941012089	dbo	Course	Sub_code	varchar	8	8	NULL	NULL	1	NULL	NULL	12
2	S1941012089	dbo	Course	Title	varchar	30	30	NULL	NULL	1	NULL	NULL	12
3	S1941012089	dbo	Course	Departments	varchar	20	20	NULL	NULL	1	NULL	NULL	12
4	S1941012089	dbo	Course	Credits	numeric	2	4	0	10	1	NULL	NULL	2

TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH	SCALE	RADIX	NULLABLE	REMARKS	COLUMN_DEF	SQL_DA
1	S1941012089	dbo	Department	Departments	varchar	20	20	NULL	NULL	1	NULL	NULL	12
2	S1941012089	dbo	Department	BUDLER	varchar	20	20	NULL	NULL	1	NULL	NULL	12
3	S1941012089	dbo	Department	Budget	numeric	10	12	0	10	1	NULL	NULL	2

A message at the bottom indicates "Query executed successfully."

n. Change the name of “Prereq” table with new name “Prerequisite”

Ans.

```
--1n
SP_RENAME 'PREREQ','Prerequisite';
```

o. Change Course_id column name to Sub_code.

Ans.

```
--1o
SP_RENAME 'COURSE.Course_id','Sub_code';
SP_RENAME 'Prerequisite.Course_id','Sub_code';
SP_RENAME 'TEACHES.Course_id','Sub_code';
```

- p. Change the data type of name to varchar2 (50).

Ans.

--1p

```
ALTER TABLE Instructor ALTER COLUMN name VARCHAR(50);
```

- q. Change the name of Instructor table to Faculty_Info.

Ans.

--1q

```
SP_rename 'Instructor', 'Faculty_Info';
```

- r. Change the Column size of Course_id in Course table from 10 to 8.

Ans.

--1r

```
ALTER TABLE Course ALTER COLUMN Sub_code VARCHAR(8);
```

- s. Delete the content of the table Prereq along with its description.

Ans.

--1s

```
DROP TABLE Prerequisite;
```

- t. Change the column name 'Building' of Department table by new column name 'Builder'.

Ans.

--1t

```
sp_rename 'DEPARTMENT.BUILDING', 'BUILDER';
```

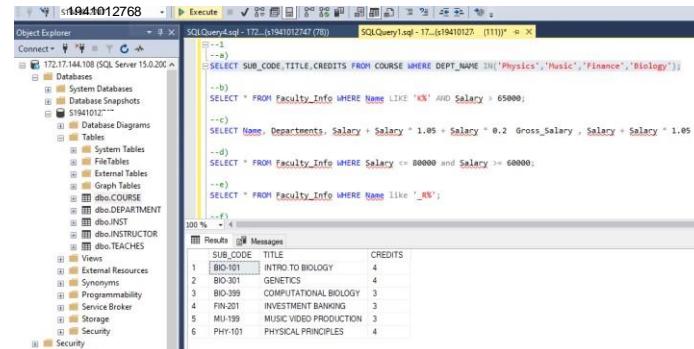
ASSINGMENT-3

1. Write the SQL Expressions for the following queries using suitable SQL operators .

a) Display the Course_ids, Titles and Credits of course that are offered in any of the departments namely: Physics, Music, Finance and Biology.

Ans.

```
SELECT SUB_CODE,TITLE,CREDITS FROM COURSE WHERE DEPT_NAME  
IN('Physics','Music','Finance','Biology');
```

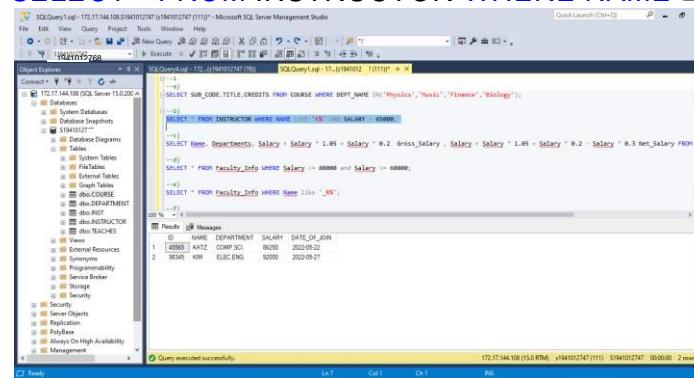


SUB_CODE	TITLE	CREDITS
BIO-101	INTRO TO BIOLOGY	4
BIO-301	GENETICS	4
BIO-399	COMPUTATIONAL BIOLOGY	3
FIN-201	INVESTMENT BANKING	3
MU-199	MUSIC VIDEO PRODUCTION	3
PHY-101	PHYSICAL PRINCIPLES	4

b) Display records of the instructors whose name starts with “K” and who get salary more than 65000.

Ans.

```
SELECT * FROM INSTRUCTOR WHERE NAME LIKE 'K%' AND SALARY > 65000;
```



NAME	DEPARTMENT	SALARY
KATZ	COMP SCI	8020
KIM	ELEC ENG	9200

c) Display name, department, gross salary and net salary of instructors with 105% DA, 20% HRA, 30% IT. (gross salary = salary + DA + HRA, net salary = gross salary – IT).

Ans.

```
SELECT NAME,DEPARTMENT,SALARY+SALARY*1.05+SALARY*0.2 GROSS_SALARY ,SALARY  
+SALARY*1.05+SALARY*0.2-SALARY*0.3NET_SALARY  
FROM INSTRUCTOR;
```

```

SELECT * FROM INSTRUCTOR WHERE SALARY <= 80000 AND SALARY >= 60000;

```

d) Display records of the Instructors with salary range 60000 to 80000. Ans.

`SELECT * FROM INSTRUCTOR WHERE SALARY <= 80000 AND SALARY >= 60000;`

```

SELECT * FROM INSTRUCTOR WHERE NAME LIKE '_R%';

```

e) Display the records of the instructors having the second letter in their name as 'r'. Ans.

`SELECT * FROM INSTRUCTOR WHERE NAME LIKE '_R%';`

```

-- (E)
SELECT * FROM INSTRUCTOR WHERE NAME LIKE '_R%';

```

f) Display the names of the instructors of Comp.Sci. Department in the descending order of their salary.

Ans.

`SELECT NAME FROM INSTRUCTOR WHERE DEPARTMENT = 'COMP.SCI.' ORDER BY SALARY;`

SQL Server Management Studio screenshot showing the results of a query:

```
--(F)
SELECT NAME FROM INSTRUCTOR WHERE DEPARTMENT = 'COMP.SCI.' ORDER BY SALARY;
```

NAME
SRINIVASAN
KATZ
BRANDT

g) Update all records of Instructor table with a salary hike of 15%. Ans.

UPDATE INSTRUCTOR SET SALARY = SALARY * 1.15;

SQL Server Management Studio screenshot showing the execution of an UPDATE query:

```
--(G)
UPDATE INSTRUCTOR SET SALARY = SALARY * 1.15;
```

(12 rows affected)

Completion time: 2022-06-27T07:55:52.1415901+05:30

h) Update the records with a salary hike of 3% for Comp.Sci. Dept instructors having salary less than 70000.

Ans.

UPDATE INSTRUCTOR SET SALARY = SALARY * 1.03 WHERE SALARY < 70000;

SQL Server Management Studio screenshot showing the execution of an UPDATE query:

```
--(H)
UPDATE INSTRUCTOR SET SALARY = SALARY * 1.03 WHERE SALARY < 70000;
```

(1 row affected)

Completion time: 2022-06-27T07:57:41.9848137+05:30

i) Display the annual salary of each

instructor. Ans.

SELECT NAME , (SALARY*12)Annual_Salary FROM INSTRUCTOR;

SQL Server Management Studio screenshot showing the results of a query:

```
--(I)
SELECT NAME , (SALARY*12)Annual_Salary FROM INSTRUCTOR;
```

NAME	Annual_Salary
SRINIVASAN	103556
WU	142300
Mozart	673464
EINSTEIN	150756
EL SAID	98772
SOLO	100056
KATZ	1190256
CAULFIELD	803940
SINGH	1269600
CRICK	112640
BRANDT	1460040
KIM	1269600

j) Update the title of the course having title 'Game Design' to 'Game Theory'. Ans.

`UPDATE COURSE SET TITLE= 'GAME THEORY' WHERE TITLE = 'GAME DESIGN';`

```
SQLQuery2.sql - 172...(S1941012747(74)) -> ASS3QUERY.sql - 17...(S1941012747(56))
-- (3)
UPDATE COURSE SET TITLE= 'GAME THEORY' WHERE TITLE = 'GAME DESIGN';

(0 rows affected)

Completion time: 2022-06-27T08:02:50.8666178+05:30
```

k) Delete the instructor records of History department. Ans.

`DELETE FROM INSTRUCTOR WHERE DEPARTMENT = 'HISTORY';`

```
SQLQuery2.sql - 172...(S1941012747(74)) -> ASS3QUERY.sql - 17...(S1941012747(56))
-- (K)
DELETE FROM INSTRUCTOR WHERE DEPARTMENT = 'HISTORY';

(2 rows affected)

Completion time: 2022-06-27T08:05:07.8251458+05:30
```

l) Delete the course records of the courses having course_id starting with 'BIO' Ans.

`DELETE FROM COURSE WHERE DEPT_NAME LIKE 'BIO%';`

```
SQLQuery2.sql - 172...(S1941012747(74)) -> ASS3QUERY.sql - 17...(S1941012747(56))
-- (L)
DELETE FROM COURSE WHERE DEPT_NAME LIKE 'BIO%';

(3 rows affected)

Completion time: 2022-06-27T08:08:28.0859560+05:30
```

2. Write the SQL Expressions for the following queries using suitable SQL aggregate function.

a) Display the Avg. salary of instructors of Physics department. Ans.

