**“Lab 1”**

* **Question 1**

Select emp no, ename, sal from emp

Where job= ‘MANAGER’;

* **Question 2**

Select ename, dept no from emp where hiredate >= ‘1-jan-1982’

* **Question 3**

Select ename from emp

Where job= ‘MANAGER’

And dept no=20

* **Question 4**

Select ename || ‘:1 month salary=’|| sal from emp

**“Lab 2”**

* **Question 1**

Select ename, dept no from emp

Where emp no no=7566

* **Question 2**

Select ename, dept no from emp

Where dept no=10 or dept no=30

Order by ename

* **Question 3**

Select ename, dept no, hiredate from emp

Where to\_char(hiredate, ‘YYYY’)= ‘1982’

* **Question 4**

Select ename from emp

Where ename like (%LL%)

& (dept no=30 or mgr=7782)

* **Question 5**

Select ename from emp

Where job= (‘CLERK’) & dept no in(10,20) & hiredate< ‘1-jan-1983’

* **Question 6**

Select ename, sal from emp

Where sal not(between 1500 or 2850)

* **Question 7**

Select ename, sal, comm from emp

Where comm>sal(1+0.1)

**“Lab 3”**

* **Question 1**

select e.ename, d.dname,d.location

from employee e, department d

where NVL(comm,0)<>0

* **Question 2**

select worker.ename||'works for'|| manager.ename

from emp worker, emp manager

where worker.mgr=manager.empno(+)

* **Question 3**

select w.ename||'works for'||m.ename

from emp w, emp m

where(w.mgr=m.empno AND m.empno = 'KING')

* **Question 4**

select distinct e.job, d.loc

from emp e, dept d

where( e.deptno=d.deptno AND d.deptno=30)

* **Question 5**

select e.ename,e.job, d.deptno, d.dname

from emp e, dept d

where (d.loc= 'DALLAS' AND e.deptno=d.deptno)

* **Question 6**

select e.ename as "employee", e.emp no as "emp#",m.ename as "manager", m.emp no as"manager #" from emp e, emp m

where e.mgr=m.emp no;

**“Lab 4”**

* **Question 1**

1. HSALAR

2. FORMTECH

3. GETING , 3

4. 69.5

5. 1-MAR-90

6. 1-JAN-90

7. 6

8. 16.5

* **Question 2**

**i)**

select \* sysdate current\_date from dual;

**ii)**

select empno, ename, trunc(((sal\*0.15)+sal)) New\_sal, trunc(((sal\*0.15)+sal)) incremented

from emp

**iii)**

select ename, round(months\_between(sysdate, hiredate)) months\_worked

from emp order by months\_worked;

**iv)**

select ename ||'earns'||sal||'monthly'

from emp

**v)**

select initcap(ename), length(ename)

from emp

where(enmae like'J%' OR ename like 'A%' OR ename like 'M%')

**vi)**

select ename, hiredate, to\_char(hiredate, 'day'), to\_char(hiredate\_1,'d')num from emp order by num;

**vii)**

select deptno, job, count(\*), num\_emp from emp

group by deptno, job

**viii)**

select d.dname, d.loc, count(e.empno) no\_of\_emp

round( AVG(e.sal),2)

from emp e, dept d where e.deptno=d.deptno

group by d.dname, d.loc;

**ix)**

select e.ename, e.deptno, e.job

from emp e, dept d

where (d.deptno=e.deptno AND d.loc= 'DALLAS')

**x)**

select (max(sal)-min(sal)) difference

from emp;

**xi)**

select manager.ename, MGR, count(\*)num

from empworker, emp manager

where(worker.mgr=manager.empno)

group by manager.ename;

**“Lab 5”**

* **Question 1**

select empno. ename from emp

where sal> (select avg(sal) from emp)

order by sal desc

* **Question 2**

select ename, sal from emp

where mgr = ( select empno from emp where ename = "KING")

* **Question 3**

select d.name, e.job, d.dept no

from emp e, dept d

where d.deptno = (select dept no from dept where d.dname = 'SALES' )

* **Question 4**

select ename, hiredate, sal from emp

where ( sal, NVL(comm, 0) = (select sal, comm from emp where ename = 'SCOTT')

* **Question 5**

select ename, deptno, job from emp

Natural Join dept where loc = 'DALLAS'

* **Question 6**

select emp.no, job from emp

where job <> 'ANALYST' minus ( select emp no, job from job\_history where job = 'ANALYST')

