S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
01		Accessing Database		
02		BASIC SQL		
03		INTERMEDIATE SQL		
04		INTERMEDIATE AND ADVANCED SQL		
05		ADVANCED SQL		
06		ACCESSING THE DB USING PYTHON		
07		ADVANCE TECHNIQUE THROUGH PYTHON		
08		OODBMS - OBJECT ORIENTED DATABASE MANAGEMENT SYSTEM		

EXPERIMENT-1

ACCESSING THE DATABASE

1. Create the following relation (Tables) with primary key integrity constraint

```
Create Table instructor
(
 ID INT PRIMARY KEY,
 Name TEXT NOT NULL
 Dept_Name TEXT NOT NULL,
 Salary INT
 );
INSERT INTO INSTRUCTOR VALUES
(
 10101, 'Srinivasan', 'Comp. Sci', '65000'
 );
INSERT INTO INSTRUCTOR VALUES
(
 12121, 'Wu', Finance, '90000'
 );
INSERT INTO INSTRUCTOR VALUES
(
 15151, 'Mozart', 'Music', '40000'
 );
INSERT INTO INSTRUCTOR VALUES
(
 22222, 'Einstein', 'Physics', '95000'
 );
```

```
INSERT INTO INSTRUCTOR VALUES
);
  32343, 'El Said', 'History', '60000'
);
INSERT INTO INSTRUCTOR VALUES
(  33456, 'Gold', 'Physics', '87000'
);
INSERT INTO INSTRUCTOR VALUES
(INSERT 'Katz', 'Comp. Sci.', '75000'
);
INSERT INTO INSTRUCTOR VALUES
(  58583, 'Califieri', 'History', 62000'
);
INSERT INTO INSTRUCTOR VALUES
(  76543, 'Singh', Finance', '80000'
):
INSERT INTO INSTRUCTOR VALES
(  76766, 'Crick', 'Biology', '72000'
);
INSERT INTO INSTRUCTOR VALUES
(
  83821, 'Brandt', 'Comp. Sci.', '920000'
);
INSERT INTOI NSTRUCTOR VALUES
(
  98345, 'Kim', Elec. Eng',, '80000'
);
```

ID	NAME	Dept_NAme	Salary
10101	srinivasan	comp. sci.	650000
12121	Wu	Finance	900000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Glod	Physics	870000
45565	katz	comp. sci.	750000
58583	Califori	History	620000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	comp. sci	92000
98345	kim	Elec. Eng.	80000

2)

```
(
IDINT,
Course_ID TEXT NOT NULL,
Sec_ID TEXT NOT NULL,
Semester TEXT NOT NULL,
Year TEXT NOT NULL
);
INSERT INTO TEACHES VALUES
(
 10101, 'CS-101', '1', FALL, '2017'
);
 INSERT INTO TEACHES VALUES
(
 10101, 'CS-315', '1', Spring, '2018'
);
 INSERT INTO TEACHES VALUES
(
 12121, 'FIN-201', '1', spring', '2018'
);
 INSERT INTO TEACHES VALUES
(
 10101, 'CS-347', '1', Fall', '2017'
);
 INSERT INTO TEACHES VALUES
(
 15151, 'MU-315', '1', 'Spring', '2018'
);
```

CS-101	1	Fall
CS-315	1	spring
CS-347	1	Fall
FIN-201	1	spring
MU-315	1	spring
PHY-101	1	Fall
HIS-315	1	spring
CS-101	1	spring
CS 319	1	spring
BIO-301	1	summer
BIO-301	1	summer
CS-190	1	spring
CS-190	2	spring
CS-319	2	spring
EE181	1	spring

```
INSERT INTO TEACHES VALUES
(22222, 'PHY-101', '1', 'Fall', 2017'
);
INSERT INTO TEACHES VALUES
(32343, 'HIS-315', '1', 'Spring', 2018'
);
INSERT INTO TEACHES VALUES
(
45565, 'CS-101', '1', 'Spring', '2018'
);
INSERT INTO TEACHES VALUES
(
45565, 'CS-319', '1', Spring, '2018'
);
INSERT INTO TEACHES VALUES
(76766, 'Bio-101', '1', 'Summer', '2017'
);
INSERT INTO TEACHES VALUES
(
83821, 'CS-190', '1', 'Spring', '2017'
);
INSERT INTO TEACHES VALUES
(
82821, 'CS-319', '2', 'Spring', 2017'
);
INSERT INTO TEACHES VALUES
(
```