`SELECT AVG(SALARY) AVG_SALARY FROM INSTRUCTOR WHERE DEPARTMENT = 'PHYSICS';`

```
SQLQuery2.sql - 172...(S1941012747(74)) -> ASS3QUERY.sql - 17...(S1941012747(56))
-- (2)
-- (A)
SELECT AVG(SALARY) AVG_SALARY FROM INSTRUCTOR WHERE DEPARTMENT = 'PHYSICS';

AVG_SALARY
1 120348.000000
```

b) Display the Dept_ name and Average salary paid to instructor of each department Ans.

```
SELECT DEPARTMENT , AVG(SALARY) AVG_SALARY FROM INSTRUCTOR GROUP BY DEPARTMENT;
```

The screenshot shows the SQL Server Management Studio interface with a query window titled 'SQLQuery2.sql'. The query is:

```
--(B)
SELECT DEPARTMENT , AVG(SALARY) AVG_SALARY FROM INSTRUCTOR GROUP BY DEPARTMENT;
```

The results pane displays the following data:

DEPARTMENT	AVG_SALARY
BIOLOGY	95220.00000
COMP.SCI.	102273.66666
ELEC.ENG.	105800.00000
FINANCE	105800.00000
PHYSICS	120348.00000
MUSIC	56122.00000
1	119025.00000

c) Display the ID, Name & Department of the instructor drawing the highest salary. Ans.

```
SELECT ID , NAME FROM INSTRUCTOR WHERE SALARY = (SELECT MAX(SALARY)SALARY FROM INSTRUCTOR);
```

The screenshot shows the SQL Server Management Studio interface with a query window titled 'SQLQuery2.sql'. The query is:

```
--(C)
SELECT ID , NAME FROM INSTRUCTOR WHERE SALARY = (SELECT MAX(SALARY)SALARY FROM INSTRUCTOR)
```

The results pane displays the following data:

ID	NAME
22222	EINSTEIN

d) Display the number of instructors available in Comp. Sci. Department. Ans.

```
SELECT COUNT(ID)NO_OF_INSTRUCTOR FROM INSTRUCTOR WHERE DEPARTMENT = 'COMP.SCI.';
```

The screenshot shows the SQL Server Management Studio interface with a query window titled 'SQLQuery2.sql'. The query is:

```
--(D)
SELECT COUNT(ID)NO_OF_INSTRUCTOR FROM INSTRUCTOR WHERE DEPARTMENT = 'COMP.SCI.';
```

The results pane displays the following data:

NO_OF_INSTRUCTOR
3

e) Display the total credits of all courses offered in Comp.Sci. Department. Ans.

```
SELECT SUM(Credits)TOTAL_CREDITS FROM COURSE WHERE DEPT_NAME= 'COMP.SCI.';
```

The screenshot shows the SQL Server Management Studio interface with a query window titled 'SQLQuery2.sql'. The query is:

```
--(E)
SELECT SUM(Credits)TOTAL_CREDITS FROM COURSE WHERE DEPT_NAME= 'COMP.SCI.';
```

The results pane displays the following data:

TOTAL_CREDITS
14

f) Display the number of instructors and total salary drawn by Physics and Comp.Sci. departments.

Ans.

```
SELECT DEPARTMENT, COUNT(ID)NO_OF_INSTRUCTOR,SUM(SALARY)TOTAL_SALARY FROM
INSTRUCTOR
WHERE DEPARTMENT = 'PHYSICS' OR DEPARTMENT = 'COMP.SCI.' GROUP BY DEPARTMENT
```

The screenshot shows the SQL Server Management Studio interface. A query window displays the following results:

DEPARTMENT	NO_OF_INSTRUCTOR	TOTAL_SALARY
COMP.SCI.	3	306521
PHYSICS	2	240696

g) Display the total credits of Comp.Sci. and Biology departments from course table. **Ans.**

```
SELECT DEPT_NAME, SUM(CREDITS)TOTAL_CREDIT FROM COURSE WHERE DEPT_NAME =
'COMP.SCI.' OR DEPT_NAME = 'BIOLOGY' GROUP BY DEPT_NAME;
```

The screenshot shows the SQL Server Management Studio interface. A query window displays the following results:

DEPT_NAME	TOTAL_CREDIT
COMP.SCI.	14

h) Display building wise total budget values. **Ans.**

```
SELECT DEPARTMENT, COUNT(ID)NO_OF_INSTRUCTOR FROM INSTRUCTOR GROUP BY
DEPARTMENT;
```

i) Display the number of instructors of each department. **Ans.**

The screenshot shows the SQL Server Management Studio interface. A query window displays the following results:

DEPARTMENT	NO_OF_INSTRUCTOR
BIOLOGY	1
COMP SCI	3
ELEC ENG.	1
FINANACE	1
FINANCE	1
MUSIC	1
PHYSICS	2

j) Display the number of instructors of each department sorted in high to low. **Ans.**

```
SELECT DEPARTMENT, COUNT(ID)NO_OF_INSTRUCTOR FROM INSTRUCTOR GROUP BY
DEPARTMENT ORDER BY COUNT(ID);
```

```
--(1)
SELECT DEPARTMENT, COUNT(ID)NO_OF_INSTRUCTOR FROM INSTRUCTOR GROUP BY DEPARTMENT ORDER BY COUNT(ID);
```

DEPARTMENT	NO_OF_INSTRUCTOR
ELEC.ENG.	1
FINANCE	1
FINANCE	1
MUSIC	1
BIOLOGY	1
PHYSICS	2
COMP.SCI.	3

k) Display the number of courses offered semester wise. Ans.

SELECT SEMESTER, COUNT(COURSE_ID) FROM TEACHES GROUP BY SEMESTER

```
--(K)
SELECT SEMESTER, COUNT(COURSE_ID) FROM TEACHES GROUP BY SEMESTER
```

SEMESTER	(No column name)
Fall	6
Spring	20
Summer	4

l) Display the name of departments having number of instructors less than 2; Ans.

SELECT DEPARTMENT,COUNT(DEPARTMENT)NO_OF_INSTRUCTOR FROM INSTRUCTOR GROUP BY DEPARTMENT HAVING COUNT(ID)<2

```
--(L)
SELECT DEPARTMENT,COUNT(DEPARTMENT)NO_OF_INSTRUCTOR FROM INSTRUCTOR GROUP BY DEPARTMENT HAVING COUNT(ID)<2
```

DEPARTMENT	NO_OF_INSTRUCTOR
BIOLOGY	1
ELEC.ENG.	1
FINANCE	1
FINANCE	1
MUSIC	1

m) List the number of instructors of each department having 2 or more than 2 instructors except Finance department, sorted in high to low order of their number.

Ans.

SELECT DEPT_NAME,COUNT(NAME) FROM INSTRUCTOR WHERE DEPT_NAME!='FINANCE' GROUP BY(DEPT_NAME) HAVING COUNT(NAME)>=2 ORDER BY COUNT(NAME) DESC;

```
--(M)
SELECT COUNT(*) DEPARTMENT FROM INSTRUCTOR GROUP BY DEPARTMENT ORDER BY COUNT(*) DESC;
--(N)
SELECT COUNT(*),SEMESTER FROM TEACHES GROUP BY SEMESTER;
--(O)
SELECT DEPT_NAME FROM INSTRUCTOR GROUP BY DEPT_NAME HAVING COUNT(*)>=2;
--(P)
SELECT DEPT_NAME,(SUM(SALARY)) FROM INSTRUCTOR WHERE DEPT_NAME!='FINANCE' GROUP BY(DEPT_NAME) HAVING COUNT(NAME)>=2 ORDER BY COUNT(NAME) DESC;
```

DEPT_NAME	(No column name)
COMP.SCI.	6
COMP.SCI.	3
BIOLOGY	2
ELEC.ENG.	2
MUSIC	2

n) Display the Dept_name that has paid total salary more than 50000. Ans.

```
SELECT DEPT_NAME, SUM(SALARY) FROM INSTRUCTOR WHERE SALARY > 50000 GROUP BY(DEPT_NAME);
```

o) Display the total budget for the building built by Watson. Ans.

```
SELECT SUM(BUDGET) FROM DEPARTMENT WHERE BUILDING='WATSON';
```

p) Display the highest salary of the instructor of Comp.Sci. Department. Ans.

```
SELECT MAX(SALARY) FROM INSTRUCTOR WHERE DEPT_NAME='COMP.SCI';
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQueryLog - 172.17.144.108 (SQL Server 15.0.2005.1 - 5)" and "Microsoft SQL Server Management Studio". The "Object Explorer" sidebar on the left lists database objects like "AdventureworksLT", "AdventureworksLTDW", "AdventureworksLTReport", "AdventureworksLTReportDW", "AdventureworksLTReportDWReport", and "AdventureworksLTReportReport". The main pane displays a query result set:

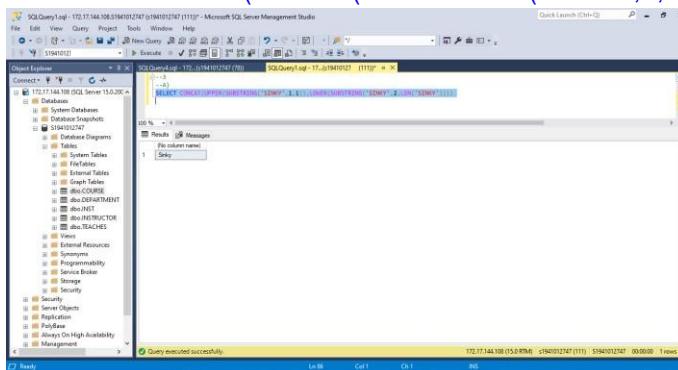
Count
100000

Below the table, a message says "Query executed successfully." The status bar at the bottom shows "172.17.144.108 (15.0 RTM) 5194510212172 (B) 5194510212172 (B) 00:00:00 | Transact-SQL | 100% |".

3. Write the SQL Expressions for the following queries using suitable SQL scalar function.

a) Display your name with first letter being capital, where the entered name is in lower case. Ans.

`SELECT CONCAT(UPPER(SUBSTRING('ANKIT',1,1)),LOWER(SUBSTRING('ANKIT',2,LEN('ANKIT'))))`



The screenshot shows the Microsoft SQL Server Management Studio interface. A query window titled 'SQLQuery1' is open with the following SQL code:

