NEAR ITPB, CHANNASANDRA, BENGALURU – 560 067
Affiliated to VTU, Belagavi
Approved by AICTE, New Delhi
Recognized by UGC under 2(f) & 12(B)
Accredited by NBA & NAAC

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

V SEMESTER

MVJ19CSL56– DBMS LABORATORY WITH MINI PROJECT

ACADEMIC YEAR 2020 – 2021 [ODD]

LABORATORY MANUAL

NAM E OF THE STUDENT : ----------

BRANCH : ----------

UNIVERSITY SEAT NO. : ----------

SEMESTER & SECTION : ----------

BATCH : ----------

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION:

To create an ambiance in excellence and provide innovative emerging programs in Computer Science and Engineering and to bring out future ready engineers equipped with technical expertise and strong ethical values.

MISSION:

1. **Concepts of Computing Discipline:** To educate students at under graduate, postgraduate and doctoral levels in the fundamental and advanced concepts of computing discipline.
2. **Quality Research:** To provide strong theoretical and practical background across the Computer Science and Engineering discipline with the emphasis on computing technologies, quality research, consultancy and training's.
3. **Continuous Teaching Learning:** To promote a teaching learning process that brings advancements in Computer Science and Engineering discipline leading to new technologies and products.
4. **Social Responsibility and Ethical Values:** To inculcate professional behavior, innovative research Capabilities , leadership abilities and strong ethical values in the young minds so as to work with the commitment for the betterment of the society

PROGRAM EDUCATIONAL OBJECTIVES (PEOs):

PEO1: Current Industry Practices: Graduates will analyze real world problems and give solution using current industry practices in computing technology.

PEO2: Research and Higher Studies: Graduates with strong foundation in mathematics and engineering fundamentals that will enable graduates to pursue higher learning, R&D activities and consultancy.

PEO3: Social Responsibility: Graduates will be professionals with ethics, who will provide industry growth and social transformation as responsible citizens.

PEO4: Entrepreneur: Graduates will be able to become entrepreneur to address social, technical and business challenges.

PROGRAM OUTCOMES (POs):

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the informationto provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs):

PSO1: Programming: Ability to understand, analyze and develop computer programs in the areas related to algorithms, system software, multimedia, web design, DBMS, and networking for efficient design of computer-based systems of varying complexity.

PSO2: Practical Solution: Ability to practically provide solutions for real world problems with a broad range of programming language and open source platforms in various computing domains.

PSO3: Research: Ability to use innovative ideas to do research in various domains to solve societal problems.

COURSE OBJECTIVES:

- Foundation knowledge in database concepts, technology and practice to groom studentsinto well-informed database application developers.
- Strong practice in SQL programming through a variety of database problems.

PREREQUISITES:

- Set Theory.
- Data structures and algorithms.
- Programming language.

COURSE OUTCOMES (CO's):

CO No	CO's
C308.1	Demonstrate the creation of relational tables using DDL/DML
C308.2	Design and demonstrate the execution of simple queries retrieve information
C308.3	Demonstrate the execution of complex queries
C308.4	Design and implement a front end using modern tools
C308.5	Implement, analyze and evaluate the project developed for an application.

SOFTWARE REQUIREMENT:

1.The exercises are to be solved in an RDBMS environment like Oracle or DB2.

2. Front end may be created using either VB or VAJ or any other similar tool.

Operating System:

- Windows / Linux.

CONTENTS

Sl. No.	Part A: SQL Programming	RBTL	CO
1	**The following relations keep track of airline flight information:** **FLIGHTS (no: integer, from: string, to: string, distance: integer, Departs: time, arrives: time, price: real)** **AIRCRAFT (aid: integer, aname: string, cruisingrange: integer)** **CERTIFIED (eid: integer, aid: integer)** **EMPLOYEES (eid: integer, ename: string, salary: integer)** Note that the Employees relation describes pilots and other kinds of employees as well; every pilot is certified for some aircraft, and only pilots are certified to fly. Write each of the following queries in SQL. 1. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80, 000. 2. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified. 3. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt. 4. For all aircraft with cruising range over 1000 Kms, .find the name of the aircraft and the average salary of all pilots certified for this aircraft. 5. Find the names of pilots certified for some Boeing aircraft. 6. Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.	L3	CO 1,2,3,4

2	**Consider the Schema for a banking enterprise:** **BRANCH(branch-name:string, branch-city:string, assets:real)** **ACCOUNT(accno:int, branch-name:string, balance:real)** **DEPOSITOR(customer-name:string, accno:int)** **CUSTOMER(customer-name:string, customer-Street:string, customer-city:string)** **LOAN(loan-number:int, branch-name:string, amount:real)** **BORROWER(customer-name:string, loan-number:int)** 1. Create the above tables by properly specifying the primary keys and the foreign keys 2. Enter at least five tuples for each relation 3. Find all the customers who have at least two accounts at the Main branch. 4. Find all the customers who have an account at all the branches located in a specific city. 5. Demonstrate how you delete all account tuples at every branch located in a specific city.	L3	CO 1,2,3,4
3	**Consider the schema for College Database:** **STUDENT(USN, SName, Address, Phone, Gender)** **SEMSEC(SSID, Sem, Sec) CLASS(USN, SSID)** **SUBJECT(Subcode, Title, Sem, Credits)** **IAMARKS(USN, Subcode, SSID, Test1, Test2, Test3, FinalIA)** Write SQL queries to 1. List all the student details studying in fourth semester 'C' section. 2. Compute the total number of male and female students in each semester and in each section. 3. Create a view of Test1 marks of student USN '1MJ15CS101' in all subjects. 4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students. 5. Categorize students based on the following criterion: If FinalIA = 17 to 20 then CAT = 'Outstanding' If FinalIA = 12 to 16 then CAT = 'Average' If FinalIA< 12 then CAT = 'Weak' Give these details only for 8th semester A, B, and C section students.	L3	CO 1,2,3,4

4	**Consider the schema for Company Database:** **EMPLOYEE(SSN, Name, Address, Sex, Salary, SuperSSN, DNo)** **DEPARTMENT(DNo, DName, MgrSSN, MgrStartDate)** **DLOCATION(DNo,DLoc)** **PROJECT(PNo, PName, PLocation, DNo)** **WORKS_ON(SSN, PNo, Hours)** Write SQL queries to 1. Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controls the project. 2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise. 3. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department 4. Retrieve the name of each employee who works on all the projects controlled by department number 5 (use NOT EXISTS operator). 5. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs. 6,00,000.	L3	CO 1,2,3,4
	Part B: Mini project • For any problem selected, write the ER Diagram, apply ER-mapping rules, normalize the relations, and follow the application development process. • Make sure that the application should have five or more tables, at least one trigger and one stored procedure, using suitable front end tool. • Indicative areas include; health care, education, industry, transport, supply chain, etc.	L2	CO 1,2,3,4,5

Mapping of Course Outcomes to Program Outcomes:

CO-PO/PSO Mapping														
CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	3	1	3	1	2	1	-	1	-	-	2	2	-
CO2	3	3	2	3	2	2	-	-	1	-	-	2	2	1
CO3	3	3	2	3	2	1	-	-	1	-	-	2	1	-
CO4	3	3	2	2	2	1	-	-	-	-	-	2	1	3
CO5	3	3	2	2	1	1	1	-	-	-	-	2	1	3

High-3, Medium-2, Low-1

1. Direct Assessment

a. Continuous:

1. 2 Internal Assessment Tests→ [Average of 2 tests] -- Max. Marks 30
2. Record work and Weekly evaluation → Max. Marks 10
3. Total IA Marks → 1+2 (40 Marks)

b. Semester End Assessment:

Semester end lab examination evaluated for 60 Marks.

c. Total Marks : a + b = 100 Marks

2. Indirect Assessment

Indirect Assessment is through course exit survey

Conduction of Practical Examination:

1. All laboratory experiments from part A are to be included for practical examination.
2. Mini project has to be evaluated for 40 Marks.
3. Report should be prepared in a standard format prescribed for project work.
4. Students are allowed to pick one experiment from the lot.
5. Strictly follow the instructions as printed on the cover page of answer script.
6. Marks distribution: a) Part A: Procedure + Conduction + Viva: 09 + 42 +09 =60 Marks
7. Part B: Demonstration + Report + Viva voce = 20+14+06 = 40 Marks
8. Change of experiment is allowed only once and marks allotted to the procedure part to be made zero.