* **Question 7**

select ename, hiredate from emp

where (deptno IN (select deptno from emp where ename= 'BLAKE') and ename <> 'BLAKE')

* **Question 8**

select empno, ename, sal from emp

where sal > (select avg(sal) from emp) and deptno IN(select deptno from emp where ename like '%T%')

**“Lab 6”**

* **Question 1**

Set echo off

Set verify off

Accept low\_date Date format ‘MM/DD/YYYY’

Prompt ‘Please enter the low date range (MM/DD/YYYY)’;

Accept high\_date Date format ‘MM/DD/YYYY’

Prompt ‘Please enter the high date range (MM/DD/YYYY);

Column employee format A25

Select ename, job, hiredate from emp where hiredate between to\_date(‘slowdate’, ‘MM/DD/YYYY’ ) & to\_date(‘shigh 1 date’,’MM/DD/YYYY’) undefine lowdate undefine highdate

Column employee clear set verify on set echo on

* **Question 2**

Set echo off

Set verify off

Set feedback off

Break on name

Accept p\_location Prompt ‘Please enter the locationname’;

Column dname heading “Department Name” format A15

Column ename heading “Employee Name” format A15

Column sal heading “Salary” format $99,990,00

Column asal heading “Ansal+1” format $99,990,000

Select d.dname,e.ename,e.hiredate,e.sal

e.sal\*12 ansal from emp e, dept d where e.dept no=d.dept no

and lower(d.loc) like low(‘%p\_location%’)

order by d.dname undefine p\_location

column dname clear, column ename clear

column hiredate clear,column sal clear, column ansal clear

clear break set verify on set feedback on set echo on

**“Lab 7”**

* **Question 1**

Insert into emp values (7123, 'Ralph', Designer', 7566, '21-APR-85', 2300, NULL, 50);

Insert into emp values (7890, 'George', 'Clerk', 7566, '03-MAY-85', 1235, NULL, 50);

Insert into emp values (7629, 'Bob', 'Salesman', 7698, '06-Mar-86', 1800, 1000, 30);

* **Question 2**

**a**)

update emp

set sal=sal+250

where job= 'Clerk'

And sal< 900;

**b)**

update emp

set deptno = 20, sal = sal \* 1.15

where empno = 7890

**c)**

update emp

set sal = (select sal \* 1.10 from emp where empno = 7499 where empno = 7369)

**d)**

update emp

set mgr = (select mgr from emp where empno = 7900 where empno = 7876)

**e)**

select from emp

where hiredate > to\_date ('1981', YYYY)

**“Lab 8”**

* **Question 1**

Create table grade(designation varchar(30) constraint grade\_designation\_pk primary key, grade number(2), totalposts number(4), posts available(4))

create table employee(empno number(4) constraints employee\_empno\_pk primary key, name varchar2(18) not null, designation varchar2(30), qualification varchar2(18), joindate date default sysdate, constraint employee\_designation\_fk foreign key(designation) references grade(designation))

create table project(pid varchar2(4) constraint project\_pid\_pk primary key, title varchar2(18) not null, client varchar2(30), duration number(4), status varchar2(18))

create table training(tcode varchar2(5) constraint training\_tcode\_pk primary key, title varchar2(30) not null, startdate date, enddate date)

create table emp\_project(empno number(4), pid varchar2(4), performance varchar2(18), constraint emp\_project\_empno\_fk foreign key(empno) references(employee(empno), constraint emp\_project\_pid\_fk foreign key(pid) references project(pid))

create table emp\_training(empno number(4), tcode varchar2(5), attendance number(9,2), constraint emp\_training\_empno\_fk foreign key(empno) references( employee(empno), constraint emp\_training\_tcode\_fk foreign key(tcode) references training(tcode))

* **Question 2**

Alter table employee add(gender varchar 2(8) where (gender('Male','Female')))

Alter table training add(instructor\_name varchar 2(20))

Atler table grade add(salary \*\*\*\*\*)

* **Question 3**

Insert into grade values('CLERK', 12, 8, 8)

Insert into grade values('PROGRAMMAR', 17, 6, 6)

Insert into grade values('DATABASE DEVELOPER', 17, 4, 4)

Insert into grade values('SENIOR SYSTEM ANALYST', 18, 2, 2)

* **Question 4**

Insert into employee values (3400,'ASIF','DATABASE DEVELOPER','B.E.','25thNov,'2000')

