SQL

- **Data Types:-**

1. Varchar(len) Or Varchar2(len) :- Used to store variable lengths pf alpha-numeric data not padded with space.

2. Char(len) :- Used to specify fixed length characters, maximum size of 255 characters. Extra size is padded with white space.

3. Number(p, s) :- Used to store numbers fixed of floating. P is the precision or length and s is the scale that determines the no. of places to right of decimal.

4. Date :- Used to represent date and time. DD - MM - YYYY

- **Commands:-**

S.no	Syntax	Usage
1.	Create database db_name;	To create a database with db_name as the name of the database.
2.	create table <table name> (col1 data type, col2 data type…)	To create a table of n columns.
3.	insert into <table name> values (val1,val2,…)	To insert n values into the table.
4.	select * from tab;	To view all existing system tables and databases.
5.	select * from <table name>;	To view all rows and columns of a given table.
6.	select (col) from <table name>;	To view selected columns and all rows.

20.	drop database <db_name>;	Used to delete a database completely.
7.	select * from <table name> where <exp>;	To view selected rows and all columns based on given condition.
21.	drop table <table name>;	Used to delete a table completely including its schema.
8.	select distinct(col1, clo2, ···) from <table name>;	To retrieve all the distinct fields in the table.
22.	truncate table <table name>;	Used to delete a table leaving behind its schema.
9.	create table <table name> (col name) as select (col1, col2,···) from <old table name>;	To create a table from an existing table.
23.	grant <system privilege> on <table> to <user>;	Grants the specified privileges on a particular table to the users.
24.	revoke <system privilege> on <table> of <user>;	Used to revoke the granted privileges on a table from the users.
10.	select * from <table name> order by (col);	Sorts a table in ascending order based on given expression.
25.	select (column) from <table name> group by <colname> having exp;	Groups all the rows with same column value based on a given condition.
11.	select * from <table name> order by (col) desc;	Sorts the table in descending order based on given expression.
12.	delete from <table name>;	Removes all rows.
13.	delete from <table name> where <cond>;	Removes selected rows based on given condition.
14.	update <table name> set colname= exp where colname = exp1;	Updates only one row depending on expression.
15.	update <table name> set colname = exp;	Updates all rows/ a complete column.
16.	alter table <table name> add (col data type);	Adds a new column to the table.
17.	alter table <table name> modify (col name data type);	Modifies the existing column.
18.	rename <old table name> to <new table name>;	Renaming a table;
19.	desc <table name>;	Describes structure of a table.

- **Aggregate Functions:-**

S.no.	Funtion	Usage
1.	AVG	Used to find the average of the values in a given column.
2.	MAX	Used to find the maximum or the largest of all the values in a given column.
3.	MIN	Used to find the smallest of all the values in a given column.
4.	COUNT	It returns the no. of rows which satisfy a particular condition which is given in the where clause.
5.	SUM	It is used to find the sum of all values in a column.
6.	STDDEV	It is used to find the standard deviation of all the values in a given column.
7.	VAR	It gives the variance i.e. square of the standard deviation of all the values in a given column, which is a very useful measure in statistics.

- **Date and Time functions:-**

S.no.	Function	Usage
1.	SYSDATE	Used to find the system date.
2.	ADD_MONTHS(date, months);	Used to add a no. of months to the date.
3.	MONTHS_BETWEEN(date1, date2);	Returns the no. of months between the two dates.
4.	NEXT_DAY(date, day);	Returns the name of the first day of the week that is equal to or later than another specified date.
5.	LAST_DAY(date);	Returns the last day of the specified month.
6.	NEW_TIME	It is used to adjust the time according to the desired time zone.

- **Number Functions:-**

S.no	Function	Usage
1.	ABS(n)	Returns the absolute value of the argument passed.
2.	POWER(**m**, n)	Returns the value of m raised to the power of n.
3.	ROUND(n)	Rounds the value of the argument to the nearest integer.
4.	SQRT(n)	Returns the square root of the argument passed.
5.	EXP(n)	Returns the value of E raised to the power n.
6.	LN(n)	Returns the natural logarith**m** value of argu**m**ent.
7.	LOG(n)	Returns the value of logarith**m**ic value of the argu**m**ent.
8.	CEIL(n)	Returns the s**m**allest integer greater than or equal to that of the argument passed.