1. The following relations keep track of airline flight information:

FLIGHTS (no: integer, from: string, to: string, distance: integer, Departs: time, arrives: time, price: real)

AIRCRAFT (aid: integer, aname: string, cruisingrange: integer)

CERTIFIED (eid: integer, aid: integer)

EMPLOYEES (eid: integer, ename: string, salary: integer)

Note that the Employees relation describes pilots and other kinds of employees as well; every pilot is certified for some aircraft, and only pilots are certified to fly. Write each of the following queries in SQL.

Note that the Employees relation describes pilots and other kinds of employees as well; every pilot is certified for some aircraft, and only pilots are certified to fly. Write each of the following queries in SQL.

1. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80, 000.
2. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.
3. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.
4. For all aircraft with cruising range over 1000 Kms, .find the name of the aircraft and the average salary of all pilots certified for this aircraft.
5. Find the names of pilots certified for some Boeing aircraft.

Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.

Creation of Tables

FLIGHTS TABLE

```
CREATE TABLE flight( no INT, frm VARCHAR(20), too VARCHAR(20), distance
INT, departs VARCHAR(20), arrives VARCHAR(20), price REAL, PRIMARY KEY
(no) );
```

AIRCRAFT TABLE

```
CREATE TABLE aircraft( aid INT, aname VARCHAR(20), cruisingrange INT,
PRIMARY KEY (aid) );
```

CERTIFIED TABLE

CREATE TABLE certified(eid INT, aid INT, PRIMARY KEY (eid,aid), FOREIGN KEY (eid) REFERENCES employees (eid), FOREIGN KEY (aid) REFERENCES aircraft (aid));

EMPLOYEES TABLE

CREATE TABLE employees(eid INT, ename VARCHAR(20), salary INT, PRIMARY KEY (eid));

Insertion into tables

```
INSERT INTO flight (no,frm,too,distance,departs,arrives,price) VALUES
    (1,'Bangalore','Mangalore',360,'10:45:00','12:00:00',10000),
    (2,'Bangalore','Delhi',5000,'12:15:00','04:30:00',25000),
    (3,'Bangalore','Mumbai',3500,'02:15:00','05:25:00',30000),
    (4,'Delhi','Mumbai',4500,'10:15:00','12:05:00',35000),
    (5,'Delhi','Frankfurt',18000,'07:15:00','05:30:00',90000),
    (6,'Bangalore','Frankfurt',19500,'10:00:00','07:45:00',95000),
    (7,'Bangalore','Frankfurt',17000,'12:00:00','06:30:00',99000);
```

SQL> SELECT * FROM flight;

no	frm	too	distance	departs	arrives	price
1	Bangalore	Mangalore	360	10:45:00	12:00:00	10000
2	Bangalore	Delhi	5000	12:15:00	04:30:00	25000
3	Bangalore	Mumbai	3500	02:15:00	05:25:00	30000
4	Delhi	Mumbai	4500	10:15:00	12:05:00	35000
5	Delhi	Frankfurt	18000	07:15:00	05:30:00	90000
6	Bangalore	Frankfurt	19500	10:00:00	07:45:00	95000
7	Bangalore	Frankfurt	17000	12:00:00	06:30:00	99000

```
INSERT INTO aircraft (aid,aname,cruisingrange) values
    (123,'Airbus',1000),
    (302,'Boeing',5000),
    (306,'Jet01',5000),
    (378,'Airbus380',8000),
    (456,'Aircraft',500),
    (789,'Aircraft02',800),
    (951,'Aircraft03',1000);
```

SQL> SELECT * FROM aircraft;

Aid	Aname	Cruisingrange
123	Airbus	1000
302	Boeing	5000
306	Jet01	5000
378	Airbus380	8000
456	Aircraft	500
789	Aircraft02	800
951	Aircraft03	1000

INSERT INTO employees (eid,ename,salary) VALUES
(1,'Ajay',30000),
(2,'Ajith',85000),
(3,'Arnab',50000),
(4,'Harry',45000),
(5,'Ron',90000),
(6,'Josh',75000),
(7,'Ram',100000);

SQL> SELECT * FROM employees;

eid	ename	salary
1	Ajay	30000
2	Ajith	85000
3	Arnab	50000
4	Harry	45000
5	Ron	90000
6	Josh	75000
7	Ram	100000

INSERT INTO certified (eid,aid) VALUES
(1,123),
(2,123),
(1,302),
(5,302),
(7,302),
(1,306),
(2,306),
(1,378),
(2,378),
(4,378),

(6,456),
(3,456),
(5,789),
(6,789),
(3,951),
(1,951),
(1,789);

SQL> SELECT * FROM certified;

eid	aid
1	123
2	123
1	302
5	302
7	306
1	306
2	378
1	378
2	378
4	456
3	456
6	789
1	789
5	789
6	951
1	951
3	951

1. **Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80, 000.**

   SELECT DISTINCT a.aname FROM aircraft a,certified c,employees e WHERE a.aid=c.aid AND c.eid=e.eid AND NOT EXISTS (SELECT *FROM employees e1 WHERE e1.eid=e.eid AND e1.salary<80000);

2. **For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.**

   SELECT c.eid,MAX(cruisingrange) FROM certified c,aircraft a WHERE c.aid=a.aid GROUP BY c.eid HAVING COUNT(*)>3;

3. **Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.**

   SELECT DISTINCT e.ename FROM employees e WHERE e.salary< (SELECT MIN(f.price) FROM flight f WHERE f.frm='Bangalore' AND f.too='Frankfurt');

4. **For all aircraft with cruising range over 1000 Kms, .find the name of the aircraft and the average salary of all pilots certified for this aircraft.**

   SELECT a.aid,a.aname,AVG(e.salary) FROM aircraft a,certified c,employees e WHERE a.aid=c.aid AND c.eid=e.eid AND a.cruisingrange>1000 GROUP BY a.aid,a.aname;

5. **Find the names of pilots certified for some Boeing aircraft.**

   SELECT distinct e.ename FROM employees e,aircraft a,certified c WHERE e.eid=c.eid AND c.aid=a.aid AND a.aname='Boeing';

6. **Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.**

   SELECT a.aid FROM aircraft a WHERE a.cruisingrange> (SELECT MIN(f.distance) FROM flight f WHERE f.frm='Bangalore' AND f.too='Delhi');

2. **Consider the Schema for a banking enterprise:**
   BRANCH(branch-name:string, branch-city:string, assets:real)
   ACCOUNT(accno:int, branch-name:string, balance:real)
   DEPOSITOR(customer-name:string, accno:int)
   CUSTOMER(customer-name:string, customer-Street:string, customer-city:string)
   LOAN(loan-number:int, branch-name:string, amount:real)
   BORROWER(customer-name:string, loan-number:int)
   1. Create the above tables by properly specifying the primary keys and the foreign keys
   2. Enter at least five tuples for each relation
   3. Find all the customers who have at least two accounts at the Main branch.
   4. Find all the customers who have an account at all the branches located in a specific city.
   5. Demonstrate how you delete all account tuples at every branch located in a specific city.