Insert into employee values(3108,'MEHMOOD','SENIOR SYSTEM ANALYST','M.S.','16thMay,1996')

Insert into employee values(3345,'FARHAN','DATABASE DEVELOPER','B.E.','23rdApril,2001')

Insert into employee values(3315,'NAVEED','CLERK','B.A.','11thJan,1997')

Insert into employee values(3300,'NAVEEN','CLERK','B.A.','24thFeb,1998')

* **Question 5**

Insert into training values(03,'Developer 6i','23rdMay,2002','13thJuly,2002')

Insert into training values(13,'Java','14thJan,2001','15thMarch,2001')

Insert into training values(17,'Typing/Short Hand','22ndJan,2001','22ndJuly,2001')

* **Question 6**

Insert into project values(023,'Payroll','Superior Bank',12,'New')

Insert into project values(321,'Accounts','ABC Leasing',16,'In Progress')

Insert into project values(178,'Taxation','Farhan Motors',4,'Complete')

Insert into project values(315,'Payroll','ABC Leasing',8,'New')

Insert into project values(300,'Inventory','Khurram Textile Mills',12,'In Progress')

* **Question 7**

**i)**

Insert into emp\_training values(87,'Developer6i',3400)

**ii)**

Insert into emp\_training values(300,'Typing Short Hand',3300)

* **Question 8**

**i)**

Insert into emp\_project values(3400,023,'Good')

**ii)**

Insert into emp\_project values(3300,023,'Excellent')

**iii)**

Insert into emp\_project values(3345,321,'Bad')

* **Question 9**

**i)**

Create view v1 as select empno,designation from employee

**ii)**

Create view v2 as select e.empno,e.ename,p.title,m.performance from employee e, project p , emp\_project m where(e.empno=manager) and (p.PID=m.PID)

**iii)**

Create view v3 (empno, name, countperf) as select e.ename, e.empno, count(p.performance) from employee e, emp\_project p where((e.empno=p.empno) and p.performance='Excellent')) group by e.empno, e.ename

**“Lab 9”**

* **Question 1**

Create sequence emp\_empno increment by 1 start with 1 max value 10000

* **Question 2**

i) Create index employee\_name\_idx on employee(name)

ii) Create index employee\_designation\_idx on employee(designation)

iii) Create index training\_title\_idx on training(substring(title,1,10))

* **Question 3**

i) Create bitmap index employee\_gender\_idx on employee(gender)

ii) Create bitmap index employee\_project\_performance\_idx on employee\_project(performance)

“**Lab 10”**

* **Question 1**

SET SERVEROUTPUT ON

DECLARE

v\_sal\_ann emp.sal%TYPE;

v\_sal NUMBER(7):=&v\_sal\_ann;

BEGIN

v\_sal:=(1.45\*v\_sal)-(0.11\*v\_sal);

dbms\_output.put\_line('The Net salary is :' || v\_sal);

END;

/

**“Lab 11”**

* **Question 2**

SET SERVEROUTPUT ON

VARIABLE g\_total NUMBER

DECLARE

v\_sal NUMBER(7);

v\_percentage NUMBER(7);

v\_sal\_ann NUMBER(7):=&v\_sal;

v\_bonus\_per NUMBER(3):=&v\_percentage;

BEGIN

:g\_total:=(NVL(v\_sal\_ann,0)\*(v\_bonus\_per/100))+NVL(v\_sal\_ann,0);

END;

/

PRINT g\_total

* **Question 3**

SET SERVEROUTPUT ON

DECLARE

v\_num\_emp NUMBER(4);

v\_max\_sal NUMBER(7);

v\_avg\_sal NUMBER(7);

BEGIN

SELECT COUNT(EMPNO)

INTO v\_num\_emp

from emp

where deptno=10;

SELECT MAX(SAL)

INTO v\_max\_sal

from emp

where deptno=20;

SELECT AVG(SAL)

INTO v\_avg\_sal

from emp;

dbms\_output.put\_line('Number of employees in dept 10 = ' || v\_num\_emp);

dbms\_output.put\_line('Max salary in dept 20 = ' || v\_max\_sal);

dbms\_output.put\_line('Avg salary = ' || v\_avg\_sal);

END;

/

**“Lab 12”**

* **Question 1**

SET SERVEROUTPUT ON

ACCEPT p\_emp\_num PROMPT 'Enter employee number: '

DECLARE

v\_emp\_num emp.empno%TYPE;

v\_emp\_sal emp.sal%TYPE;

v\_emp\_bonus emp.sal%TYPE;