9.	FLOOR(n)	Returns the greatest integer smaller than or equal to that of the argument passed.
10.	SIGN(n)	Judges whether a given no. is positive or negative.
11.	MOD(a, b)	Returns the remainder obtained when a is divided by b.
12.	COS(a), SIN(a), TAN(a)	Returns the trigonometric ratios of 'a' radians.
13.	COSH(a), SINH(a), TANH(a)	Returns the hyperbolic trigonometric values of the argument passed in radians

- **String Functions:-**

S.no.	Function	Value
1.	LOWER(A)	Returns all the characters in the argument converted into lower case.
2.	UPPER(A)	Returns all the characters in the argument converted into upper case.
3.	LENGTH(A)	Returns the length of the string passed as argument.
4.	CONCAT(A, B)	Concatenates two strings A and B.
5.	CHR(n)	Returns the character equivalent to the no. passed in the argument.
6.	INITCAP(A)	Returns the string passed as an argument with the first letter capitalized.
7.	REPLACE	Used to replace a portion of a string with a new string.
8.	SUBSTR	Returns a set of characters from the given string.
9.	INSTR	Used to find where in a string a particular pattern

		occurs.

- **Conversion Functions:-**

S.no.	Function	Usage
1.	TO_CHAR	Used to convert a number or a date into character format.
2.	TO_NUMBER	Used to convert characters into numbers.

SUPPLIER - PARTS DATABASE

/* TABLES INVOLVED */

1. SUPPLIER(SID, SNAME, CITY)
2. PARTS (PID, PNAME, COLOR)
3. ORDERS (SID, PID, QUANTITY)

/* SUPPLIER DATABASE */

SID	SNAME	CITY
s1	smith	london

s2	jones	paris
s3	blake	paris
s4	clark	london
s5	adams	athens

/* PARTS TABLE */

PID	PNAME	COLOR
p2	bolt	green
p3	screw	blue
p4	screw	red
p5	cam	blue
p6	cog	red

/*ORDERS TABLE */

SID	PID	QUANTITY
s1	p1	300
s1	p2	200
s1	p3	400
s1	p4	200
s1	p5	100
s1	p6	100
s2	p1	300
s2	p2	400
s3	p2	200

s4	p2	200
s4	p4	300

/* QUERIES */

1. Get Sid, sname for non Paris parts.
2. Get full details of all suppliers.
3. Get the total no. of suppliers.
4. Get maximum and Minimum Quantity for part p2.
5. For each part supply gets the part no. and total shipment quantity.
6. Get part number for all parts supplied by more than one supplier.
7. Get supplier names for suppliers who supplied at least one red part.
8. Get the names of the suppliers who supply the part p2.
9. Get the supplier names for suppliers who do not supply part p2.
10. Get snames who supply all parts.
11. Get pname for parts supplied by xyz supplier and by no one else.
12. Find the sid of suppliers who supply red or green parts.
13. Find the sid of suppliers who supply both red and green parts.

EMPLOYEE DATABASE

/* TABLES INVOLVED */

1. EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO).
2. DEPT (DEPTNO, DNAME, LOC).
3. SALGRADE (GRADE, LOSAL, HISAL).

1) EMP Table

Name	Null?	Type

EMPNO	NOT NULL	NUMBER(5)
ENAME		VARCHAR2(10)
JOB		VARCHAR2(9)
MGR		NUMBER(4)
HIREDATE		DATE
SAL		NUMBER(4)
COMM		NUMBER(4)
DEPTNO		NUMBER(6)

2) DEPT Table

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(6)
DNAME		VARCHAR2(20)
LOC		VARCHAR2(20)

3) SALGRADE Table

Name	Null?	Type
GRADE		NUMBER(5)
LOSAL		NUMBER(6)
HISAL		NUMBER(6)

```
/* DEPT TABLE */
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

```
/* EMP TABLE */
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000	0	20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500		30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

```
/* SALGRADE TABLE */
```

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

/* QUERIES */

1. Display all different job types.
2. List the details of all employees in deptno 10 and 20 in alphabetical order.

3. List the names of employees who have "th" or "ll" in their names.
4. List the names, jobs and salaries of all employees who have a manager.

5. Give name and remuneration of all employees.
6. List name and salary increased by 15% of all employees.
7. Find all the employees who were hired during 1982.

8. Display name, annual salary, commission of all salesmen whose monthly salary is greater than commission.

9. Produce the output as "Smith has held the position of clerk in department 20 since 17-dec-80".

10. Find average salary and average total remuneration of all employees other than sales man
11. Find maximum, minimum and average salaries in each department.

12. Find the maximum, minimum and average salaries of each job.
13. Find the departments which have more than 3 employees.
14. Display employee names and their respective department numbers.
15. Give the salary grades for all the employees.
16. Display the employee names who earn highest salary in each job.
17. Find the employee details whose salary is greater than Blake's salary.
18. Find employee details of employees who have the same job and salary as that of Scott.
19. Display the maximum salaries in accounting and research department.

20. Display salary grades of all employees except of those employees whose salary grade is 3 and 4.

21. Give the names and salaries of the employees whose salary is maximum in their respective departments.
22. List the employees whose salary is greater than the salaries of all employees who are working as salesmen.