10101	Srinivasan	comp. sci.	65000
10211	smith	Biology	66000
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
83821	Crick	Biology	72000
76543	Brandt	Comp sci.	92000
98345	Kim	Elec. Eng.	80000

98945, 'EEE-181', '1', 'Spring', '2017'

);

SELECT * from INSTRUCTOR;

3)

INSERT INTO INSRUCTOR VALUES

(

'10211', 'Smith', 'Biology', '66000'

);

10101	srinivasan	Comp. Sci.
12121	Wu	Finance
15151	Mozart	Music
22222	Einstein	Physics
32343	El Said	History
45565	Gold	Physics
33456	Kato	Comp. Sci.
58583	Califieri	History
76543	Sing	Finance
76766	Crick	Biology
83821	Brandt	Comp. Sci
98345	Kim	Elec. Eng

4)

```
DELETE FROM INSTRUCTOR WHERE
    ID = '10211';

SELECT * FROM INSTRUCTOR.
```

ID	Name	Dept_Name	Salary
32343	El said	History	60000
58583	Califori	History	62000

5)

SELECT * FROM INSTRUCTOR WHERE Dept_Name = 'History'

).

SELECT * FROM INSTRUCTOR CROSS JOIN TEACHES.

Name	Course_ID
Srinivasan	CS-101
Srinivasan	CS-315
Srinivasan	CS-347
Wu	FIN-201
Mozart	MU-315
Einstein	PHY-101
El said	HIS-315
Katz	CS-101
Katz	CS-319
Crick	BIO-101
Crick	BIO-301
Brandt	CS-190
Brandt	CS-190
Brandt	CS-319
Kim	EE-181

7)

SELECT Name, Course_ID FORM INSTRUCTOR INNER JOIN TEACHES ON INSTRUCTOR.ID = TEACHES.ID;

ID	Name	Dept_Name	Salary
12121	Wu	Finance	90000
22222	Einstein	Physics	95000
33821	Brandt	Comp. Sci	920000

8)

SELECT * FROM INSTRUCTOR WHERE Salary BETWEEN 90000 AND 10000;

9)

SELECT * FROM INSTRUCTOR WHERE Salary BETWEEN 90000 AND 100000;

ID	Name	Dept_Name
5151	Mozhart	Music
2348	El said	History
8583	Califieri	History
0101	Srinivasan	Comp. Sci.
16766	Crick	Biology
45565	Katz	Comp. Sci.
76543	Singh	Finance
98345	Kim	Elec. Eng
33456	Gold	Physics
12121	Wu	Finance
83821	Brandt	Comp. Sci
22222	Einstein	Physics

EXPERIMENT - 2

)

SELECT * FROM INSTRUCTOR order by salary

ASC;

ID	Name	Dept_Name
15151	Mozart	Music
32343	El said	History
58583	Califieri	History
10101	Srinivasan	Biology
76766	Crick	Comp. sci.
45565	Katz	Comp. sci
76543	Singh	Finance
98345	Kim	Elec. Eng
33456	Gold	Physics
12121	Wu	Finance
83821	Brandt	Comp. sci.
22222	Einstein	Physics

2)

SELECT COURSE I.D FROM TEACHES WHERE (semester = 'Fall' and year = '2017') OR (semester = 'Spring' and year = "2018")

Course_ID
CS-101
CS-315
CS-347
FIN-201
MDU-315
PHY-101
HIS-315
CS-101
CS-319

)

SELECT COURSE_ID FROM TEACHES WHERE
semester = 'fall' and year = '2017') and(
semester = 'spring' and year = '2018')

Course_ID
CS-101
CS-347
PHY-101

4)

SELECT COURSE ID_ FORM TEACHES WHERE

semester = 'Fall' and year = '2014')

ID	Name	Dept_Name
10101	Srinivasan	Comp. Sci
10211	Smith	Biology
10212	Tom	Biology
12121	Wu	Finance
15151	Mozart	Music
22222	Einstein	Physics
32343	El Said	History
45565	Katz	Comp. Sci.
58583	Califieri	History
76543	Singh	Finance
83821	Crick	Biology
76766	Brandt	Comp. Sci
98345	Kim	Elec. Eng