```
SELECT CONCAT(UPPER(SUBSTRING('ANKIT',1,1)),LOWER(SUBSTRING('ANKIT',2,LEN('ANKIT'))))
```

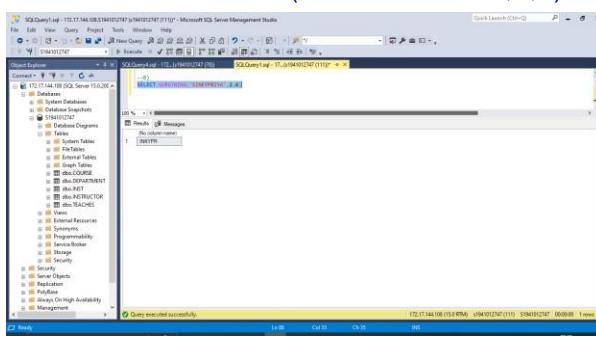
The results pane shows a single row with the output:

Result
ANKIT KUMAR

Below the results, a message box indicates: 'Query executed successfully.'

b) Display 2nd- 6th characters of your name. Ans.

`SELECT SUBSTRING('ANKIT KUMAR',2,6)`



The screenshot shows the Microsoft SQL Server Management Studio interface. A query window titled 'SQLQuery1' is open with the following SQL code:

```
SELECT SUBSTRING('ANKIT KUMAR',2,6)
```

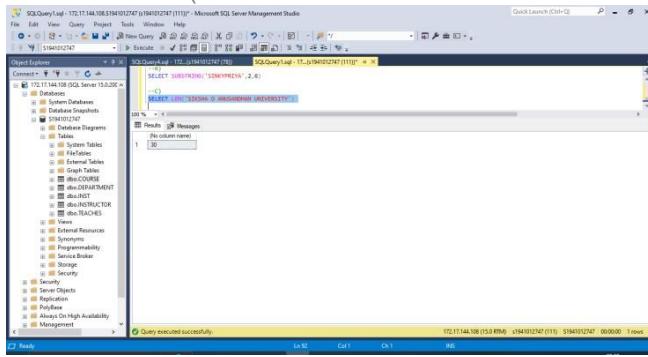
The results pane shows a single row with the output:

Result
KUMAR

Below the results, a message box indicates: 'Query executed successfully.'

c) Find length of your full university name. Ans.

`SELECT LEN('SIKSHA O ANUSANDHAN UNIVERSITY')`



The screenshot shows the Microsoft SQL Server Management Studio interface. A query window titled 'SQLQuery1' is open with the following SQL code:

```
SELECT LEN('SIKSHA O ANUSANDHAN UNIVERSITY')
```

The results pane shows a single row with the output:

Result
24

Below the results, a message box indicates: 'Query executed successfully.'

d) Display all the Instructor names with its first letter in upper case. Ans.

`SELECT CONCAT(UPPER(SUBSTRING(NAME,1,1)),(LOWER(SUBSTRING(NAME,2,LEN(NAME)))) FROM INSTRUCTOR`

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left lists the database structure for 'SCHOOL' (version 15.0.2020). The 'Messages' node under 'SCHOOL' is expanded, showing a list of 12 messages with their IDs and content:

ID	Message
1	SCHOOL
2	Wu
3	Maket
4	Ermest
5	El seal
6	Goku
7	Keto
8	Calleen
9	Dingo
10	Croki
11	Braidi
12	Kiri

The 'Messages' node is highlighted in yellow. The status bar at the bottom indicates 'Query executed successfully.' and provides connection and session details.

e) List the department name of each instructor as a three letter code. Ans.

```
SELECT SUBSTRING (DEPARTMENT,1,3) FROM INSTRUCTOR
```

f) Display the month of the joining of each instructor. Ans.

```
SELECT MONTH(DATE_OF_JOIN)DATE_OF_JOIN FROM INSTRUCTOR
```

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The title bar reads "SQLQuery14 - TDS-17.14A.S19012747 [11] - Microsoft SQL Server Management Studio". The main area displays a query window with the following T-SQL code:

```
SELECT SUBSTRING(DEPARTMENT,1,1) AS DEPT,  
       COUNT(*) AS COUNT  
  FROM INSTRUCTOR  
 GROUP BY DEPT  
 ORDER BY COUNT DESC;
```

The results pane shows the output of the query:

DEPT	COUNT
1	5
2	3
3	5
4	5
5	5
6	5
7	3
8	3
9	5
10	5
11	5
12	5

The Object Explorer on the left shows the database structure, including tables like SYSTEM_TABLES, COURSE, and INSTRUCTOR.

g) Display the date of joining of each instructor in dd/mm/yy format. Ans.

```
SELECT CONCAT(DAY(DATE_OF_JOIN), '/',  
MONTH(DATE_OF_JOIN), '//', YEAR(DATE_OF_JOIN)) FROM INSTRUCTOR
```

```

--ASSQUERY.q1-172-(19410127-'06)*
--SELECT MONTH(DATE_OF_JOIN)DATE_OF_JOIN FROM INSTRUCTOR
----(0)
--SELECT CONCAT(DAY(DATE_OF_JOIN), '/', MONTH(DATE_OF_JOIN), '/', YEAR(DATE_OF_JOIN))FROM INSTRUCTOR
----(0)

```

(No column name)
1 16/5/2022
2 17/5/2022
3 18/5/2022
4 19/5/2022
5 20/5/2022
6 21/5/2022
7 22/5/2022
8 23/5/2022
9 24/5/2022
10 25/5/2022
11 26/5/2022
12 27/5/2022

h) Display the experience of each instructor in terms of months. Ans.

`SELECT DATEDIFF(MM,DATE_OF_JOIN,GETDATE()) FROM INSTRUCTOR`

```

--ASSQUERY.q1-172-(19410127-'06)*
--SELECT MONTH(DATE_OF_JOIN)DATE_OF_JOIN FROM INSTRUCTOR
----(0)
--SELECT CONCAT(DAY(DATE_OF_JOIN), '/', MONTH(DATE_OF_JOIN), '/', YEAR(DATE_OF_JOIN))FROM INSTRUCTOR
----(0)
--SELECT DATEDIFF(MM,DATE_OF_JOIN,GETDATE()) FROM INSTRUCTOR
----(0)

```

(No column name)
1 1
2 1
3 1
4 1
5 1
6 1
7 1
8 1
9 1
10 1
11 1
12 1

i) Display the experience of each instructor in term of years and months. Ans.

`SELECT CONCAT(DATEDIFF(MM,DATE_OF_JOIN,GETDATE())/12, 'YEAR', DATEDIFF(MM,DATE_OF_JOIN,GETDATE())%12,'MONTH') FROM INSTRUCTOR`

```

--ASSQUERY.q1-172-(19410127-'06)*
--SELECT DATEDIFF(MM,DATE_OF_JOIN,GETDATE()) FROM INSTRUCTOR
----(0)
--SELECT CONCAT(DATEDIFF(MM,DATE_OF_JOIN,GETDATE())/12, 'YEAR', DATEDIFF(MM,DATE_OF_JOIN,GETDATE())%12,'MONTH') FROM INSTRUCTOR
----(0)

```

(No column name)
1 1YEAR0MONTH
2 1YEAR0MONTH
3 1YEAR0MONTH
4 1YEAR0MONTH
5 1YEAR0MONTH
6 1YEAR0MONTH
7 1YEAR0MONTH
8 1YEAR0MONTH
9 1YEAR0MONTH
10 1YEAR0MONTH
11 1YEAR0MONTH
12 1YEAR0MONTH

j) Display the date of the next Friday after today's date. Ans.

`SELECT DATEADD(DD,5,'2022-06-13')`

```
--I)
SELECT CONCAT(DATEDIFF(MM,DATE_OF_JOIN,GETDATE()),' months')

--J)
SELECT DATEADD(DD,5,'2022-06-13')

--K)

```

Results

(No column name)
1 2022-06-18 00:00:00.000

k) Display the day of joining of each instructor. Ans.

`SELECT ID,DATE_OF_JOIN FROM INSTRUCTOR`

```
--K)
SELECT ID,DATE_OF_JOIN FROM INSTRUCTOR

--L)
SELECT DATEADD(DD,15,GETDATE())

--M)

```

Results

ID	DATE_OF_JOIN
1	2022-05-16
2	2022-05-17
3	2022-05-18
4	2022-05-19
5	2022-05-20
6	2022-05-21
7	2022-05-22
8	2022-05-23
9	2022-05-24
10	2022-05-25
11	2022-05-26
12	2022-05-27

l) Display the date corresponding to 15 days after today's date. Ans.

`SELECT DATEADD(DD,15,GETDATE())`

```
--K)
SELECT ID,DATE_OF_JOIN FROM INSTRUCTOR

--L)
SELECT DATEADD(DD,15,GETDATE())

--M)

```

Results

(No column name)
1 2022-06-28 09:09:33.803

m) Display the value 94204.27348 truncated up to 2 digits after decimal point. Ans.

`SELECT CAST((94204.27348-(94204.27348%0.01)) AS DECIMAL(18,2))`

```
--N)
SELECT CAST((94204.27348 - (94204.27348 * 0.01)) AS DECIMAL(18,2))

--N)
SELECT 5+8

--O)

```

Results

(No column name)
1 94204.27

n) Display the value of the expression $5 +$

89 Ans.

SELECT 5+89

```
--N)
SELECT CAST((94204.27348 - (94204.27348 * 0.01)) AS DECIMAL(18,2))

--N)
SELECT 5+89

--O)
SELECT SQRT(6464312)

--P)
SELECT LOWER('HELLO ITER')HELLO_ITER
```

Results

(No column name)
1 94

o) Find out the square root of 6464312.

SELECT SQRT(6464312)

Ans.