BEGIN

v\_emp\_num := &p\_emp\_num;

SELECT sal

into v\_emp\_sal

from emp

where empno = v\_emp\_num;

if v\_emp\_sal<1000 then

v\_emp\_bonus:=(v\_emp\_sal\*(10/100));

elsif v\_emp\_sal between 1000 and 1500 then

v\_emp\_bonus:=(v\_emp\_sal\*(15/100));

elsif v\_emp\_sal>1500 then

v\_emp\_bonus:=v\_emp\_sal\*(20/100);

elsif v\_emp\_sal=NULL then

v\_emp\_bonus:=0;

end if;

dbms\_output.put\_line(' Bonus Amount = ' || v\_emp\_bonus);

END;

/

* **Question 2**

SET SERVEROUTPUT ON

ACCEPT p\_emp\_num PROMPT 'Enter employee number: '

DECLARE

v\_emp\_num emp.empno%TYPE;

v\_emp\_sal emp.sal%TYPE;

v\_star VARCHAR2(200);

BEGIN

v\_emp\_num := &p\_emp\_num;

SELECT sal

into v\_emp\_sal

from emp

where empno = v\_emp\_num;

LOOP

v\_emp\_sal:=v\_emp\_sal-100;

EXIT WHEN v\_emp\_sal<100;

v\_star:=v\_star || '\*’;

END LOOP;

dbms\_output.put\_line(' Stars = ' || v\_star);

END;

/

**“Lab 13”**

* **Question 1**

SET SERVEROUTPUT ON

DECLARE

v\_n NUMBER(3);

v\_emp\_count NUMBER(3);

CURSOR emp\_cursor IS

SELECT ename,sal

from emp

order by sal DESC;

emp\_record emp\_cursor%ROWTYPE;

BEGIN

v\_emp\_count:=&v\_n;

OPEN emp\_cursor;

FOR I IN 1..v\_emp\_count

LOOP

FETCH emp\_cursor INTO emp\_record;

EXIT WHEN emp\_cursor%NOTFOUND;

dbms\_output.put\_line('employee' || emp\_record.ename || ',' || emp\_record.sal);

END LOOP;

close emp\_cursor;

END;

/

* **Question 2**

SET SERVEROUTPUT ON

DECLARE

v\_emp\_sal NUMBER(7);

CURSOR emp\_cursor IS

SELECT empno,ename,sal

from emp;

emp\_record emp\_cursor%ROWTYPE;

BEGIN

OPEN emp\_cursor;

LOOP

FETCH emp\_cursor INTO emp\_record;

EXIT WHEN emp\_cursor%NOTFOUND;

emp\_record.sal:= (emp\_record.sal+0.45\*emp\_record.sal+0.12\*emp\_record.sal+0.15\*emp\_record.sal)-(0.20\*emp\_record.sal);

dbms\_output.put\_line('Employee Number: ' || emp\_record.empno || ' Employee Name: ' || emp\_record.ename || ' Net salary: ' || emp\_record.sal);

END LOOP;

close emp\_cursor;

END;

/

**“Lab 14”**

* **Question 1**

**a)**

WHEN too\_many\_rows THEN

INSERT INTO message (result)

VALUES ('More than one employee with a salary of '||

TO\_CHAR(v\_emp\_sal));

**b)**

EXCEPTION

WHEN no\_data\_found THEN

INSERT INTO message (result)

VALUES ('No employee with a salary of '|| TO\_CHAR(v\_emp\_sal));

**c)**

BEGIN

SELECT last name

INTO v\_ename

FROM employee

WHERE salary = v\_emp\_sal;

INSERT INTO messages (result)

VALUES (v\_ename || ' - ' || v\_emp\_sal);

**d)**

WHEN other THEN

INSERT INTO message (result)

VALUES ('some other error occurred');

END;

* **Question 2**

**a)**

SET ECHO OFF

SET SERVEROUTPUT ON

DEFINE EmpSal NUMBER(4)

ACCEPT EmpSal PROMPT 'Enter the Salary'

declare

SAL number(5):='&EmpSal';

Empno NUMBER(4);

begin

SELECT count(\*) INTO Empno FROM EMPLOYEES where SALARY BETWEEN (sal-100) and (sal+100);

If Empno =0 THEN

RAISE NO\_DATA\_FOUND;

ELSE

DBMS\_OUTPUT.PUT\_LINE(EmpNo);