Creation of Tables

BRANCH TABLE

```
CREATE TABLE branch ( branch_name VARCHAR(15),  branch_city
VARCHAR(15), assets NUMBER(10,2), PRIMARY KEY(branch_name) );
```

ACCOUNT TABLE

```
CREATE TABLE account ( accno INTEGER(8), branch_name VARCHAR(15),
balance NUMBER(10,2), PRIMARY KEY(accno), FOREIGN KEY(branch_name)
REFERENCES branch(branch_name)ON DELETE CASCADE );
```

CUSTOMER TABLE

```
CREATE TABLE customer ( customer_name VARCHAR(15),  customer_street
VARCHAR(15), customer_city VARCHAR(15), PRIMARY KEY(customer_name) );
```

LOAN TABLE

```
CREATE TABLE loan ( loan_number INTEGER(8), branc_hname VARCHAR(15),
amount NUMBER(10,2), PRIMARY KEY(loan_number), FOREIGN
KEY(branch_name)  REFERENCES branch(branch_name) );
```

DEPOSITOR TABLE

```
CREATE TABLE depositor ( customer_name VARCHAR(15), accno INTEGER,
PRIMARY KEY(customer_name, accno), FOREIGN KEY(customer_name)
REFERENCES customer(customer_name), FOREIGN KEY(accno) REFERENCES
account(accno) );
```

BORROWER TABLE

```
CREATE TABLE borrower ( customer_name  VARCHAR(15), loan_number
INTEGER(8), PRIMARY KEY(customer_name, loan_number), FOREIGN
KEY(customer_name) REFERENCES customer(customer_name), FOREIGN
KEY(loan_number) REFERENCES loan(loan_number) );
```

Insertion into tables

```
INSERT INTO branch (branch_name,branch_city,assets) values
   ("b1","c1",10000),
   ("b2","c2",20000),
   ("b3","c3",30000),
   ("b4","c4",40000),
   ("b5","c5",50000);
```

SQL> SELECT * FROM branch;

branch_name	branch_city	assets
b1	c1	10000
b2	c2	20000
b3	c3	30000
b4	c4	40000
b5	c5	50000

```
INSERT INTO account (accno,branch_name,balance) VALUES
   (12,"b1",3000),
   (22,"b2",4000),
   (32,"b3",5000),
   (42,"b4",6000),
   (52,"b5",7000);
```

SQL> SELECT * FROM account;

accno	branch_name	balance
12	b1	3000
22	b2	4000
32	b3	5000
42	b4	6000
52	b5	7000

```
INSERT INTO customer (customer_name,customer_street,customer_city) VALUES
   ("cust1","cstreet1","ccity1"),
   ("cust2","cstreet2","ccity2"),
   ("cust3","cstreet3","ccity3"),
   ("cust4","cstreet4","ccity4"),
   ("cust5","cstreet5","ccity5");
```

SQL> SELECT * FROM customer;

customer_name	customer_street	customer_city
cust1	cstreet1	ccity1
cust2	cstreet2	ccity2
cust3	cstreet3	ccity3
cust4	cstreet4	ccity4
cust5	cstreet5	ccity5

INSERT INTO depositor (customer_name,accno) VALUES
("cust1",12),
("cust2",22),
("cust3",32),
("cust4",42),
("cust5",52);

SQL> SELECT * FROM depositor;

customer_name	accno
cust1	12
cust2	22
cust3	32
cust4	42
cust5	52

INSERT INTO loan (loan_number,branch_name,amount) VALUES
(10,"b1",10000),
(20,"b2",20000),
(30,"b3",30000),
(40,"b4",40000),
(50,"b5",50000);

SQL> SELECT * FROM loan;

loan_number	branch_name	amount
10	b1	10000
20	b2	20000
30	b3	30000
40	b4	40000
50	b5	50000

INSERT INTO borrower (customer_name,loan_number) VALUES

("cust1",10),
("cust2",20),
("cust3",30),
("cust4",40),
("cust5",50);

SQL> SELECT * FROM borrower;

customer_name	loan_number
cust1	10
cust2	20
cust3	30
cust4	40
cust5	50

1. **Find all the customers who have at least two accounts at the Main branch.**

   SELECT customer_name FROM depositor d,account a WHERE d.accno=a.accno AND a.branch_name='Main' GROUP BY d.customer_name HAVING COUNT(d.customer_name)>=2;

2. **Find all the customers who have an account at all the branches located in a specific city.**

   SELECT d.customer_name FROM account a,branch b,depositor d WHERE b.branch_name=a.branch_name AND a.accno=d.accno AND b.branch_city='c3' GROUP BY d.customer_name HAVING COUNT(distinct b.branch_name)=(SELECT COUNT(branch_name) FROM branch WHERE branch_city='c3');

3. **Demonstrate how you delete all account tuples at every branch located in a specific city.**

   DELETE FROM account WHERE branch_name IN(SELECT branch_name FROM branch WHERE branch_city='c5');

3. Consider the schema for College Database:

STUDENT(USN, SName, Address, Phone,

Gender)SEMSEC(SSID, Sem, Sec)

CLASS(USN, SSID)

SUBJECT(Subcode, Title, Sem, Credits)

IAMARKS(USN, Subcode, SSID, Test1, Test2, Test3, FinalIA)

Write SQL queries to

a. **List all the student details studying in fourth semester 'C' section.**

b. **Compute the total number of male and female students in each semester and in each section.Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.**

c. **Calculate the FinalIA (average of best two test marks) and update the corresponding tablefor all students.**

d. **Categorize students based on the following criterion:**

If FinalIA = 17 to 20 then CAT = 'Outstanding'If FinalIA = 12 to 16 then CAT = 'Average'

If FinalIA< 12 then CAT = 'Weak' Give these details only for 8th semester A, B, and C sectionstudents.

Creation of Tables

STUDENT TABLE

```
CREATE TABLE STUDENT (USN INTEGER PRIMARY
KEY,SNAME VARCHAR(20), ADDRESS VARCHAR(20),
PHONE INTEGER (10), GENDER VARCHAR(7) );
```

SEMSEC TABLE

```
CREATE TABLE SEMSEC (SSID INTEGER(5) PRIMARY
KEY,SEM INTEGER (2), SEC VARCHAR(2));
```

CLASS TABLE

```
CREATE TABLE CLASS (USN VARCHAR (10), SSID INTEGER (5),
```

PRIMARYKEY(USN,SSID), FOREIGN KEY (USN) REFERENCES STUDENT(USN), FOREIGN KEY (SSID) REFERENCES SEMSEC(SSID));

SUBJECT TABLE

CREATE TABLE SUBJECT(SUBCODE VARCHAR(20) PRIMARY KEY,TITLE VARCHAR(20), SEM INTEGER (2), CREDITS INTEGER(2));

IAMARKS TABLE

CREATE TABLE IAMARKS(USN VARCHAR(20) PRIMARY KEY, SUBCODE VARCHAR(20), SSID INTEGER (5), TEST1 INTEGER (2),TEST2 INTEGER (2), TEST3 INTEGER (2), FINALIA INTEGER (2), PRIMARY KEY(USN,SUBCODE,SSID), FOREIGN KEY(SUBCODE) REFERENCES SUBJECT(SUBCODE), FOREIGN KEY(SSID) REFERENCES SEMSEC(SSID));

Insertion into tables

insert into student
values('&usn','&sname','&address',&phone,'&gender');Enter value for usn: cs057
Enter value for sname: raju
Enter value for address:
bangalore Enter value for
phone: 9278566666Enter
value for gender: female
old 1: insert into student
values('&usn','&sname','&address',&phone,'&gender')new 1: insert into student values('cs057','raju','bangalore',9278566666,'female')

select * from student;