23. Write a query which will return the day of the week entered in the format of sysdate.
24. Find the difference between highest and lowest salaries.
25. Generate the output as Smith - clerk.
26. Generate the output as Smith (Clerk).
27. Give the details of all employees those who r working as manager.
28. List the departments where there are no employees
29. Generate the following list :

EMPLOYEE NAME JOB SAL GRADE

30. List the information of those employees in department number 10.
31. Find the department location of James.
32. Get the manager of Jones.
33. Get the sub-ordinates of Jones.
34. Write a query to calculate the length of time of all employees with the company.
35. List out name, job, salary, grade, department name of all employees who are not clerks.
36. List out name, job, salary, grade, department name of all employees who are clerks.
37. Display the most recently employed employee in each department.
38. Display each employee with name, hiredate and review date - 1 year from now.
39. Display the year in which most people joined.
40. List out minimum salary of those employees under manager.
41. Print the employee names and List salary as:
    i. 'above target' if salary is greater than 1500
    ii. 'below target' if salary is lesser than 1500
    iii. 'on the target' if salary is exactly 1500.
42. Display ename, loc and dnames of the employees who earn more than 1500 salary.
43. Who are the top 3 earners in the company.
44. Write a query to display a '*' against the employee who joined most recently.
45. Find the most recently joined employee in the company in each department in order of hiredate.
46. Find all the employees who joined before their manager.
47. List jobs and department names of employees whose names are 5 letters long and must begin with 'ALL' and end with 'N'.
48. List employees whose commission is greater than 25% of their salary.
49. List all the departments with at least 2 clerks.
50. List employees in dept 30 with job not found in dept 10.
51. Who earns the second highest salary?
52. Display the employee hiredate in the format of "Friday, 10-AUG-03".

53. Count the people in department 30 who receive both salary and commission.
54. List all the employees who were hired between " 1st January 1981" and "15th April 1985". In the order of their seniority.

55. Display name and period of service till date.

PL/SQL Programs

1) Sum and product of two numbers.

2) Sum of first 'n' natural numbers.

3) Numbers in reverse order.

4) Even numbers up to 100.

5) Sum of Digits of a number.

6) Multiplication table of 'n'.

7) 'n' Multiplication tables.

8) Reverse of a number.

9) Prime number or not?

10) Fibonacci Series.

11) Prime Numbers up to 'n'.

12) Factorial of a given number.

13) Area of a circle.

14) Areas Table.

15) Armstrong or not?

16) Armstrong numbers up to 'n'.

17) Greatest of 2 numbers.

18) And some other sample programs in PL/SQL

CURSORS

1) Cursor programs to find :-

2) Details of employees

3) Employees in department 20.

4) Hike in salary.

5) Total number of employees.

6) Number of employees in each job.

TRIGGERS

1) Trigger on Insertion.

2) Trigger on Updating.

3) Statement Level Trigger.

4) Row Level Trigger.

5) Trigger on Deletion.

S. NO.	Exp No.	Name Of The Experiment	Date	Remarks
I		**Commands and Functions in SQL**		
		SQL		
01.	**01**	**Supplier – Parts Database.**		
02.	**02**	**Employee Database**		
II		PL/SQL		
01.	**03**	**Sum and product of two numbers.**		
02.	**04**	**Sum of first 'n' natural numbers.**		
03.	**05**	**Numbers in reverse order.**		
04.	**06**	**Even numbers up to 100.**		
05.	**07**	**Sum of Digits of a number.**		
06.	**08**	**Multiplication table of 'n'.**		
07.	**09**	**'n' Multiplication tables.**		
08.	**10**	**Reverse of a number.**		
09.	**11**	**Prime number or not?**		
10.	**12**	**Fibonacci Series.**		
11.	**13**	**Prime Numbers up to 'n'.**		
12.	**14**	**Factorial of a given number.**		
13.	**15**	**Area of a circle.**		
14.	**16**	**Areas Table.**		
15.	**17**	**Armstrong or not?**		
16.	**18**	**Armstrong numbers up to 'n'.**		
17.	**19**	**Greatest of 2 numbers.**		
18.	**20**	**Palindrome Number.**		
19.	**21**	**Greatest of three numbers.**		
20.	**22**	**Adams Number.**		