5)

```
INSERT INTO INSTRUCTOR VALUES
(
'10211', 'Smith', 'Biology', 66000
);
INSERT INTO INSTRUCTOR VALUES
(
'10212', 'Tom', Biology', NULL
);
SELECT * FROM INSTRUCTOR.
```

ID	Name	Dept_Name	salary
10212	Tom	Biology	NULL

AVG(salary)
74153.8462

6)

SELECT * FROM INSTRUCTOR WHERE Salary is NULL

7)

SELECT AVG(Salary) FROM INSTRUCTOR;

Total_Instructors_spring-2018
5

Number-of-Tuples
15

EXPERIMENT-3

1)

```
SELECT COUNT(DISTINCT ID) AS Total_instructors_
spring_2018 FROM TEACHES WHERE semester=
spring' AND year='2018';
```

2)

```
SELECT COUNT(*) AS Number_of_tuples FROM
TEACHES;
```

Dept_Name	Avg_salary
comp. sci.	77333.3333
Biology	69000.0000
Finance	85000.0000
Music	40000.0000
Physics	91000.0000
History	61000.0000
Ele. Eng	80000.0000

5)

```
SELECT Dept_Name, AVG(salary) AS Avg_salary
FROM INSTRUCTOR GROUP BY Dept_Name;
```

Dept_Name	Avg_Salary
Comp. Sci.	77333.333
Biology	69000.0000
Finance	85000.0000
Physics	91000.0000
History	61000.0000
Ele. Eng	80000.0000

)

ELECT Dept_Name, AVG(salary) As Avg_Salary
ROM INSTRUCTOR GROUP BY Dept_Name HAVING
vg_Salary > 42000;

Name
Srinivasan
Smith
Tom
Wu
El Said
Gold
Katz
Califeri
Singh
Guck
Brandt
Kim

5)

SELECT Name FROM INSTRUCTOR OR WHERE

Name NOT IN ('Mozart', 'Einstein');

NAme
Wu
Einstein
Gold
Katz
Singh
Crick
Brandt
kim

Name
Wu
Einstein
Gold
Katz
singh
Crick
Brandt
kim

6)

```
SELECT. cName FROM INSTRUCTOR c WHERE
c.salary > ANY (SELECT salary FROM INSTRUCTOR
WHERE Dept_Name = 'Biology');
```

7)

```
SELECT c.Name FROM INSTRUCTOR c WHERE c.salary>
ALL (SELECT Salary FROM INSTRUCTOR WHERE
Dep_Name = 'Biology');
```

Dept-Name	Avg-salary
Comp Sci.	77333.3333
Biology	90000.0000
Finance	85000.0000
Physics	91000.0000
History	81000.0000
Elec-Eng	80000.0000

8)

SELECT Dept_Name AVG (salary) FROM INSTRUCTOR
GROUP BY Dept_Name HAVING AVG(salary) >42000

Dept_Name
Finance
Physics
Comp. Sci.
Finance
Comp. Sci.
Elec. Eng.

EXPERIMENT - 4

1)

```
SELECT Dept_name FROM INSTRUCTOR
WHERE salary > 74513.8462;
```

Name	Course_ID
Srinivasan	CS-101
Srinivasan	CS-315
Srinivasan	CS-347
Wu	FN-201
Mozart	MU-315
Einstein	PHY-101
El Said	HIS-315
Katz	CS-101
Katz	CS-319
Crick	BIO-301
Brandt	CS-190
Brandt	CS-319
Kim	EE-181

2)

SELECT Name, Course_ID FROM INSTRUCTOR
INNER JOIN TEACHES ON INSTRUCTOR.ID = TEACHES
ID;

SELECT INSTURCTOR.NAME, TEACHER COURSE_ID
FROM INSTRUCTOR LEFT JOIN TEACHER ON INSTRUCTOR
ID = TEACHER.ID;

dept_name	total_salary
Comp. Sci	332000
Biology	138000
Finance	17000
Music	4000
Physics	182000
History	122200
Elec. Eng.	80000

5)

4)

CREATE VIEW FACULTY AS SELECT ID, NAME, DEPT.