USN	SNAME	PHONE	ADDRESS	GENDER
57	raju	927	bangalore	f

58	raja	934	bangalore	m
59	mahesh	962	bang	m
60	mala	988	bang	f
61	swetha	988	bang	f
62	lakshmi	980	bang	f
63	jose	980	bang	m
64	jhon	980	bang	m
65	daniel	980	bang	m
67	sam	980	bang	m
68	hashni	999	bang	f
69	prisha	980	bang	f
70	nandini	980	bang	F

insert into semsec
values(&ssid,&sem,'&sec');Enter value
for ssid: 512
Enter value for
sem: 5Enter
value for sec: c
old 1: insert into semsec
values(&ssid,&sem,'&sec')new 1: insert into
semsec values(512,5,'c')

select * from semsec;

SSID	SEM	SEC
512	5	c
513	4	c
514	5	a
515	3	b
516	3	d
517	2	c
518	2	a
519	1	b

520	1	b
521	7	c
522	7	d
523	8	c
524	8	a
525	4	c

insert into class
values('&usn',&ssid);Enter
value for usn: cs057
Enter value for ssid: 512
old 1: insert into class
values('&usn',&ssid)new 1: insert
into class values('cs057',512)

USN	SSID
57	512
58	523
59	521
60	513
61	517
67	515
68	516
69	520
70	520

insert into subject
values('&subcode','&title',&sem,&credits);Enter value
for subcode: 111
Enter value for title:
dbmsEnter value for
sem: 5 Enter value
for credits: 10

old 1: insert into subject

values('&subcode','&title',&sem,&credits)new 1: insert into

subject values('111','dbms',5,10)

select * from subject;

SUBCODE	TITLE	SEM	CREDITS
111	dbms	5	10
112	cprog	1	8
113	java	8	9
114	se	7	9
115	cd	5	8
116	ec	4	7
117	oops	4	8
118	st	7	8
119	os	5	9
120	phy	2	8

insert into iamarks

values('&usn','&subcode',&ssid,&test1,&test2,&yest3,&finalia);Enter value for

usn: cs057

Enter value for

subcode: 111Enter

value for ssid: 512

Enter value for test1:

25 Enter value for

test2: 25 Enter value

for yest3: 25 Enter

value for finalia: 25

old 1: insert into iamarks

values('&usn','&subcode',&ssid,&test1,&test2,&yest3,&finalia)new 1: insert into

iamarks values('cs057','111',512,25,25,25,25)

select * from iamarks;

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINALIA
057	111	512	17	17	17	17
058	113	523	19	19	20	20
059	114	521	19	19	20	20
060	116	513	12	12	12	12
061	119	517	15	15	15	15
063	112	519	8	8	8	8
064	117	522	20	20	20	20
065	118	514	19	20	20	20
068	119	516	20	20	21	20
069	112	520	19	19	19	19
070	113	524	10	10	10	10

Write SQL queries to

1. List all the student details studying in fourth semester 'C' section.

```
SELECT S.USN,S.SNAME,ADDRESS,PHONE,GENDER
FROM STUDENT S,SEMSEC M,CLASS C WHERE
S.USN=C.USNAND  M.SSID=C.SSID AND  SEM='4' AND
SEC='C';
```

2. Compute the total number of male and female students in each semester and in each section.

```
SELECT SS.SEM, SS.SEC, S.GENDER, COUNT (S.GENDER) AS COUNT
FROM STUUDENT S, SEMSEC SS, CLASS C WHERE S.USN = C.USN AND
SS.SSID = C.SSID  GROUP BY SS.SEM, SS.SEC, S.GENDER ORDER BY SEM;
```

3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

```
CREATE VIEW TEST1  AS  SELECT
I.TEST1,B.TITLE,I.SSID,B.SUBCODEFROM IAMARKS I,STUUDENT
S,SUBJECT  B
WHERE I.USN=S.USN AND I.SUBCODE=B.SUBCODE AND S.USN='057';
```

4.Calculate the FinalIA (average of best two test marks) and update the corresponding table forall students.

```
UPDATE IAMARKS SET FINALIA=( CASE WHEN((TEST1=TEST2) AND
(TEST2=TEST3))THEN (TEST1+TEST2)/2
   WHEN((TEST1=TEST2) AND ((TEST2<TEST3) OR (TEST1<TEST3)))
THEN(TEST1+TEST3)/2
   WHEN((TEST1=TEST3) AND ((TEST3<TEST2) OR (TEST1<TEST2)))
THEN(TEST3+TEST2)/2
   WHEN((TEST2=TEST3) AND ((TEST2<TEST1) OR (TEST3<TEST1)))
THEN(TEST2+TEST1)/2
   WHEN((TEST1=TEST2) AND ((TEST2>TEST3) OR (TEST1>TEST3)))
THEN(TEST1+TEST2)/2
   WHEN((TEST1=TEST3) AND ((TEST3>TEST2) OR (TEST1>TEST2)))
THEN(TEST1+TEST3)/2
   WHEN((TEST2=TEST3) AND ((TEST2>TEST1) OR (TEST3>TEST1)))
THEN(TEST2+TEST3)/2
   WHEN ((TEST1>TEST2 AND TEST2>TEST3)) THEN
   (TEST1+TEST2)/2 WHEN ((TEST2>TEST3 AND TEST3>TEST1))
   THEN (TEST2+TEST3)/2 WHEN ((TEST3>TEST1 AND
   TEST1>TEST2)) THEN (TEST1+TEST3)/2 WHEN ((TEST1 < TEST2
   AND TEST2< TEST3)) THEN (TEST2+TEST3)/2WHEN ((TEST1 >
   TEST2 AND TEST2< TEST3)) THEN (TEST1+TEST3)/2WHEN
   ((TEST2 < TEST3 AND TEST3< TEST1)) THEN (TEST1+TEST3)/2
   WHEN ((TEST2 > TEST3 AND TEST3< TEST1)) THEN
   (TEST1+TEST2)/2WHEN ((TEST3 < TEST1 AND TEST1< TEST2))
   THEN (TEST1+TEST2)/2WHEN ((TEST3 > TEST1 AND TEST3<
   TEST2)) THEN (TEST3+TEST2)/2END);
```

5. Categorize students based on the following criterion:

If FinalIA = 17 to 20 then CAT = 'Outstanding'If FinalIA = 12 to 16 then CAT = 'Average'

If FinalIA< 12 then CAT = 'Weak' Give these details only for 8th semester A, B, and C section

students.

```
SELECT S.USN,S.SNAME,I.FINALIA, CASE  WHEN I.FINALIA BETWEEN 17
AND 20THEN 'OUTSTANDING’ WHEN I.FINALIA BETWEEN 12 AND 16 THEN
'AVERAGE' ELSE 'WEAK' END CASE FROM STUUDENT S,SEMSEC
SS,IAMARKS I,SUBJECT BWHERE S.USN=I.USN AND SS.SSID=I.SSID AND
B.SUBCODE=I.SUBCODE AND SS.SEM=8;
```

5. Consider the schema for Company Database:

EMPLOYEE(SSN, Name, Address, Sex, Salary, SuperSSN, DNo)

DEPARTMENT(DNo, DName, MgrSSN, MgrStartDate)

DLOCATION(DNo,DLoc) PROJECT(PNo, PName, PLocation, DNo)

WORKS_ON(SSN, PNo, Hours)

Write SQL queries to

1. **Make a list of all project numbers for projects that involve an employee whose last nameis 'Scott', either as a worker or as a manager of the department that controls the project.**
2. **Show the resulting salaries if every employee working on the 'IoT' project is given a 10percent raise.**
3. **Find the sum of the salaries of all employees of the 'Accounts' department, as well as themaximum salary, the minimum salary, and the average salary in this department**
4. **Retrieve the name of each employee who works on all the projects controlledby department number 5 (use NOT EXISTS operator).**
5. **For each department that has more than five employees, retrieve the department numberand the number of its employees who are making more than Rs. 6,00,000.**