```
--N)
SELECT 5+8

--O)
SELECT SQRT(6464312)
```

Results

(No column name)
1 2542.50113077654

p) Display the string “HELLO ITER” in lower case with a column heading lower case.

Ans.

SELECT LOWER('HELLO ITER')HELLO_ITER

The screenshot shows the SQL Server Management Studio (SSMS) interface. On the left is the Object Explorer tree, which includes nodes for 'Tables' (containing 'System Tables', 'FileTables', 'External Tables', 'Graph Tables', 'dbo.COURSE', 'dbo.DEPARTMENT', and 'dbo.INST'). In the center is a query editor window titled 'ASS3QUERY.sql - 172...\$194101277 (56)*'. The query is:

```
--P  
SELECT LOWER('HELLO ITER')HELLO_ITER
```

On the right is the Results pane, which displays the output of the query:

	HELLO_ITER
1	hello iter

Laboratory Assignments
Subject: Introduction to Databases
Subject code: CSE 3151

Assignment 4: Table creation using Database Constraints

- 1. Create the Tables for the following descriptions using the constraints as specified.**

TABLE NAME-- CUSTOMER

COLUMN NAME	CONSTRAINT DESCRIPTION
<i>CUST_NO</i>	-PRIMARY KEY -MUST BE 5 CHARACTER LONG & START WITH LETTER 'C'
<i>NAME</i>	NOT NULL
<i>PHONE_NO</i>	
<i>CITY</i>	NOT NULL

```
CREATE TABLE CUSTOMER (CUST_NO VARCHAR(5) PRIMARY KEY ,  
CHECK(CUST_NO LIKE 'C%'),  
NAME VARCHAR(10),  
PHONE_NO INT(10),  
CITY VARCHAR(10) NOT NULL);
```

```
mysql> DESC CUSTOMER;
```

Field	Type	Null	Key	Default	Extra
cust_no	varchar(5)	NO	PRI	NULL	
name	varchar(10)	YES		NULL	
phone_no	int	YES		NULL	
city	varchar(10)	NO		NULL	

4 rows in set (0.00 sec)

TABLE NAME-- ACCOUNT

COLUMN NAME	CONSTRAINT DESCRIPTION
ACCOUNT_NO	-PRIMARY KEY -MUST BE 5 CHARACTER LONG & START WITH LETTER 'A'
TYPE	-TYPES- 'SB', 'FD', 'CA'
BALANCE	-MUST BE < 10000000
BRANCH_CODE	-FOREIGN KEY, REFERS TABLE BRANCH

```
CREATE TABLE ACCOUNT( ACCOUNT_NO VARCHAR(5) PRIMARY KEY,  
CHECK (ACCOUNT_NO LIKE 'A%'),  
TYPE VARCHAR(2) CHECK(TYPE IN('SB','FD','CA')),  
BALANCE INT(10) CHECK(BALANCE<10000000),  
BRANCH_CODE VARCHAR(6) REFERENCES BRANCH(BRANCH_CODE)  
);
```

```
mysql> DESC ACCOUNT;
```

Field	Type	Null	Key	Default	Extra
account_no	varchar(5)	NO	PRI	NULL	
type	varchar(2)	YES		NULL	
balance	int	YES		NULL	
branch_code	varchar(6)	YES		NULL	

4 rows in set (0.00 sec)

TABLE NAME-- DEPOSITOR

COLUMN NAME	CONSTRAINT DESCRIPTION
CUST_NO	-FOREIGN KEY, REFERS TABLE CUSTOMER -PRIMARY KEY
ACCOUNT_NO	-FOREIGN KEY, REFERS TABLE ACCOUNT -PRIMARY KEY

CREATE TABLE DEPOSITOR(CUST_NO VARCHAR(5) REFERENCES CUSTOMER(CUST_NO)

,

ACCOUNT_NO VARCHAR(5) REFERENCES ACCOUNT(ACCOUNT_NO));

```
mysql> DESC DEPOSITOR;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| cust_no | varchar(5) | YES | | NULL | |
| account_no | varchar(5) | YES | | NULL | |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

TABLE NAME-- LOAN

COLUMN NAME	CONSTRAINT DESCRIPTION
LOAN-NO	-PRIMARY KEY -MUST BE 5 CHARACTER LONG & START WITH LETTER 'L'
CUST_NO	-FOREIGN KEY, REFERS TABLE CUSTOMER
AMOUNT	-MUST BE > 1000
BRANCH_CODE	-FOREIGN KEY, REFERS TABLE BRANCH

CREATE TABLE LOAN (LOAN_NO VARCHAR(5) CHECK(LOAN_NO LIKE 'L%') PRIMARY KEY,

CUST_NO VARCHAR(5) REFERENCES CUSTOMER(CUST_NO),

AMOUNT INT(10) CHECK(AMOUNT>1000),

BRANCH_CODE VARCHAR(6) REFERENCES BRANCH(BRANCH_CODE));

```
mysql> DESC LOAN;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| LOAN_NO | varchar(5) | NO | PRI | NULL | |
| CUST_NO | varchar(5) | YES | | NULL | |
| AMOUNT | int | YES | | NULL | |
| BRANCH_CODE | varchar(6) | YES | | NULL | |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

TABLE NAME-- BRANCH

COLUMN NAME	CONSTRAINT DESCRIPTION
<i>BRANCH_CODE</i>	-PRIMARY KEY
<i>BRANCH_NAME</i>	-NOT NULL
<i>BRANCH_CITY</i>	-ANY METRO CITY OF INDIA – DELHI, MUMBAI, KOLKATA, CHENNAI.

```
CREATE TABLE BRANCH( BRANCH_CODE VARCHAR(6) PRIMARY KEY,
BRANCH_NAME VARCHAR(20) NOT NULL,
BRANCH_CITY VARCHAR(10) ,
CHECK(BRANCH_CITY IN('DELHI','MUMBAI','KOLKATA','CHENNAI'))
);
```

```
mysql> DESC BRANCH;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| branch_code | varchar(6) | NO | PRI | NULL | |
| branch_name | varchar(20) | NO | | NULL | |
| branch_city | varchar(10) | YES | | NULL | |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

TABLE NAME- INSTALLMENT

COLUMN NAME	CONSTRAINT DESCRIPTION
<i>INST_NO</i>	-PRIMARY KEY -MUST BE<=10
<i>LOAN_NO</i>	-PRIMARY KEY -FOREIGN KEY, REFERS TABLE LOAN
<i>INST_AMOUNT</i>	-NOT NULL

```

CREATE TABLE INSTALLMENT(
INST_NO VARCHAR(5) CHECK(INST_NO<=10),
LOAN_NO VARCHAR(5) REFERENCES LOAN(LOAN_NO),
INST_AMOUNT NUMERIC(10) NOT NULL,
CONSTRAINT COM PRIMARY KEY(INST_NO,LOAN_NO)
);

```

```

mysql> CREATE TABLE INSTALLMENT(
-> INST_NO VARCHAR(5) CHECK(INST_NO<=10),
-> LOAN_NO VARCHAR(5) REFERENCES LOAN(LOAN_NO),
-> INST_AMOUNT NUMERIC(10) NOT NULL,
-> CONSTRAINT COM PRIMARY KEY(INST_NO,LOAN_NO)
-> );

```

Query OK, 0 rows affected (0.04 sec)

```
mysql> DESC INSTALLMENT;
```

Field	Type	Null	Key	Default	Extra
INST_NO	varchar(5)	NO	PRI	NULL	
LOAN_NO	varchar(5)	NO	PRI	NULL	
INST_AMOUNT	decimal(10,0)	NO		NULL	

3 rows in set (0.01 sec)

2. Insert the records to the tables as specified:

CUSTOMER			
CUST_NO	NAME	PHONE_NO	CITY
C0001	RAJ ANAND SINGH	9861258466	DELHI
C0002	ANKITA SINGH	9879958651	BANGALORE
C0003	SOUMYA JHA	9885623344	MUMBAI
C0004	ABHIJIT MISHRA	9455845425	MUMBAI
C0005	YASH SARAF	9665854585	KOLKATA
C0006	SWAROOP RAY	9437855466	CHENNAI
C0007	SURYANARAYAN	9937955212	GURGAON
C0008	PRANAV PRAVEEN	9336652441	PUNE
C0009	STUTI MISRA	7870266534	DELHI
C0010	ASLESHA TIWARI		MUMBAI

```

INSERT INTO CUSTOMER VALUES('C0001','RAJ ANAND
SINGH','9861258466','DELHI');
INSERT INTO CUSTOMER VALUES('C0002','ANKITA
SINGH','9879958651','BANGALORE');
INSERT INTO CUSTOMER VALUES('C0003','SOUMYA JHA','9885623344','MUMBAI');
INSERT INTO CUSTOMER VALUES('C0004','ABHIJIT
MISHRA','9455845425','MUMBAI');
INSERT INTO CUSTOMER VALUES('C0005','YASH