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('There is no employee in that salary range');

END;

**b,c)**

SET ECHO OFF

SET SERVEROUTPUT ON

DEFINE EmpSal NUMBER(4)

ACCEPT EmpSal PROMPT 'Enter the Salary'

declare

NO\_DATA\_FOUNDING EXCEPTION;

sal NUMBER(5):='&EmpSal';

Empno NUMBER(3);

begin

SELECT count(\*) INTO Empno FROM EMPLOYEE where SALARY BETWEEN (sal-100) and (sal+100);

if EMPNO =0 then

RAISE NO\_DATA\_FOUNDING;

ELSE

DBMS\_OUTPUT.PUT\_LINE(EmpNo||' Employees are in salary range');

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUNDING THEN

DBMS\_OUTPUT.PUT\_LINE('There is no emp in that salary range');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('SOME OTHER ERRORS');

END;

**“Appendix A”**

* **Question 1**

A trigger is a PL/SQL block that executes implicitly whenever a particular event takes place. Database triggers executes implicitly whenever a particular event occurs within an application. A row trigger fires each time the table is affected by the triggering event. If the triggering event affects no rows, a row trigger is not executed at all.

* **Question 2**

A statement trigger is fired once on behalf of the triggering event, even if no rows are affected at all. Statement triggers are useful if the trigger action does not depend on data of rows that are affected or data provided by the triggering event itself. For example, a trigger that performs a complex security check on the current user. A Row trigger fires each time the table is affected by the triggering event. If the triggering event affects no row(s), a row trigger is not executed at all. Row triggers are useful if the trigger action depends on data of rows that are affected or data provided by the triggering event itself.

* **Question 3**

The triggering event or statement can be an INSERT, UPDATE, or DELETE statement on a table.

* When the triggering event is an UPDATE, we can include a column list to identify which column(s) must be changed to fire the trigger. We cannot specify a column list for an INSERT or for a DELETE statement, as they always affect entire rows.
* The triggering event can contain multiple DML statements. In this way, we can differentiate what code to execute depending on the statement that caused the triggers.
* **Question 4**
* **Question 5**
* **Question 6**

**“Appendix B”**

* **Question 1**
* **IN TYPE PARAMETER**

Passes a constant value from the calling environment into theprocedure.

* **OUT TYPE PARAMETER**

Passes a value from the procedure to the calling environment.

* **IN OUT PARAMETER**

Passes a value from the calling environment into the procedure and a possibly different value from the procedure back to the calling environment using the same parameter.

* **Question 2**

A variable pass as mode IN is always read only and used by procedure but cannot be changed. IN OUT mode has characteristic of both IN and OUT mode. The variable value is passed in and can be read by procedure. The procedure can also change the value and it will copied back to the passed variable when procedure completes.

* **Question 3**

Create or replace procedure project.check

(V\_project id IN project.project D % type,

V\_title out project.title % type , V\_clientname OUT project\_clientname % type, V\_dur project.duration % type, V\_status OUT project.status % type)

Is begin select title, client\_name,duration,status INTO V\_title, V\_clientname,V\_dur, V\_status from project where project id =V\_project id;

End project.check

* **Question 4**

Create or replace update desg

(V\_empno IN emp.empno % type ,V\_desg IN emp.desg % type, V\_grade OUT emp.grade % type)

Is begin update emp set desg = V\_desg where empno=V\_empno

Dbms\_ouput.put\_line(‘New grade is’|| V\_grade)

End update\_desg

End;

* **Question 5**

**“Appendix C”**

* **Question 1**

Functions and procedures are structured alike, except that a function must return a value to the calling environment. Functions promote reusability and maintainability. Once validated, they can be used in any number of applications. If the definition changes, only the function is affected, this greatly simplifies maintenance. Functions can be called as part of a SQL expression or as part of a PL/SQL expression.

* **Question 2**

Maintainability, testing, isolation of business roles, speed, optimization, utilization of set based processing, security.

* **Question 3**

Create or replace function perform(V\_empno IN number)

Return number

Is V\_percent number;

V\_pid emp\_project.pid % type;

V\_perform emp\_project.performance % type;

Begin

Select performance pid, into V\_perform, from emp\_project V\_pid

Where empno=V\_empno;

If V\_perform= ‘excellent’then

V\_percent:=(1/count(V\_pid))/100;

Else Dbms \_ouput.put\_line( ‘No excellent performance’||V\_emp);

End if;

Return V\_perform;

End