Creation of Tables

DEPARTMENT TABLE

```
CREATE TABLE DEPARTMENT (DNO INTEGER PRIMARY
KEY,DNAME VARCHAR(20), MGRSSN INTEGER(5) ,
MGRSTARTDATE DATE
FOREIGN KEY (MGESSN) REFERENCES EMPLOYEE(SSN));
```

EMPLOYEE TABLE

```
CREATE TABLE EMPLOYEE (SSN INTEGER(4) PRIMARY KEY,
NAME VARCHAR(20), ADDRESS VARCHAR(20), SEX
VARCHAR(20),SALARY INTEGER (8), SUPERSSN INTEGER (5),
DNO INTEGER (5), FOREIGN KEY (DNO) REFERENCES
DEPARTMENT(DNO));
```

DLOCATION TABLE

```
CREATE TABLE DLOCATION (DNO INTEGER (4), DLOC VARCHAR(20) ,
PRIMARY KEY(DNO,DLOC),
FOREIGN KEY(DNO) REFERENCES DEPARTMENT(DNO));
```

PROJECT TABLE

```
CREATE TABLE PROJECT (PNO INTEGER PRIMARY KEY,
PNAME VARCHAR(20), PLOCATION VARCHAR(20), DNO
INTEGER,FOREIGN KEY(DNO) REFERENCES
DEPARTMENT(DNO));
```

WORKS_ON TABLE

```
CREATE TABLE WORKS_ON (SSN INTEGER (5), PNO INTEGER
(4),HOURS DECIMAL(3,2), PRIMARY KEY(SSN,PNO),
FOREIGN KEY(SSN) REFERENCES
EMPLOYEE(SSN),FOREIGN KEY(PNO)
REFERENCES PROJECT(PNO));
```

INSERTION INTO THE TABLE

```
insert into EMPLOYEE
values('&Ssn','&Ename','&Address','&Sex',&Salary,'&Super_ssn',&Dno);Enter value for ssn:
123456789
Enter value for ename: John
Enter value for address: 731 Fondren
HoustonEnter value for sex: M
Enter value for salary:
30000 Enter value for
super_ssn: 112Enter
value for dno: 5
old 1: insert into EMPLOYEE
values('&Ssn','&Ename','&Address','&Sex',&Salary,'&Super_ssn',&Dno)
new  1: insert into EMPLOYEE values('123456789','John','731 Fondren Houston','M',30000,'112',5)
```

SQL> select * from EMPLOYEE;

SSN	ENAME	ADDRESS	SEX	SALARY	SUPER_SSN	DNO
111	John	731 Fondren Houston	M	30000	112	5
112	Scott	638 voss Houston	M	40000	118	5
113	Alicia	3321 Castle Spring	F	25000	114	4
114	Jennifer	291 Berry Bellaire	F	25000	114	4
115	Ramesh	975 Fire oak Humble	M	38000	112	5
116	Joyce	5631 Rile Houston	F	25000	112	5
117	Ahmad	9801 Dallas Houston	M	25000	117	4
118	Dinesh	450 Stone Houston	M	55000	NULL	1

insert into DEPARTMENT values('Research',5,112,DATE '1988-05-

22');select * from DEPARTMENT;

DNAME	DNUMBER	MGR_SSN	MGR_START
Accounts	5	112	22-MAY-88
Administration	4	117	01-JAN-95
Headquarters	1	118	19-JUN-81

insert into DEPT_LOCATIONS values(1,'Houston');select *

from DEPT_LOCATIONS;

DNUMBER	DLOCATION
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

insert into PROJECT

values('&Pname',&Pnumber,'&Plocation',&Dnum);Enter value for

pname: ProductX

Enter value for pnumber: 1

Enter value for plocation:

BellaireEnter value for

dnum: 5

old 1: insert into PROJECT

values('&Pname',&Pnumber,'&Plocation',&Dnum)new 1: insert into

PROJECT values('ProductX',1,'Bellaire',5)

select * from PROJECT;

PNAME	PNUMBER	PLOCATION	DNUM
ProductX	1	Bellaire	5
IOT	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

insert into WORKS_ON

values('&Essn',&Pno,&Hours);Enter value for

essn: 111

Enter value for pno:

1 Enter value for

hours: 32.5

old 1: insert into WORKS_ON

values('&Essn',&Pno,&Hours)new 1: insert into

WORKS_ON values('111',1,32.5)

select * from WORKS_ON;

ESSN	PNO	HOURS
111	1	32.5
111	2	7.5
115	3	40
116	1	20
116	2	20
112	2	10
112	3	10
113	30	30
117	10	35
117	20	15

Write SQL queries to

1. **Make a list of all project numbers for projects that involve an employee whose last name is‘Scott’,**
   either as a worker or as a manager of the department that controls the project.

```
(SELECT PNUMBER FROM PROJECT P, DEPARTMENT D, EMPLOYEE E
WHEREP.DNUM=D.DNUMBER AND D.MGR_SSN=E.SSN AND
E.ENAME='SCOTT') UNION  (SELECT PNUMBER FROM PROJECT P,
WORKS_ON W, EMPLOYEE E
WHERE P.PNUMBER=W.PNO AND E.SUPER_SSN=E.SSN AND E.ENAME='SCOTT');
```

2. **Show the resulting salaries if every employee working on the ‘IoT’ project is given a 10percent raise.**

```
SELECT ENAME, 1.1*SALARY AS INCREASED_SAL FROM EMPLOYEE E,
WORKS_ON W, PROJECT P WHERE P.PNAME = 'IOT' AND P.PNUMBER =
W.PNO AND W.ESSN = E.SSN;
```

3. **Find the sum of the salaries of all employees of the ‘Accounts’ department, as well as**

themaximum salary, the minimum salary, and the average salary in this department

```
SELECT SUM(SALARY), MAX(SALARY), MIN(SALARY), AVG(SALARY)
FROMEMPLOYEE E JOIN DEPARTMENT D ON E.DNO = D.DNUMBER
WHERE D.DNAME = 'ACCOUNTS';
```

4. **Retrieve the name of each employee who works on all the projects controlled by departmentnumber 5 (use NOT EXISTS operator).**

```
SELECT ENAME FROM EMPLOYEE E WHERE NOT EXISTS ( ( SELECT
PNUMBER FROM PROJECT P WHERE P.DNUM=5) MINUS ( SELECT PNO
FROM WORKS_ON WWHERE E.SSN=E.SUPER_SSN) );
```

5. **For each department that has more than five employees, retrieve the department number andthe number of its employees who are making more than Rs. 6,00,000.**

```
SELECT DNUMBER, COUNT(*) FROM DEPARTMENT D, EMPLOYEE E WHERE
D.DNUMBER=E.DNO AND E.SALARY>6,00000 AND D.DNUMBER IN ( SELECT
E.DNO FROM EMPLOYEE GROUP BY E.DNO HAVING COUNT (*) > 5) GROUP
BY D.DNUMBER;
```

VIVA VOCE

1. What is database?

A database is a collection of data in an organized manner.

2. What is DBMS?

DBMS is a software that is used to perform operations on a database. These operations may include reading, writing, modifying of data and even provide control over accessing of data when multiple users were accessing the data at the same time or even at different times.

3. Disadvantage in File Processing System?

- ✓ Data redundancy & inconsistency.
- ✓ Difficult in accessing data.
- ✓ Data isolation.
- ✓ Data integrity.
- ✓ Concurrent access is not possible.
- ✓ Security Problems

4. Advantages of DBMS?

- ✓ Redundancy is controlled.
- ✓ Unauthorized access is restricted.
- ✓ Providing multiple user interfaces.
- ✓ Enforcing integrity constraints.
- ✓ Providing backup and recovery.

5. Define the "integrity rules"

There are two Integrity rules.

- Entity Integrity: States that? Primary key cannot have NULL value?