```

```

SARAF','9665854585','KOLKOTA');
INSERT INTO CUSTOMER VALUES('C0006','SWAROOP
RAY','9437855466','CHENNAI');
INSERT INTO CUSTOMER VALUES('C0007','SURYA
NARAYAN','9937955212','GURGAON');
INSERT INTO CUSTOMER VALUES('C0008','PRANAB
PRAVEEN','9336652441','PUNE');
INSERT INTO CUSTOMER VALUES('C0009','STUTI MISRA','7870266534','DELHI');
INSERT INTO CUSTOMER VALUES('C0010','ASLESHA TIWARI',NULL,'MUMBAI');

```

```

mysql> select * from customer;
+-----+-----+-----+-----+
| cust_no | name           | phone_no | city      |
+-----+-----+-----+-----+
| C0001   | RAJ ANAND SINGH | 9861258466 | DELHI    |
| C0002   | ANKITA SINGH    | 9879958651 | BANGALORE|
| C0003   | SOUMYA JHA       | 9885623344 | MUMBAI   |
| C0004   | ABHIJT MISHRA   | 9455845425 | MUMBAI   |
| C0005   | YASH SARAF       | 9665854585 | KOLKOTA  |
| C0006   | SWAROOP RAY      | 9437855466 | CHENNAI  |
| C0007   | SURYA NARAYAN    | 9937955212 | GURGAON  |
| C0008   | PRANAB PRAVEEN  | 9336652441 | PUNE     |
| C0009   | STUTI MISRA      | 7870266534 | DELHI    |
| C0010   | ASLESHA TIWARI   | NULL       | MUMBAI   |
+-----+-----+-----+-----+
10 rows in set (0.00 sec)

```

ACCOUNT

ACCOUNT_NO	TYPE	BALANCE	BRANCH_CODE
A0001	SB	200000	B003
A0002	SB	15000	B002
A0003	CA	850000	B004
A0004	CA	35000	B004
A0005	FD	28500	B005
A0006	FD	550000	B005
A0007	SB	48000	B001
A0008	SB	7200	B002
A0009	SB	18750	B003
A0010	FD	99000	B004

```

INSERT INTO ACCOUNT VALUES('A0001','SB',200000,'B003');
INSERT INTO ACCOUNT VALUES('A0002','SB',15000,'B002');
INSERT INTO ACCOUNT VALUES('A0003','CA',850000,'B004');
INSERT INTO ACCOUNT VALUES('A0004','CA',35000,'B004');
INSERT INTO ACCOUNT VALUES('A0005','FD',28500,'B005');
INSERT INTO ACCOUNT VALUES('A0006','FD',550000,'B005');
INSERT INTO ACCOUNT VALUES('A0007','SB',48000,'B001');
INSERT INTO ACCOUNT VALUES('A0008','SB',7200,'B002');
INSERT INTO ACCOUNT VALUES('A0009','SB',18750,'B003');
INSERT INTO ACCOUNT VALUES('A0010','FD',99000,'B004');

```

```

mysql> select * from account;
+-----+-----+-----+-----+
| account_no | type | balance | branch_code |
+-----+-----+-----+-----+
| A0001      | SB   | 200000 | B003      |
| A0002      | SB   | 15000  | B002      |
| A0003      | CA   | 850000 | B004      |
| A0004      | CA   | 35000  | B004      |
| A0005      | FD   | 28500  | B005      |
| A0006      | FD   | 550000 | B005      |
| A0007      | SB   | 48000  | B001      |
| A0008      | SB   | 7200   | B002      |
| A0009      | SB   | 18750  | B003      |
| A0010      | FD   | 99000  | B004      |
+-----+-----+-----+-----+
10 rows in set (0.00 sec)

```

DEPOSITOR

<i>CUST_NO</i>	<i>ACCOUNT_NO</i>
C0003	A0001
C0004	A0001
C0004	A0002
C0006	A0003

C0006	A0004
C0001	A0005
C0002	A0005
C0010	A0006
C0009	A0007
C0008	A0008
C0007	A0009
C0006	A0010

```
INSERT INTO DEPOSITOR VALUES('C0003','A0001');
INSERT INTO DEPOSITOR VALUES('C0004','A0001');
INSERT INTO DEPOSITOR VALUES('C0004','A0002');
INSERT INTO DEPOSITOR VALUES('C0006','A0003');
INSERT INTO DEPOSITOR VALUES('C0006','A0004');
INSERT INTO DEPOSITOR VALUES('C0001','A0005');
INSERT INTO DEPOSITOR VALUES('C0002','A0005');
INSERT INTO DEPOSITOR VALUES('C0010','A0006');
INSERT INTO DEPOSITOR VALUES('C0009','A0007');
INSERT INTO DEPOSITOR VALUES('C0008','A0008');
INSERT INTO DEPOSITOR VALUES('C0007','A0009');
INSERT INTO DEPOSITOR VALUES('C0006','A0010');
```

```
mysql> select * from depositor;
+-----+-----+
| cust_no | account_no |
+-----+-----+
| C0003   | A0001    |
| C0004   | A0001    |
| C0004   | A0002    |
| C0006   | A0003    |
| C0006   | A0004    |
| C0001   | A0005    |
| C0002   | A0005    |
| C0010   | A0006    |
| C0009   | A0007    |
| C0008   | A0008    |
| C0007   | A0009    |
| C0006   | A0010    |
+-----+-----+
12 rows in set (0.00 sec)
```

LOAN

LOAN-NO	CUST_NO	AMOUNT	BRANCH_CODE
L0001	C0005	3000000	B006
L0002	C0001	50000	B005
L0003	C0002	8000000	B004
L0004	C0010	100000	B004
L0005	C0009	9500000	B005
L0006	C0008	25000	B006

```
INSERT INTO LOAN VALUES('L0001','C0005',3000000,'B006');
INSERT INTO LOAN VALUES('L0002','C0001',50000,'B005');
INSERT INTO LOAN VALUES('L0003','C0002',8000000,'B004');
INSERT INTO LOAN VALUES('L0004','C0010',100000,'B004');
INSERT INTO LOAN VALUES('L0005','C0009',9500000,'B005');
INSERT INTO LOAN VALUES('L0006','C0008',25000,'B006');
```

```
mysql> SELECT * FROM LOAN;
```

LOAN_NO	CUST_NO	AMOUNT	BRANCH_CODE
L0001	C0005	3000000	B006
L0002	C0001	50000	B005
L0003	C0002	8000000	B004
L0004	C0010	100000	B004
L0005	C0009	9500000	B005
L0006	C0008	25000	B006

6 rows in set (0.00 sec)

BRANCH

BRANCH_CODE	BRANCH_NAME	BRANCH_CITY
B001	JANAKPURI BRANCH	DELHI
B002	CHANDNICHOWK BRANCH	DELHI
B003	JUHU BRANCH	MUMBAI
B004	ANDHERI BRANCH	MUMBAI
B005	SALTLAKE BRANCH	KOLKATA
B006	SRIRAMPURAM BRANCH	CHENNAI

```
INSERT INTO BRANCH VALUES('B001','JANAKPURI BRANCH','DELHI');
INSERT INTO BRANCH VALUES('B002','CHANDINICHOWK
BRANCH','DELHI');
INSERT INTO BRANCH VALUES('B003','JUHU BRANCH','MUMBAI');
INSERT INTO BRANCH VALUES('B004','ANDHERI BRANCH','MUMBAI');
INSERT INTO BRANCH VALUES('B005','SALTLAKE BRANCH','KOLKATA');
INSERT INTO BRANCH VALUES('B006','SRIRAMPUR BRANCH','CHENNAI');
SELECT * FROM BRANCH;
```

```
mysql> SELECT * FROM BRANCH;
+-----+-----+-----+
| branch_code | branch_name | branch_city |
+-----+-----+-----+
| B001 | JANAKPURI BRANCH | DELHI |
| B002 | CHANDINICHOWK BRANCH | DELHI |
| B003 | JUHU BRANCH | MUMBAI |
| B004 | ANDHERI BRANCH | MUMBAI |
| B005 | SALTLAKE BRANCH | KOLKATA |
| B006 | SRIRAMPUR BRANCH | CHENNAI |
+-----+-----+-----+
6 rows in set (0.00 sec)
```

INSTALLMENT

INST_NO	LOAN_NO	INST_AMOUNT
1	L0005	500000
1	L0002	10000
1	L0003	50000
1	L0004	20000
2	L0005	500000
1	L0006	3000
2	L0002	10000
3	L0002	10000
2	L0003	50000
2	L0004	20000

```
INSERT INTO INSTALLMENT VALUES(1,'L0005',500000);
INSERT INTO INSTALLMENT VALUES(1,'L0002',10000);
INSERT INTO INSTALLMENT VALUES(1,'L0003',50000);
INSERT INTO INSTALLMENT VALUES(1,'L0004',20000);
INSERT INTO INSTALLMENT VALUES(2,'L0005',500000);
INSERT INTO INSTALLMENT VALUES(1,'L0006',3000);
INSERT INTO INSTALLMENT VALUES(2,'L0002',10000);
INSERT INTO INSTALLMENT VALUES(3,'L0002',10000);
INSERT INTO INSTALLMENT VALUES(2,'L0003',50000);
INSERT INTO INSTALLMENT VALUES(2,'L0004',20000);
```

```
mysql> SELECT * FROM INSTALLMENT;
+-----+-----+-----+
| INST_NO | LOAN_NO | INST_AMOUNT |
+-----+-----+-----+
| 1      | L0002  | 10000   |
| 1      | L0003  | 50000   |
| 1      | L0004  | 20000   |
| 1      | L0005  | 500000  |
| 1      | L0006  | 3000    |
| 2      | L0002  | 10000   |
| 2      | L0003  | 50000   |
| 2      | L0004  | 20000   |
| 2      | L0005  | 500000  |
| 3      | L0002  | 10000   |
+-----+-----+-----+
10 rows in set (0.00 sec)
```


Laboratory Assignments

Subject: Introduction to Databases

Subject code: CSE 3151

Assignment 5: Sub queries and Joins

1. Write the expression for the following set of queries in SQL, based on the set of schemas of Assignment (4), using concept of sub query.