FROM INSTRUCTOR;

6) GRANT SELECT ON FACULT TO NEW USER

EXPERIMENT-5 ADVANCED SQL

1) CREATE A view of instructor without their salary called faculty

CREATE VIEW FACULTY AS SELECT ID, NAME, dept_name FROM INSTRUCTOR WHERE SALARY IS NULL;

2) Create a view of department salary totals

CREATE VIEW DEPARTMENT_SALARY_TOTALS AS SELECT dept_name, SUM(salary) AS TOTAL_SALARY FROM INSTRUCTOR GROUP BY DEPT_name;

SELECT * FROM DEPARTMENT_SALARY_TOTALS;

3) CREATE a Role of Student.

CREATE ROLE STUDENT;

4) give Select priviliges on the view faculty to the role student.

GRANT SELECT ON FACULTY TO STUDENT;

5) Create a new user and assign her the role of Student

CREATE USER. New_ USER. IDENTIFIED BY:

'PASSWORD'

) Login as this new user and find all
Instructors in the Biology department

SELECT NAME FROM INSTRUCTOR WHERE

Dept_ NAME = 'BIOLOGY';

7) Revoke privileges of the new user

REVOKE STUDENT FROM NEW_USER;

8) Remove the role of Student

DROP ROLE STUDENT;

9) Give Select privileges on the view
faculty to the new user.

GRANT SELECT ON FACULTY TO NEW_USER;

10) Login as this new user and find all
Instructors in the finance Department

SELECT * FROM INSTRUCTOR WHERE DEPT_NAME
= 'FINANCE'.

) Login again as root user

SELECT HOST, USER, Plugin FROM MYSQL.

SER;

(2) Create table teachers2 with same column as teachers but with additional constraint that the semester is one of fall, winter, spring or summer.

SELECT * FROM TEACHERS;

(3) CREATE INDEX TEACHERS_ID_UNDER ON TEACHERS (ID);

(4) DROP INDEX TEACHERS_ID_INDEX ON TEACHERS;

EXPERIMENT - 6

ACCESSING THE DATABASE THROUGH PYTHON

1) Insert following additional tuple in instructor: (1021, 'smith', 'Biology', 6600)

```
mycursor = mydb.cursor()
mycursor.execute("INSERT INTO INSTRUCTOR (ID, NAME, dept_NAME, SALARY) VALUES (%s, %s, %s, %s)", val = (1057, 'Sum', 'BIOLOGY', 46000)
for i in mycursor
    print(i)
```

2) Select this tuple from instructor.

MYCURSOR.EXECUTE ("DELETE FROM INSTRUCTOR WHERE ID = 1057")

3) Select tuples from instructor where Dept_name = 'History'.

MYCURSOR.EXCUTE ("SELECT * FROM INSTRUCTOR WHERE DEP.Name = 'HISTORY'")

4) Find the cartesium product instructor × Teachers

MYCURSOR.EXCUTE ("SELECT * FROM INSTRUCTOR WHERE DEPT_NAME = 'HISTORY'")

5) Find the name of all instructors who have taught some courses and the course_ID

MYCURSOR.EXCUTE ("SELECT NAME, COURSE.ID FROM INSTRUCTOR INNER JOIN TEACHERS ON INSTRUCTOR ID = TEACHES.ID")

6) FIND THE Name of all instructors whose name includes to Sub String

SELECT * FROM INSTRUCTOR WHERE NAME LIKE "Sam";

7) Find the name of all instructors with salary between 90,000 and 10,000

MYCURSOR.EXCETE ("SELECT * FROM INSTRUCTOR CROSS JOIN TEACHER") BETWEEN 90000 AND 10,0000

EXPERIMENT - 7

ADVANCED TECHNIQUES THROUGH PYTHON.

1) ORDER THE TYPES IN THE INSTRUCTORS RELATION AS PER THEIR SALARY

MYCURSOR - EXCUTE ("SELECT * FROM TEACHERS ORDER BY SALARY USC")

2) FIND COURSES THAT ran in Fall 2017 or in Saving 2018

MYCURSOR - EXCUTE ("SELECT COURSE_ID FROM TEACHERS WHERE (SEMESTER = 'FALL' AND YEAR = '2017') OR (SEMESTER = 'SPRING' AND YEAR = '2018')")

3) Find courses that ran in fall 2017 and in spring 2018

MYCURSOR . EXCEUTE ("SELECT COURSE_ID FROM TEACHES WHERE (SEMESTER = 'FALL AND YEAR = '2017') AND (SEMESTER = 'SPRING' AND YEAR = '2018')");

4) Find courses that ran in fall 2017 but not in spring 2018

mycursor.execute("select course_id from teachers where semester = 'Spring' and year = '2018'")

5) Insert following calulational tuples in instructor

mycursor.execute("select * from instructor where salary is null")

6) Find all instructor whose salary in Null

mycursor.execute("insert into instructor, values ('10211', 'Smith', 'Biology', 66000) ('10212', 'Tom', 'Biology', 'Null')").