Referential Integrity: States that? Foreign Key can be either a NULL value or should be Primary Keyvalue of other relation

6. Describe the three levels of data abstraction?

Three levels of abstraction:

- Physical level: The lowest level of abstraction describes how data are stored.
- Logical level: The next higher level of abstraction, describes what data are stored in databaseand what relationship among those data.
- View level: The highest level of abstraction describes only part of entire database.

7. What is Data Independence?

Data independence means, the ability to modify the schema definition in one level should notaffect the schema definition in the next higher level.

Two types of Data Independence:

- Physical Data Independence: Modification in physical level should not affect the logical level.
- Logical Data Independence: Modification in logical level should affect the view level.

8. What is a view? How it is related to data independence?

A view may be thought of as a virtual table, that is, a table that does not really exist in its own right but is instead derived from one or more underlying base tables. Growth and restructuring of base tables is not reflected in views. Thus the view can insulate users from the effects of restructuring and growth in the database. Hence accounts for logical data independence.

9. What is Data Model?

A collection of conceptual tools for describing data, data relationships, data semantics and constraints is called Data Model.

10. What is E-R model?

This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

11. What is an Entity?

It is a 'thing' in the real world with an independent existence.

12. What is an Entity type?

It is a collection (set) of entities that have same attributes.

13. What is an Entity set?

It is a collection of all entities of particular entity type in the database.

14. What is Weak Entity set?

An entity set may not have sufficient attributes to form a primary key, and its primary key compromises of its partial key and primary key of its parent entity, then it is said to be Weak Entity set.

15. What is an attribute?

It is a particular property, which describes the entity

16. What is Object Oriented model?

This model is based on collection of objects. An object contains values stored in instance variables within the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods aregrouped together into classes.

.

17. What is a Relation Schema and a Relation?

A relation Schema denoted by R (A1, A2,…?, An) is made up of the relation name R and the list of attributes Ai that it contains. A relation is defined as a set of tuples. Let r be the relation which contains set tuples (t1, t2, t3... tn). Each tuple is an ordered list of n-values t= (v1, v2... vn).

18. What is Relational Algebra?

It is procedural query language. It consists of a set of operations that take one or two relationsas input and produce a new relation.

19. What is degree of a Relation?

It is the number of attributes of its relation schema.

20. What is Relationship?

It is an association among two or more entities.

21. What is Relationship set?

The collection (or set) of similar relationships.

22. What is Relationship type?

Relationship type defines a set of associations or a relationship set among a given set of entity types.

23. What is DML Compiler?

It translates DML statements in a query language into low-level instruction that the query evaluation engine can understand.

24. What is Query evaluation engine?

It executes low-level instruction generated by compiler.

25. What is DDL Interpreter?

It interprets DDL statements and record them in tables containing metadata.

26. What is Record-at-a-time?

The Low level or Procedural DML can specify and retrieve each record from a set of records. This retrieve of a record is said to be Record-at-a-time.

27. What is normalization?

It is a process of analysing the given relation schemas based on their Functional Dependencies(FDs) and primary key to achieve the properties

- Minimizing redundancy
- Minimizing insertion, deletion and update anomalies.

28. What is 1 NF (Normal Form)?

The domain of attribute must include only atomic (simple, indivisible) values.

29. What is 2NF?

A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on primary key.

30. What is 3NF?

A relation schema R is in 3NF if it is in 2NF and for every FD X A either of the following is true

- X is a Super-key of R.
- A is a prime attribute of R.

In other words, if every non prime attribute is non-transitively dependent on primary key.

31. What is BCNF (Boyce-Codd Normal Form)?

A relation schema R is in BCNF if it is in 3NF and satisfies an additional constraint that forevery FD X A, X must be a candidate key.

32. What is 4NF?

A relation schema R is said to be in 4NF if for every multivalued dependency X Y that holdsover R, one of following is true

? X is subset or equal to (or) XY = R.

? X is a super key.

33. What is 5NF?

A Relation schema R is said to be 5NF if for every join dependency {R1, R2... Rn} that holdsR, one the following is true

? Ri = R for some i.

? The join dependency is implied by the set of FD, over R in which the left side is key of R.

34. What is Domain-Key Normal Form?

A relation is said to be in DKNF if all constraints and dependencies that should hold on the constraint can be enforced by simply enforcing the domain constraint and key constraint on the relation

35. What is Functional Dependency?

A Functional dependency is denoted by X Y between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuple that can form a relation state r of R. The constraint is for any two tuples t1 and t2 in r if t1[X] = t2[X] then they have t1[Y] = t2[Y]. This means the value of X component of a tuple uniquely determines the value of component Y.

36. When is a functional dependency F said to be minimal?

- Every dependency in F has a single attribute for its right hand side.
- We cannot replace any dependency X A in F with a dependency Y A where Y is a propersubset of X and still have a set of dependency that is equivalent to F.
- We cannot remove any dependency from F and still have set of dependency that is equivalentto F.

37. What is Lossless join property?

It guarantees that the spurious tuples generation does not occur with respect to relation schemas after decomposition.

38. What is Fully Functional dependency?

It is based on concept of full functional dependency. A functional dependency X Y is full functional dependency if removal of any attribute A from X means that the dependency does not hold any more.

39. What is indexing and what are the different kinds of indexing?

Indexing is a technique for determining how quickly specific data can be found.

Types:

- Binary search style indexing
- B-Tree indexing
- Inverted list indexing
- Memory resident table
- Table indexing

40. What is meant by query optimization?

The phase that identifies an efficient execution plan for evaluating a query that has the least estimated cost is referred to as query optimization.

41. What do you mean by atomicity and aggregation?

Atomicity: Either all actions are carried out or none are. Users should not have to worry about the effect of incomplete transactions. DBMS ensures this by undoing the actions of incompletetransactions.

Aggregation: A concept which is used to model a relationship between a collection of entities and relationships. It is used when we need to express a relationship among relationships.

42. What is a checkpoint and when does it occur?

A Checkpoint is like a snapshot of the DBMS state. By taking checkpoints, the DBMS can reduce the amount of work to be done during restart in the event of subsequent crashes.

43. What is "transparent DBMS"?

It is one, which keeps its Physical Structure hidden from user.

44. What is a query?

A query with respect to DBMS relates to user commands that are used to interact with a data base.

45. What do you mean by Correlated sub query?

Sub queries, or nested queries, are used to bring back a set of rows to be used by the parent query. Depending on how the sub query is written, it can be executed once for the parent query or it can be executed once for each row returned by the parent query. If the sub query is executed for each row of the parent, this is called a correlated sub query.

E.g. Select * From CUST Where '10/03/1990' IN (Select ODATE from ORDER Where CUST.CNUM = ORDER.CNUM)

46. What are the primitive operations common to all record management systems?

Addition, deletion and modification.

47. Define SQL and state the differences between SQL and other conventional programming Languages

SQL is a nonprocedural language that is designed specifically for data access operations on normalized relational database structures. The primary difference between SQL and other conventional programming languages is that SQL statements specify what data operations should be performed rather than how to perform them.

48. What is an Oracle Instance?

The Oracle system processes, also known as Oracle background processes, provide functionsfor the user processes.

Oracle database-wide system memory is known as the SGA, the SYSTEM GLOBAL AREA or SHARED GLOBAL AREA.

The combination of the SGA and the Oracle background processes is known as an Oracle instance

49. What are the four Oracle system processes that must always be up and running for the database to be useable

DBWR (Database Writer), LGWR (Log Writer), SMON (System Monitor), and PMON (Process Monitor).

50. What is database Trigger?

A database trigger is a PL/SQL block that can be defined to perform certain actions automatically when a condition is encountered. For example: a trigger can be defined to deny any database insertions into a database on a specified system date.