- a) Find out the name, phone_no and cust_no of customer having Account_no “A0004”.

```
SELECT NAME,PHONE_NO,CUST_NO FROM CUSTOMER WHERE CUST_NO=ALL(
SELECT CUST_NO FROM DEPOSITOR WHERE ACCOUNT_NO='A0004');
```

```
mysql> select name,phone_no,cust_no from customer where cust_no=all( select cust_no from depositor where account_no='A0004');
+-----+-----+
| name | phone_no | cust_no |
+-----+-----+
| SWAROOP RAY | 9437855466 | C0006 |
+-----+-----+
1 row in set (0.00 sec)
```

- b) Find out the name of the customer who has not taken any loan.

```
SELECT NAME FROM CUSTOMER WHERE CUST_NO NOT IN (SELECT
CUST_NO FROM LOAN);
```

```
mysql> SELECT NAME FROM CUSTOMER WHERE CUST_NO NOT IN (SELECT CUST_NO FROM LOAN);
+-----+
| NAME      |
+-----+
| SOUMYA JHA |
| ABHIJIT MISHRA |
| SWAROOP RAY |
| SURYA NARAYAN |
+-----+
4 rows in set (0.00 sec)
```

- c) Find out the branch_city where “ASLESHA TIWARI” has taken a loan.

```
SELECT BRANCH_CITY FROM BRANCH WHERE BRANCH_CODE=ALL( SELECT
BRANCH_CODE FROM LOAN WHERE CUST_NO=(SELECT CUST_NO FROM
CUSTOMER WHERE NAME='ASLESHA TIWARI'));
```

```
mysql> SELECT BRANCH_CITY FROM BRANCH WHERE BRANCH_CODE=ALL( SELECT BRANCH_CODE FROM LOAN WHERE CUST_NO=(SELECT CUST_NO FROM CUSTOMER WHERE NAME='ASLESHA TIWARI'));
+-----+
| BRANCH_CITY |
+-----+
| MUMBAI |
+-----+
1 row in set (0.00 sec)
```

- d) Find out the installment details of customer named “ANKITA SINGH”.

```
SELECT INST_NO, LOAN_NO,INST_AMOUNT FROM INSTALLMENT WHERE
LOAN_NO=(SELECT LOAN_NO FROM LOAN WHERE CUST_NO=(SELECT
CUST_NO FROM CUSTOMER WHERE NAME='ANKITA SINGH'));
```

```
mysql> SELECT INST_NO, LOAN_NO,INST_AMOUNT FROM INSTALLMENT WHERE LOAN_NO=(SELECT LOAN_NO FROM LOAN WHERE CUST_NO=(SELECT CUST_NO FROM CUSTOMER WHERE NAME='ANKITA SINGH')));
+-----+-----+-----+
| INST_NO | LOAN_NO | INST_AMOUNT |
+-----+-----+-----+
| 1       | L0003   |      50000 |
| 2       | L0003   |      50000 |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

- e) Find out the branch name and branch city, in which “ABHIJIT MISHRA” has an account.

```
SELECT BRANCH_NAME,BRANCH_CITY FROM BRANCH WHERE
BRANCH_CODE=ANY( SELECT BRANCH_CODE FROM ACCOUNT WHERE
ACCOUNT_NO=ALL(SELECT ACCOUNT_NO FROM DEPOSITOR WHERE
CUST_NO=ANY(SELECT CUST_NO FROM CUSTOMER WHERE NAME='ABHIJIT
MISHRA')));
```

```
mysql> SELECT BRANCH_NAME,BRANCH_CITY FROM BRANCH WHERE BRANCH_CODE=ANY( SELECT BRANCH_CODE FROM ACCOUNT WHERE ACCOUNT_NO=ALL(SELECT ACCOUNT_NO FROM DEPOSITOR WHERE CUST_NO=ANY(SELECT CUST_NO FROM CUSTOMER WHERE NAME='ABHIJIT MISHRA')));
+-----+-----+
| BRANCH_NAME | BRANCH_CITY |
+-----+-----+
| JANAKPUR BRANCH | DELHI |
| CHAMPTIANK BRANCH | DELHI |
| JUHU BRANCH | MUMBAI |
| ANDHERI BRANCH | MUMBAI |
| SALTLAKE BRANCH | KOLKATA |
+-----+-----+
5 rows in set (0.00 sec)
```

2. Write the expression for the following set of queries in SQL, based on the set of schemas of Assignment (4), using concept of join.

- a) Find out the Loan_nos where the loans are taken from any branch with branch_city =MUMBAI.

```
SELECT LOAN_NO FROM LOAN WHERE BRANCH_CODE=ANY(SELECT
BRANCH_CODE FROM BRANCH WHERE BRANCH_CITY='MUMBAI');
```

```
mysql> SELECT LOAN_NO FROM LOAN WHERE BRANCH_CODE=ANY(SELECT BRANCH_CODE FROM BRANCH WHERE BRANCH_CITY='MUMBAI');
+-----+
| LOAN_NO |
+-----+
| L0003   |
| L0004   |
+-----+
2 rows in set (0.00 sec)
```

- b) Find the Type of the accounts available in any branch with branch_city =DELHI.

```
SELECT TYPE AS ACCOUNT_TYPE FROM ACCOUNT WHERE
BRANCH_CODE=ANY(SELECT BRANCH_CODE FROM BRANCH WHERE
BRANCH_CITY='DELHI');
```

```

mysql> SELECT TYPE AS ACCOUNT_TYPE FROM ACCOUNT WHERE BRANCH_CODE=ANY(SELECT BRANCH_CODE FROM BRANCH WHERE BRANCH_CITY='DELHI');
+-----+
| ACCOUNT_TYPE |
+-----+
| SB          |
| SB          |
| SB          |
+-----+
3 rows in set (0.00 sec)

```

- c) Find out Installment_no and Installment amount of customer with Name= RAJ ANAND SINGH.

**SELECT INST_NO,INST_AMOUNT FROM INSTALLMENT WHERE
LOAN_NO=ANY(SELECT LOAN_NO FROM LOAN WHERE CUST_NO=(SELECT CUST_NO
FROM CUSTOMER WHERE NAME='RAJ ANAND SINGH'));**

```

mysql> SELECT INST_NO,INST_AMOUNT FROM INSTALLMENT WHERE LOAN_NO=ANY(SELECT LOAN_NO FROM LOAN WHERE CUST_NO=(SELECT CUST_NO FROM CUSTOMER WHERE NAME='RAJ ANAND SINGH'));
+-----+
| INST_NO | INST_AMOUNT |
+-----+
| 1       |      10000 |
| 2       |      10000 |
| 3       |      10000 |
+-----+
3 rows in set (0.00 sec)

```

- d) Find out the Name of the customers who have paid installments of Amount 50000 against his/her loan.

**SELECT NAME FROM CUSTOMER WHERE CUST_NO=ANY(SELECT CUST_NO
FROM LOAN WHERE LOAN_NO=ALL(SELECT LOAN_NO FROM INSTALLMENT
WHERE INST_AMOUNT>50000));**

```

mysql> select name from customer where cust_no=any(select cust_no from loan where loan_no=all(select loan_no from installment where inst_amount>50000));
+-----+
| name   |
+-----+
| STUTI MISRA |
+-----+
1 row in set (0.01 sec)

```

- e) Find out the Ph_no of customers having account at branch with Branch_name equal to SALTLAKE.

**SELECT PHONE_NO FROM CUSTOMER WHERE CUST_NO=ANY(SELECT CUST_NO
FROM DEPOSITOR WHERE ACCOUNT_NO=ANY(SELECT ACCOUNT_NO FROM ACCOUNT
WHERE BRANCH_CODE=ANY(SELECT BRANCH_CODE FROM BRANCH WHERE
BRANCH_NAME='SALTLAKE BRANCH')));**

```

mysql> select phone_no from customer where cust_no=any(select cust_no from depositor where account_no=any(select account_no from account where branch_code=any(select branch_code from branch where branch_name='SALTLAKE BRANCH')));
+-----+
| phone_no |
+-----+
| 9861258466 |
| 9879958651 |
| NULL      |
+-----+
3 rows in set (0.00 sec)

```

End term project using JDBC connectivity

Objective of this Assignment:

- To design a miniature Project for a Banking Management System using Java, SQLServer and JDBC.

Requisite:

- Completion of IDB Laboratory Assignment-4
- Basic Java Programming knowledge

Overview of the Project:

A Banking Management System is to be designed, putting together the concepts learnt in theory and practised in laboratory. The Project will integrate a Java frontend menu driven program to the backend Banking Database designed in SQL Server through JDBC connectivity.

Project Description:

The Java program provides an interface to the user to access, insert, delete and update the database. The program handles user input, output to and from the database for the said operations. User should be able to do the following operations:

1. Show Customer Records:

Using this option the details of all the customers should be displayed in particular format.

2. Add Customer Record:

Using this option the user needs to provide the information such as cust_no, name, phoneno and city through user input, which will be saved in database. After that using option 1, details of all the customers will be displayed in particular format.

3. Delete Customer Record:

Using this option the user needs to provide the cust_no of a customer through user input and all the information related to that customer will be deleted from the database. After that using option 1, details of all the customers will be displayed in particular format.

4. Update Customer Information: Using this option the user needs to provide the cust_no of a customer through user input and based on the following choice the information related to the customer will be updated.

: Update name

: Update Phoneno.

: Update city

After that using option 1, details of all the customers will be displayed in particular format.

5. Show Account Details of a Customer:

Using this option the user needs to provide the cust_no of a customer through user input and all the information of that customer along with his account_no, type, balance, branch_code, branch_name and branch_city will be displayed in proper format.

6. Show Loan Details of a Customer:

Using this option the user needs to provide the cust_no of a customer through user input and all the information of that customer along with his loan_no, loan amount, branch_code, branch_name and branch_city will be displayed in proper format.

7. Deposit Money to an Account:

Using this option the user needs to provide the account_no of a customer and the amount to be deposited through user input. According to the deposited amount the updated balance will be verified in proper format using option 5.

8. Withdraw Money from an Account:

Using this option the user needs to provide the account_no of a customer and the amount to be withdraw through user input. According to the withdraw amount the updated balance will be verified in proper format using option 5.

9. Exit the Program

The operations are choice based. Appropriate option has to be chosen from a switch case based menu driven program and the operation on the database is performed accordingly. The output is displayed in the terminal screen with appropriate messages from the database. Exceptions should be handled properly by the Java program. The output should be displayed in a formatted way for clarity of understanding and visual.