7) Find the average of instructors in Computer Science department

mycursor.execute("select avg(salary) from instructor where dept_name = 'Computer Science'")

8) Find the total number of instructors who teach a course in the Spring 2018 semester

mycursor.execute("select count(*) from instructor where semester = 'Spring' and year = '2018'");

9) Find the number of tuples in the teaches relation

MYCURSOR.EXCUTE ("SELECT COUNT (*) FROMS INSTRUCTOR").

10) find the average salary of instructores in each department

MYCURSOR.EXCEUTE ("SELECT dept_name, avg (salary) FROMS INSTRUCTOR GROUP by Dept_name")

11) find the names and average salaries of all deparment whose average salary is greater than 1,2000.

MYCURSOR.EXCUTE ("SELECT dept_name, avg(CSALARY 'FROMS INSTRUCTOR GROUP by DEPT_NAME HAVING AVG (SALARY) = 1,2000";))

12) Name all instructors whose name is either "Mozart or Einstein";

MYCURSOR.EXCUTE ("SELECT NAME FROMS INSTRUCTOR WHERE NAME NOT IN ("MOZART", "EINSTEIN"

13) MYCURSOR.EXCUTE ("SELECT N. NAME FROMS INSTRUCTORS n WHERE n.SALARY > ANY (SELECT FROMS DETAILS WHERE dept_NAME = 'BIOLOGY'))

4) MYCURSOR. EXCUTE ("SELECT NAME FROM INSTRUCTOR WHERE SALARY > ALL (SELECT SALARY FROM TEACHERS WHERE DEP_NAME = 'Biology'))

5) MYCURSOR. EXCUTE ("SELECT AVG(SALARY) FROM INSTRUCTORS GROUP BY DEPT_NAME HAVING AVG (SALARY) > 42000")

6) MYCURSOR. EXCUTE ("SELECT DEPT. NAME FROM INSTRUTOR GROUP BY DEPT. NAME HAVING SUM (SALARY > (SELECT AVG (TOTAL. SALARY as TOTAL_SALARY FROM INSTRUCTOR GROUP BY DEPT_NAME) AS SUBQUERY"))

7) MYCURSOR. EXCOUNT ("SELET INSTRUCTOR NAME, TEACHERS, COURSE_ID FROM INSTRUCTOR JOIN TEACHERS ON INSTRUCTORS. ID = TEACHERS. ID")

8) MYCURSOR. EXCUTE ("SELECT INSTRUCTOR NAME, TEACHER COURSE_ID FROM INSTRUCTOR JOIN TEACHERES ON LEFT JOIN TEACHERS ON INSTRUCTOR ID = TEACHERS. ID")

Full_Emp (Emp-ID, Person (Name, Add, city))

Emp_ty ('100' Person_ty ('RAM', Addr_ty ('1000', 'TU', Patiala)

Emp_ty ('101', Person_ty ('Sham' Addr_ty ('1001', 'TU', Patiala)

name	NULL?	Type
Full-emp		emp_ty

ID	name	city
101	Ram	Patiala
101	Sham	Patiala

ID	Name	city
100	Raj	Patiala
101	Sham	Patiala

EXPERIMENT-8

OBJECT ORIENTIAL PROGRAMMING-

ORACLE

1) CREATE A TYPE addr_ty.

```
Create type addr.ty as object
    (street varchar (60),
    (city varchar (30),
        state char(2)
        zip varchar(9));
```

2) CREATE A TYPE PERSON.TY

```
Create type person.ty as object
    (name varchar (25),
    (address addrty);
```

3) CREATE A TYPE EMPTY.

```
CREATE TYPE EMPTY AS OBJECT
    (empl_id varchar(9),
        Person person_ty);
```

4) CREATE TABLE EMP_OO

```
Create table emp_oo
    (full_ emp emp_ty);
```

5) Enter values

```
Enter into emp_oo values
    (emp_ty ('100',
                person_ty ('Ram',
                addr_ty ('1000 10',
```