APPENDIX A : SQL COMMANDS

- Open SQL*Plus Command Line. Type the username and password and login.
- Type your DDL AND DML queries/statements in a notepad before pasting it in the SQLcommand line. This will make debugging easier.
- To make editing easier in SQL command line,
    - Right-click on title bar and go to properties.
    - Go to the Options tab
    - Check "Quick Edit Mode"
    - Drag and drop mouse to select. Press "Enter" to copy
    - Right-click to "Paste"
- For large queries, type queries in notepad and save with .sql extension.
- Run the sql file in the SQL* PLUS command line.
    - SQL> @C:\ <path>\<filename>.sql

SQL COMMADS

SQLCREATETABLEStatement:

```
CREATE TABLE table_name(

column1   datatype,
column2   datatype,
column3 datatype,

.....

columnN datatype,

PRIMARY KEY( one or more columns )
```

SQLDESCStatement:

```
DESC table_name;
```

SQLSELECTStatement:

```
SELECT column1, column2.....columnN

FROM  table_name;
```

SQLDISTINCTClause:

```
SELECT DISTINCT column1, column2 ..... columnN
FROM  table_name;
```

SQLWHEREClause:

```
SELECT column1, column2..... columnN
FROM table_name
WHERE  CONDITION;
```

SQLAND/ORClause:

```
SELECT column1, column2..... columnN
FROM  table_name
```

SQLINClause:

```
SELECT column1, column2..... columnN
FROM  table_name
```

SQLBETWEENClause:

```
SELECT column1, column2..... columnN
FROM  table_name
```

SQLLIKEClause:

```
SELECT column1, column2..... columnN
FROM  table_name
```

SQLORDERBYClause:

```
SELECT column1, column2..... columnN
FROM table_name
WHERE  CONDITION
```

SQLGROUPBYClause:

```
SELECT SUM(column_name)
FROM  table_name
WHERE  CONDITION
```

SQLCOUNTClause:

```
SELECT COUNT(column_name)FROM
table_name
WHERE  CONDITION;
```

SQLHAVINGClause:

```
SELECT SUM(column_name)
FROM table_name
WHERE  CONDITION
GROUP BY column_name
```

SQLTRUNCATETABLEStatement:

```
TRUNCATE TABLE table_name;
```

SQLALTERTABLEStatement:

```
ALTER TABLE table_name {ADD|DROP|MODIFY} column_name {data_ype};
```

SQLALTERTABLEStatement(Rename):

```
ALTER TABLE table_name RENAME TO new_table_name;
```

SQLINSERTINTOStatement:

```
INSERT INTO table_name( column1, column2 ..... columnN)
VALUES ( value1, value2 ..... valueN);
```

SQLUPDATEStatement:

```
UPDATE table_name
SET column1 = value1, column2 = value2 ..... columnN=valueN
```

SQLDELETEStatement:

```
DELETE FROM table_name
WHERE  {CONDITION};
```

SQLCOMMITStatement:

```
COMMIT;
```

SQLROLLBACKStatement:

```
ROLLBACK;
```

APPENDIX B : FRONT END USING VB

Setting up the ODBC Database Connectivity

1. Go to Control Panel → Administrative Tools
2. Click on Data Sources(ODBC).
3. Switch to the "System DSN" Tab
4. Click "Add"
5. Choose "Oracle in OraClient 11g_home1" driver and click "Finish".
6. Give a data source name : dbOrc.
    a. Select TNS Name → CSE (It should appear in the drop-down menu).
    b. Give user id (everything before @ in your usual login is the user it
    c. Click on "Test Connection" . Issue Password, A Connection Successful windowshould pop up.
    d. Click "OK", then "OK"again.

Visual Studio 10 (The steps may vary slightly depending on the version)

1. Go to File New Project.
2. Expand Other Languages Visual Basic Windows Windows Forms Application
3. Give a name, Click Okay.
4. Click on Data Add New Data Source
5. Select Database, Next
6. Select Datase, Next
7. Select "New Connection"
8. Select "Microsoft ODBC Data Soiurce
9. In the Data source, select the object dreated, 'dbOrc'.
    a. Issue user name and password.
    b. Click on "Test Connection". You should get a connection successful window.
10. Check "Keep Sensitive Data" Press Continue.
11. Give a name for the Connection or leave it as it is.
12. Expand the tables, select all tables related to the database for which you are generating thefront end.
13. Create a form, add the appropriate labels, text boxes, set the properties of forms.
14. Go to "data sources" tab. Drag and drop the table to the form.
15. Build Solution and execute.

CONTENT BEYOND

SYLLABUS BANK

DATABASE

Consider the following database for a banking enterprise

BRANCH(branch-name:string, branch-city:string, assets:real)ACCOUNT(accno:int, branch-name:string, balance:real) DEPOSITOR(customer-name:string, accno:int)
CUSTOMER(customer-name:string, customer-street:string, customer-city:string)LOAN(loan-number:int, branch-name:string, amount:real)
BORROWER(customer-name:string, loan-number:int)

i. Create the above tables by properly specifying the primary keys and the foreign keys

```
CREATE TABLE branch(branch_name VARCHAR(15), branch_city VARCHAR(15),
\assets real, CONSTRAINT branchpk PRIMARY KEY (branch_name));

CREATE TABLE customer(customer_name
VARCHAR(15), customer_street VARCHAR(15),
customer_city VARCHAR(15),CONSTRAINT custpk
PRIMARY KEY (customer_name));

CREATE TABLE account(accno INT, branch_name VARCHAR(15), balance real,
CONSTRAINT accpk PRIMARY KEY (accno),
CONSTRAINT accfk FOREIGN KEY (branch_name) REFERENCES
branch(branch_name) ON DELETE SET NULL);

CREATE TABLE depositor (customer_name VARCHAR(15), accno
INT,CONSTRAINT deppk PRIMARY KEY (customer_name,
accno), CONSTRAINT depfk1 FOREIGN KEY (customer_name)
REFERENCEScustomer(customer_name) ON DELETE CASCADE,
 CONSTRAINT depfk2 FOREIGN KEY (accno) REFERENCES
account(accno)ON DELETE CASCADE);
```

```
CREATE TABLE loan(loan_number INT, branch_name VARCHAR(15), amount real,
CONSTRAINT loanpk PRIMARY KEY(loan_number),
CONSTRAINT loanfk FOREIGN KEY (branch_name)
REFERENCES branch(branch_name) ON DELETE SET
NULL);

CREATE TABLE borrower(customer_name VARCHAR(15), loan_number
INT,CONSTRAINT borrowpk PRIMAR
KEY(customer_name,loan_number), CONSTRAINT borrowfk1 FOREIGN
KEY (customer_name)
REFERENCES customer(customer_name) ON DELETE SET
NULL,CONSTRAINT borrowfk2 FOREIGN KEY
(loan_number) REFERENCES loan(loan_number) ON DELETE
CASCADE);
```

ii. Enter at least five tuples for each relation

Inserting values into branch table:

```
INSERT INTO branch VALUES('SBI Main', 'Mumbai', 36000000000);
```

branch_name	*branch_city*	*assets*
SBI Main	Mumbai	36000000000
K R Puram	Bangalore	10000000000
Indira Nagar	Bangalore	10000000000
Marine Drive	Mumbai	20000000000
Anna Nagar	Chennai	25000000000

Inserting values into customer table:

```
INSERT INTO customer VALUES ('Geetha', '#2 CMH Road', 'Bangalore');
```

customer_name	*customer_street*	*customer_city*
Geetha	#2 CMH Road	Bangalore
Raj	#24 M.G. Road	Bangalore
Jithin	#40 AECS Layout	Bangalore
Akash	#8 Marine Drive	Mumbai
Varun	#44 Anna Nagar	Chennai

Inserting values into account table:

INSERT INTO account VALUES (100, 'Indira Nagar', 36000);

Accno	*branch_name*	*balance*
100	Indira Nagar	36000
101	K R Puram	25000
102	Indira Nagar	20000
103	SBI Main	25000
104	SBI Main	27000
105	SBI Main	30000
106	Anna Nagar	40000