```

import java.sql.*; import java.io.*; public class bankpro {
public static void main(String [] prgrocks) throws IOException
{
    int n; // n stores the number of rows affected by insert, update or
delete
    int ch,ch1;// for user choice switch case
    Connection con=null;// connection object
    Statement stmt=null;// statement object, helps to execute sql statements through java
try
{
    String uid="s1941012707",pwd="1ter1234#";
    // Load the driver class
    //Class.forName("oracle.jdbc.driver.OracleDriver");
    Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
    // Create the connection object
    String conurl="jdbc:sqlserver://172.17.144.108;databaseName=s1941012707";
//String conurl="jdbc:oracle:thin:@172.17.144.110:1521:ora11g";// predefined
con=DriverManager.getConnection(conurl,uid,pwd);           stmt=con.createStatement();
    System.out.println("\n\n***** Banking Management System*****\n\n");
BufferedReader br= new BufferedReader(new InputStreamReader(System.in));          do
{
    System.out.println("1: display the customer details");
    System.out.println("2: insert a customer's details");
    System.out.println("3: delete a customer's details");
    System.out.println("4: update a customer's details");
    System.out.println("5: show account details of a customer");
    System.out.println("6: show loan details of a customer");
    System.out.println("7: Deposit Money to an Account");
    System.out.println("8: Withdraw Money from an Account");
    System.out.println("9: EXIT\n");
    System.out.println("enter your choice");
ch=Integer.parseInt(br.readLine());           switch(ch)
{
    case 1://displaying customer records
int c=0;
    String sqlstr="select * from CUSTOMER order by CUST_NO";// sql query
ResultSet rs=stmt.executeQuery(sqlstr);// ResultSet reference is a matrix
while(rs.next())
{
    System.out.print(rs.getString(1)+"\t\t");// these indices are the column
number of the column
    System.out.print(rs.getString(2)+"\t\t");
    System.out.print(rs.getString(3)+"\t\t");
System.out.print(rs.getString(4)+"\t\t\n");           c++;
}
    System.out.println(c+" rows selected !!!\n\n");
}
}

```

break;

```

case 2://inserting records in customer table
    System.out.println("enter the cust_no of the customer");
    String cno=br.readLine(); // readLine() is used to take string type user
input
    System.out.println("enter the name of the customer");
    String name=br.readLine();
    System.out.println("enter the phone no. of the customer");
long phn=Long.parseLong(br.readLine());
    System.out.println("enter the city of the customer");
    String city=br.readLine();
    String insstr="insert into CUSTOMER
values("+cno+", '"+name+"', "+phn+", '"+city+"')"; // sql query
    n=stmt.executeUpdate(insstr); // n returns the number of rows
added
break;
System.out.println(n+" rows inserted\n\n");

case 3:// delete the details of a customer
    System.out.println("enter the cust_no of the customer");
    String cn=br.readLine(); // readLine() is used to take string type user
input
    String delstr="delete from CUSTOMER where CUST_NO='"+cn+"'"; // sql
query
    n=stmt.executeUpdate(delstr); // n returns the number of rows added
System.out.println("\n"+n+" rows deleted\n"); break;

case 4: // update the record's of a customer
    System.out.println("enter the cust_no of the customer");
    String cn4=br.readLine();
    String updstr=""; // since it is getting modified inside switch case
choice\n");
switch(ch1)
{
case 1:
    System.out.println("enter the name of the customer");
String name4=br.readLine();
    updstr="update CUSTOMER set NAME='"+name4+
where CUST_NO='"+cn4+"'"; // sql query break;
case 2:
    System.out.println("enter the phone no. of the customer");
long phn4=Long.parseLong(br.readLine());
    updstr="update CUSTOMER set PHONE_NO="+phn4+
where CUST_NO='"+cn4+"'"; // sql query break;
case 3:
    System.out.println("enter the city of the customer");
String city4=br.readLine();
    updstr="update CUSTOMER set CITY='"+city4+
where CUST_NO='"+cn4+"'"; // sql query break;
}

```

```
n=stmt.executeUpdate(updstr); // n returns the number of rows  
System.out.println("\n"+n+" rows updated\n");  
added  
break;
```

case 5: //Show Account Details of a Customer

```

        System.out.println("enter the cust_no of the customer");
        String cn5=br.readLine();
        String sqlstr5="select * from CUSTOMER natural join DEPOSITOR natural
join ACCOUNT natural join BRANCH where CUST_NO='"+cn5+"'";
        ResultSet rs5=stmt.executeQuery(sqlstr5); // ResultSet reference is a matrix
System.out.println("\nCUST_NO\tNAME\tPHONE_NO\tCITY\tACC_NO\tTYPE\tBALANCE\
\tBR_CODE\tBRANCH_NAME\tBRANCH_CITY\n");
        while(rs5.next())
        {
            System.out.print(rs5.getString("cust_no")+"\t");// these indices are the
column number of the column
            System.out.print(rs5.getString("name")+"\t");
            System.out.print(rs5.getString("phone_no")+"\t");
            System.out.print(rs5.getString("city")+"\t");
            System.out.print(rs5.getString("account_no")+"\t");
            System.out.print(rs5.getString("type")+"\t");
            System.out.print(rs5.getString("balance")+"\t");
            System.out.print(rs5.getString("branch_code")+"\t");
            System.out.print(rs5.getString("branch_name")+"\t");
            System.out.println(rs5.getString("branch_city")+"\n");
        }
        break;
case 6://Show Loan Details of a Customer
int z=0;
        System.out.println("enter the cust_no of the customer");
        String cn6=br.readLine();
        String sqlstr6="select * from CUSTOMER natural join LOAN natural
join BRANCH where CUST_NO='"+cn6+"'";
        ResultSet rs6=stmt.executeQuery(sqlstr6); // ResultSet reference is a matrix

System.out.println("\nCUST_NO\tNAME\tPHONE_NO\tCITY\tLOAN_NO\tAMOUNT\tBR_CO
DE\tBRANCH_NAME\tBRANCH_CITY\n");
        while(rs6.next())
        {
            System.out.print(rs6.getString("cust_no")+"\t");// these indices are the
column number of the column
            System.out.print(rs6.getString("name")+"\t");
            System.out.print(rs6.getString("phone_no")+"\t");
            System.out.print(rs6.getString("city")+"\t");
            System.out.print(rs6.getString("loan_no")+"\t");
            System.out.print(rs6.getString("amount")+"\t");
            System.out.print(rs6.getString("branch_code")+"\t");
            System.out.print(rs6.getString("branch_name")+"\t");
            System.out.println(rs6.getString("branch_city")+"\n");
z++;
        }
        if(z==0)
        System.out.println("congrats !!! you have no loan\n\n");
break;
case 7://Deposit Money to an Account
        System.out.println("enter the account number");
        String acc7=br.readLine();
        System.out.println("enter the amount to be deposited");
int amt7=Integer.parseInt(br.readLine());
        String sqlstr71="select BALANCE from ACCOUNT where ACCOUNT_NO='"+acc7+"'";
```

```
ResultSet rs71=stmt.executeQuery(sqlstr71); // ResultSet reference is a matrix
```

```
System.out.print("Previous balance is: \t");
```

```

        while(rs71.next())
            System.out.println(rs71.getString("balance")+"\n");
updstr="update ACCOUNT set BALANCE=BALANCE+"+amt7+" where
ACCOUNT_NO='"+acc7+"'"; // sql query
n=stmt.executeUpdate(updstr); // n returns the number of rows added
//System.out.println("\n"+n+" rows updated\n");
String sqlstr72="select BALANCE from ACCOUNT where ACCOUNT_NO='"+acc7+"'";
ResultSet rs72=stmt.executeQuery(sqlstr72); // ResultSet reference is a
matrix
System.out.print("Updated balance is: \t");
while(rs72.next())