Full emp

		Empcity
Id	Name	City
101	Ram	Patiala
101	Sham	Patiala

ID	Name	Emb
100	Raj	Patiala
101	Sham	Patiala

'Patides', 'BB', 14100, '111');

Insert into emp_oo values

(emp_ty ('101',

person_ty ('Shami'

addr_ty ('100, TV',

Patride', 'PB', 147001')));

6) select * from emp_oo;

7) desc emp_oo;

8) select e.full_emp.empt_id ID,

a.full_emp.person.name NAME,

e.full_emp person.address.city CITY

from emp_oo e;

9) update

update emp_oo e set

e.full_emp.person.name = 'Raj'

where

e.full.emp.empt_id = '100';

10) select e.full.emp_empt_id ID,

e.full.emp.person.name.NAME,

e.full.emp.person.address.city CITY

From emp.oo e;

11) Create or replace type newemp_ty as object(
first name varchar (25),
last name varchar (25),
birthdate date,
member function age (Birth date in DATE)
return between)

12) Create or replace type body new emp_ty as
member function age (Birthdate in date) return
number is begin
Return Round (sysdate - Birth date);

13) Create table new_emp_oo
(employee new emp_ty);

14) insert into new_emp_oo values
(new emp_ty ('Ram', 'Lal', '12-dec-1976'));

15) How to call the function age()
select e.employee. firstname, e employee.
age c.e.employee. birthdate) from
new_emp_oo e;

16) Creation of object table
Create table new. emp_of emp_ty;

17) Create type emp-ty as object
(empt-id varchar(9),
Person Person-ty);

18) Create table emp.oo
(full-emp emp-ty);

19) Insert into emp-oo values
(empty ('100',
Person-ty ('Ram',
addr-ty('1000'TV'
'Patidles', 'PB', '14700')));

20) Insert into new-emp. values ('100', Person-ty
('Raj', addr-ty ('100TV', 'PTA', 'Pb', 14700.)));

21) select * from new-emp;
Select ref(n) from new-emp;n;

22) Implementing the concept of FK
Create type new-dept-ov as object
(deptno number(3), dname varchar(20));
create table dept-table of new-dept-re;

23) insert into dept-table values (10, 'com');
insert into dept-table values (20, 'chem');

Name	Null?	Type
Empno		Number (3)
Name		Varchar2(10)
Dept		Ref of new_dept_o

Empno	Name	Deptref (e.Dept)
100	Raj	New_dept_o
101	Sham	New_dept_o

Empno	Name	Deptre (e.Dept, Dept
100	Raj	New_Dept(100, 'D
101	Sham	New_Dept(Cursor

Dept no.	Dname
10	Com
120	Che

'insert into dept_table values ('30', 'math');

2) select ref(n) from dept_table n;

25) Create table emp_dept_table
empno number(3),
name varchar(10),
dept_ref ref new_dept_oo);

26) desc empt_test_fk
set desc depth 2
desc emp_test_fk

27) insert into emp_test_fk
set desc depth 2
desc emp_test_fk

28 insert into emp_test_fk
Select 100, 'raj', ref(n) from dept_table n where
deptno = 10;

28) insert into empt_task_fk
select 101, 'sham', ref(n) from dept_table n where
deptno = 20;

29) accessing the values;
select * from emp_test_fk;

select empno, name, deref (e.dept) from
epno.no test, ffk).

select empno, name, deref (a.dept) deref(e.dept,
deptno deptno, deref (e.dept).dname dname
from (emp_test - ffk

30) Create table emp_table - ffk
(employee empty,
dept ref new - dept . 00);

31) set describe depth,
desc emp. table ffk
set emp - table - ffk
set dmp - dept - ffk
set describe depth 3
desc emp - table ffk

32) set describe depth 3
desc emp. table - ffk
set describe depth 4
desc emp - table - ffk

33) insert into emp_table_fk values (

empty_ty (100, person_ty ('Ram', addr_ty ('Hyd', 'Tel'),

(select ref (n)

from dept (n)

where deptno = 10

));

34) select * from emp_table_fk;

35) select e.employe_empt_id ID, e.employee.

person_name name, deref (e.dept), deref (e.dept.)

deptno deptno, deref (e.dept).dname

from emp_table_fk e;