Inserting values into DEPOSITOR table:

INSERT INTO depositor VALUES ('Akash', 103);

customer_name	*accno*
Akash	103
Akash	104
Varun	105
Geetha	101
Geetha	102
Jithin	100
Varun	106

Inserting values into LOAN table:

INSERT INTO loan VALUES (1000, 'K R Puram', 1000000);

loan_number	*branch_city*	*amount*
1000	K R Puram	1000000
1001	Indira Nagar	2000000
1002	SBI Main	1500000
1003	SBI Main	1800000
1004	Marine Drive	2000000

Inserting values into BORROWER table:

INSERT INTO borrower VALUES ('Raj', 1000);

customer_name	*loan_number*
Raj	1000
Raj	1001
Jithin	1002
Akash	1004
Varun	1003

iii. Find all the customers who have at least two accounts at the Main branch.

```
SELECT customer_name FROM account NATURAL JOIN
depositorWHERE upper(branch_name) LIKE '%MAIN'
GROUP BY customer_name HAVING COUNT(*) >=2;
```

iv. Find all the customers who have an account at all the branches located in a specific city.

```
SELECT DISTINCT customer_name FROM branch b,
depositor dWHERE NOT EXISTS(
    (SELECT branch_name FROM branch WHERE branch_city =
 b.branch_city)MINUS
    (SELECT branch_name FROM account NATURAL JOIN branch NATURAL JOIN depositor
WHERE
 customer_name = d.customer_name AND branch_city = b.branch_city) );
```

v. Demonstrate how you delete all account tuples at every branch located in a specific city.

```
DELETE from account WHERE branch_name IN
(SELECT branch_name FROM branch WHERE branch_city ='&branch_city');
```

vi. Generate suitable reports.

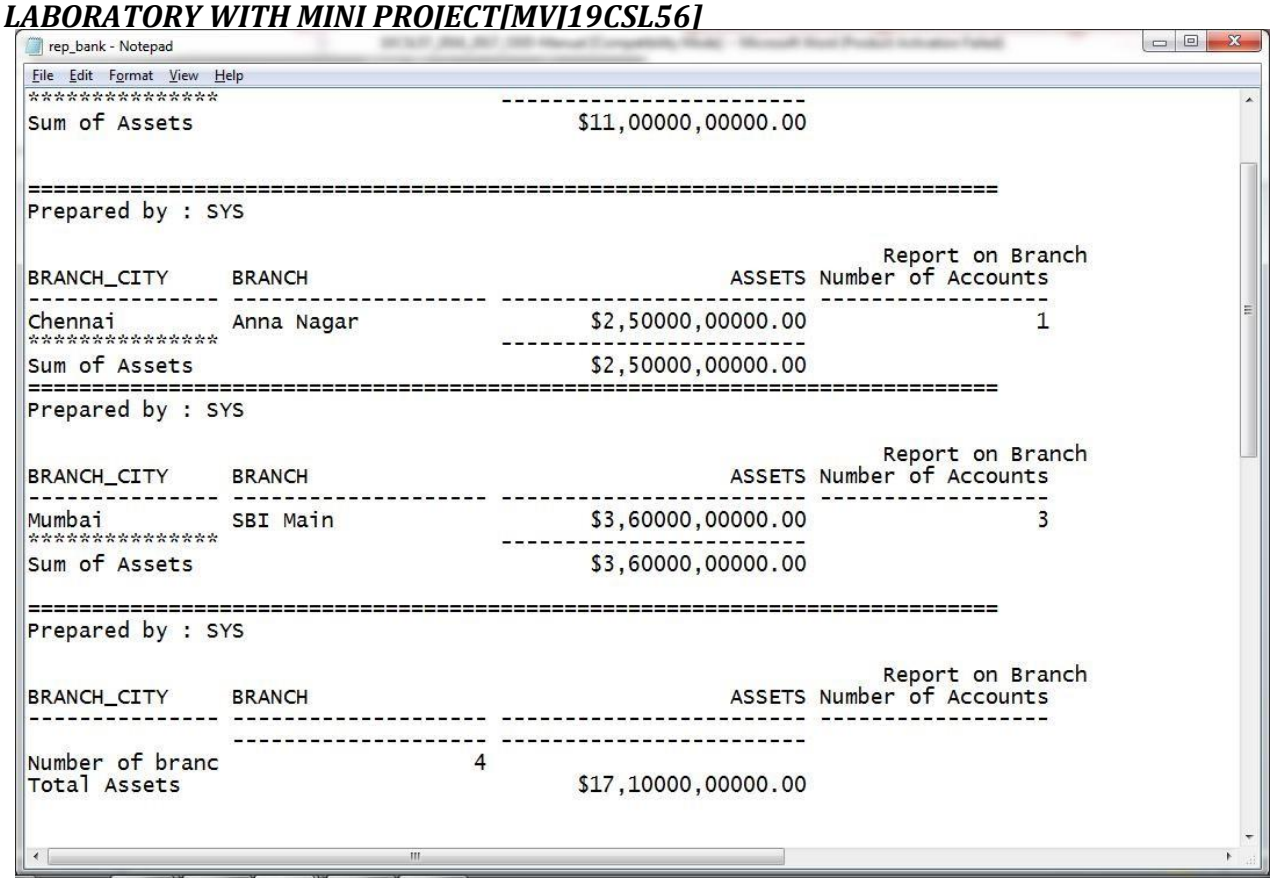
```
set pagesize 50
set linesize
150 set
feedback
OFF
```

```
set termout off

ttitle center 'Report on

Branch'btitle
'================================================================================
====='skip 1-
left 'Prepared by : '
sql.user -right 'Page
No : ' sql.pno

column assets heading 'ASSETS' FORMAT
$999,999,99999,99999.99column branch_name heading
'BRANCH' format A20

BREAK ON REPORT -
ON branch_city SKIP PAGE -

COMPUTE SUM LABEL "Sum of Assets" OF assets ON branch_city
COMPUTE COUNT LABEL "Number of branches" OF branch_name ON
reportCOMPUTE SUM LABEL "Total Assets" OF assets ON REPORT

 SPOOL C:\nf\rep_bank.lis
 SELECT branch_city, branch_name, assets, COUNT(*) AS "Number of
 Accounts"FROM branch NATURAL JOIN account NATURAL JOIN
 depositor
 GROUP BY branch_name, branch_city,
 assetsORDER BY branch_city;

SPOOL OFF
```

Expected Output :

In SQL*Plus

Sql> @C:\nf\repBank.sql



Exercise

Prepare a report on loans

vii. Create suitable front end for querying and displaying the results.

Refer to Appendix B

DO'S AND DON'TS

Do's

1. Do wear ID card and follow dress code.
2. Do log off the computers when you finish.
3. Do ask the staff for assistance if you need help.
4. Do keep your voice low when speaking to others in the LAB.
5. Do ask for assistance in downloading any software.
6. Do make suggestions as to how we can improve the LAB.
7. In case of any hardware related problem, ask LAB in charge for solution.
8. If you are the last one leaving the LAB , make sure that the staff in charge of the LAB isinformed to close the LAB.
9. Be on time to LAB sessions.
10. Do keep the LAB as clean as possible.

Don'ts

1. Do not use mobile phone inside the lab.
2. Don't do anything that can make the LAB dirty (like eating, throwing waste papers etc).
3. Do not carry any external devices without permission.
4. Don't move the chairs of the LAB.
5. Don't interchange any part of one computer with another.
6. Don't leave the computers of the LAB turned on while leaving the LAB.
7. Do not install or download any software or modify or delete any system files on any labcomputers.
8. Do not damage, remove, or disconnect any labels, parts, cables, or equipment.
9. Don't attempt to bypass the computer security system.
10. Do not read or modify other user's file.
11. If you leave the lab, do not leave your personal belongings unattended. We are not responsible for any theft.