System.out.println(rs72.getString("balance")+"\n"); break;
case 8://Withdraw Money from an Account
int bal8=0;
System.out.println("enter the account number");
String acc8=br.readLine();
System.out.println("enter the amount to be withdraw");
int amt8=Integer.parseInt(br.readLine());
String sqlstr81="select BALANCE from ACCOUNT where ACCOUNT_NO='"+acc8+"'"; 
ResultSet rs81=stmt.executeQuery(sqlstr81); // ResultSet reference is a
matrix
System.out.print("Previous balance is: \t");
while(rs81.next())
{
    System.out.println(rs81.getString("balance")+"\n");
bal8=Integer.parseInt(rs81.getString("balance"));
}
if(bal8>=amt8)
{
    updstr="update ACCOUNT set BALANCE=BALANCE-"+amt8+" where
ACCOUNT_NO='"+acc8+"'"; // sql query
n=stmt.executeUpdate(updstr); // n returns the number of rows added
//System.out.println("\n"+n+" rows updated\n");
String sqlstr82="select BALANCE from ACCOUNT where
ACCOUNT_NO='"+acc8+"'"; 
ResultSet rs82=stmt.executeQuery(sqlstr82); // ResultSet reference is a
matrix
System.out.print("Updated balance is: \t");
while(rs82.next())
{
    System.out.println(rs82.getString("balance")+"\n");
}
else
    System.out.println("Insufficient Balance !!!!!\n");
break;
case 9: //exit case
stmt.close(); con.close();
System.out.println("\nThank
you\n");
System.exit(0);
default:
System.out.println("\nWrong choice\n");
}

```

```
    } // end of switch case  
} // end of do block  
while(ch!=9);  
} // end of try block
```

```
catch(Exception e)
```

```
{  
    System.out.println(e);
```

Test Cases:

1. The program should able to produce correct answer or appropriate error message corresponding to the following test cases:

Show Customer Records:

The screenshot shows a Java application window titled "BankingSystem [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (20-Jul-2022, 10:22:53 pm)". The window contains a menu bar with "Problems", "Javadoc", "Declaration", and "Console". The console tab displays the following output:

```
*****Banking Management System*****  
1 - Display Customer Records  
2 - Add Customer Record  
3 - Delete Customer Record  
4 - Update Customer Record  
5 - Display Account Details  
6 - Display Loan Details  
7 - Deposit Money  
8 - Withdraw Money  
9 - Exit  
Enter your choice(1-9):  
1  
C0001 RAJ ANAND SINGH 9861258467 DELHI  
C0002 ANKITA SINGH 9879958651 BANGALORE  
C0003 SOUMYA JHA 9885623344 MUMBAI  
C0004 ABHIJIT MISHRA 9455845425 MUMBAI  
C0005 YASH SARAF 9665854585 KOLKATA  
C0006 SWAROOP RAY 9437855466 CHENNAI  
C0007 SURYA NARAYAN 9937955212 GURGAON  
C0008 PRANAV PRAVEEN 9336652441 PUNE  
C0009 STUTI MISRA 7870266534 DELHI  
C0010 ASLESHA TIWARI null MUMBAI
```

2. Add Customer Record:

<C0011, ANWESHA DAS, 9999999999, BHUB>

The screenshot shows a Java application window titled "BankingSystem [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (20-Jul-2022, 10:22:53 pm)". The window contains a menu bar with "Problems", "Javadoc", "Declaration", and "Console". The console tab displays the following output:

```
*****Banking Management System*****  
1 - Display Customer Records  
2 - Add Customer Record  
3 - Delete Customer Record  
4 - Update Customer Record  
5 - Display Account Details  
6 - Display Loan Details  
7 - Deposit Money  
8 - Withdraw Money  
9 - Exit  
Enter your choice(1-9):  
2  
Enter Cust_no  
C0011  
Enter Cust_Name  
ANWESHA DAS  
Enter phone number  
9999999999  
Enter City  
BHUB  
Successfully Inserted the Customer Details
```

< C0012,SACHIN SINGH, 9898989898, CTC>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
2
Enter Cust_no
C0012
Enter Cust_Name
SACHIN SINGH
Enter phone number
9898989898
Enter City
CTC
Successfully Inserted the Customer Details
```

<C0013, ARJUN MISHRA, 7777777777, BBSR>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
2
Enter Cust_no
C0013
Enter Cust_Name
ARJUN MISHRA
Enter phone number
7777777777
Enter City
BBSR
Successfully Inserted the Customer Details
```

C0001	RAJ ANAND SINGH	9861258467	DELHI
C0002	ANKITA SINGH	9879958651	BANGALORE
C0003	SOUMYA JHA	9885623344	MUMBAI
C0004	ABHIJIT MISHRA	9455845425	MUMBAI
C0005	YASH SARAF	9665854585	KOLKATA
C0006	SWAROOP RAY	9437855466	CHENNAI
C0007	SURYA NARAYAN	9937955212	GURGAON
C0008	PRANAV PRAVEEN	9336652441	PUNE
C0009	STUTI MISRA	7870266534	DELHI
C0010	ASLESHA TIWARI	null	MUMBAI
C0011	ANWESHA DAS	9999999999	BHUB
C0012	SACHIN SINGH	9898989898	CTC
C0013	ARJUN MISHRA	7777777777	BBSR

3 Delete Customer Record:

<C0013>

```
*****Banking Management System*****
```

- 1 - Display Customer Records
- 2 - Add Customer Record
- 3 - Delete Customer Record
- 4 - Update Customer Record
- 5 - Display Account Details
- 6 - Display Loan Details
- 7 - Deposit Money
- 8 - Withdraw Money
- 9 - Exit

Enter your choice(1-9):

3

Enter the CUST_NO

C0013

Successfully Deleted

<C0016>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
3
Enter the CUST_NO
C0016
No such Cust id found
```

4 Update Customer Record for any attribute except Customer Number:

<C0011> [Update each column once]

Update Customer Name:

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
4
Enter 1: For Name 2: For Phone no 3: For City to update:
1
Enter CUST_NO
C0012
State the updated Name
SAHIL KANUNGO
Customer name updated successfully
```

Update Customer Phone Number:

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
4
Enter 1: For Name 2: For Phone no 3: For City to update:
2
Enter CUST_NO
c0012
State the updated Phone No
9843512902
Customer PHONE NO updated successfully
```

Update Customer City:

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
4
Enter 1: For Name 2: For Phone no 3: For City to update:
3
Enter CUST_NO
c0012
State the updated City
BPT
Customer CITY updated successfully
```

5 Show Account Details of a Customer:

<C0003>

*****Banking Management System*****

- 1 - Display Customer Records
- 2 - Add Customer Record
- 3 - Delete Customer Record
- 4 - Update Customer Record
- 5 - Display Account Details
- 6 - Display Loan Details
- 7 - Deposit Money
- 8 - Withdraw Money
- 9 - Exit

Enter your choice(1-9):

5

Enter the CUST_NO

C0003

A0001 SB 200000 JUHU BRANCH MUMBAI

<C0005>

*****Banking Management System*****

- 1 - Display Customer Records
- 2 - Add Customer Record
- 3 - Delete Customer Record
- 4 - Update Customer Record
- 5 - Display Account Details
- 6 - Display Loan Details
- 7 - Deposit Money
- 8 - Withdraw Money
- 9 - Exit

Enter your choice(1-9):

5

Enter the CUST_NO

C0005

No Account Details for the C0005 found

<C0016>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
5
Enter the CUST_NO
C0016
No Account Details for the C0016 found
```

6. Show Loan Details of a Customer:

<C0003>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
6
Enter Customer Number
C0003
Congratulation NO LOAN AMOUNT for C0003
```

<c0005>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
6
Enter Customer Number
c0005
LOAN AMOUNT of CUSTOMER_ID ( c0005 )is  3000000
```

<C0008>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
6
Enter Customer Number
C0008
LOAN AMOUNT of CUSTOMER_ID ( C0008 )is  25000
```

<C0016>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
6
Enter Customer Number
C0016
Congratulation NO LOAN AMOUNT for C0016
```

7. Deposit Money to an Account:

<A0008, 800>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
7
Enter the account number , You want to deposit
A0008
Amount of money , You want to deposit !
800
|Transaction Successfully
```

<A0005, 10000>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
7
Enter the account number , You want to deposit
A0005
Amount of money , You want to deposit !
10000
|Transaction Successfully
```

8. Withdraw Money from an Account:

<A0008, 800>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
8
Enter the account number , You want to withdraw
A0008
Amount of money , You want to withdraw!
800
Are you sure want to withdraw ??????????
Type yes or no
YES
Successfully withdraw the money
```

<A0008, 8000>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
8
Enter the account number , You want to withdraw
A0008
Amount of money , You want to withdraw!
800
Are you sure want to withdraw ??????????
Type yes or no
YES
Successfully withdraw the money
```

<A0005, 10000>

```
*****Banking Management System*****
1 - Display Customer Records
2 - Add Customer Record
3 - Delete Customer Record
4 - Update Customer Record
5 - Display Account Details
6 - Display Loan Details
7 - Deposit Money
8 - Withdraw Money
9 - Exit
Enter your choice(1-9):
8
Enter the account number , You want to withdraw
a0005
Amount of money , You want to withdraw!
10000
Are you sure want to withdraw ??????????
Type yes or no
Yes
Successfully withdraw the money
```

9. Exit the Program

*****Banking Management System*****

- 1 - Display Customer Records
- 2 - Add Customer Record
- 3 - Delete Customer Record
- 4 - Update Customer Record
- 5 - Display Account Details
- 6 - Display Loan Details
- 7 - Deposit Money
- 8 - Withdraw Money
- 9 - Exit

Enter your choice(1-9):

9

Exit Successfully

10. Enter choice 10

*****Banking Management System*****

- 1 - Display Customer Records
- 2 - Add Customer Record
- 3 - Delete Customer Record
- 4 - Update Customer Record
- 5 - Display Account Details
- 6 - Display Loan Details
- 7 - Deposit Money
- 8 - Withdraw Money
- 9 - Exit

Enter your choice(1-9):

10

No such option exists in the menu

Option ranges from 0-9