III		CURSORS		

		Cursor programs to find :-			

SUPPLIER – PARTS DATABASE

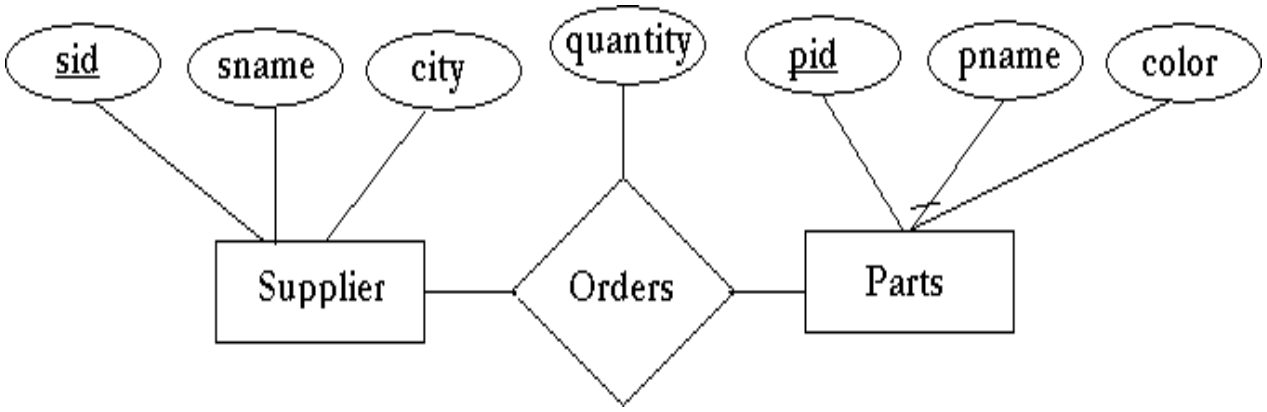
/* TABLES INVOLVED */

4. SUPPLIER(SID, SNAME, CITY)
5. PARTS (PID, PNAME, COLOR)
6. ORDERS (SID, PID, QUANTITY)

/* DDL STATEMENTS */

```
SQL> create table supplier (Sid varchar(3) primary key, sname varchar(20),
        city varchar(20));

     Table created.

SQL> create table parts(pid varchar(3) primary key , pname varchar(20),
     color varchar(20));

     Table created.

SQL> create table orders (sid varchar(3), pid varchar(3) , quantity number(4) ,
     foreign key(sid) references supplier(sid),
     foreign key(pid) references parts(pid) , primary key(sid , pid));

     Table created.

SQL> describe supplier;

 Name                      Null?            Type
 ------------------------- ---------------- ------------------------------
 SID                       NOT NULL         VARCHAR2(3)
 SNAME                                      VARCHAR2(20)
 CITY                                       VARCHAR2(20)

SQL> describe parts;

 Name                      Null?            Type
 ------------------------- ---------------- ------------------------------
 PID                       NOT NULL         VARCHAR2(3)
 PNAME                                      VARCHAR2(20)
 COLOR                                      VARCHAR2(20)
```

ER DIAGRAM FOR SUPPLIER - PART DATABASE



```
SQL> describe orders;
```

Name	Null?	Type
SID	NOT NULL	VARCHAR2(3)
PID	NOT NULL	VARCHAR2(3)
QUANTITY		NUMBER(4)

/* DML STATEMENTS */

```
SQL> insert into supplier values('s1', ' smith' , 'london' );

        1 row created.

SQL> insert into supplier values('s2', ' jones' , 'paris');

        1 row created.

SQL> select * from supplier;
```

/* SUPPLIER DATABASE */

SID	SNAME	CITY
s1	smith	london
s2	jones	paris
s3	blake	paris
s4	clark	london
s5	adams	athens

```
        5 rows selected.

SQL> insert into parts values('p1', ' nut' , 'red');

        1 row created.

SQL> insert into parts values('p2', ' bolt' , 'green' );

        1 row created.
```

SQL> select * from parts;

/* PARTS TABLE */

PID	PNAME	COLOR
p1	nut	red
p2	bolt	green
p3	screw	blue
p4	screw	red
p5	cam	blue
p6	cog	red

6 rows selected.

SQL> insert into orders values('s1', 'p1' , 300);

1 row created.

SQL> insert into orders values('s2', 'p1' , 400);

1 row created.

SQL> select * from orders;

/*ORDERS TABLE */

SID	PID	QUANTITY
s1	p1	300
s1	p2	200
s1	p3	400
s1	p4	200
s1	p5	100
s1	p6	100
s2	p1	300
s2	p2	400
s3	p2	200
s4	p2	200
s4	p4	300

/* QUERIES */

1. Get Sid, sname for non Paris parts.

SQL> select sid, sname from supplier where city != 'paris';

SID	SNAME
s1	smith
s4	clark
s5	adams

2. Get full details of all suppliers.

SQL> select * from supplier;

SID	SNAME	CITY
s1	smith	london
s2	jones	paris
s3	blake	paris
s4	clark	london
s5	adams	Athens

3. Get the total no. of suppliers.

SQL> select count(distinct(sid)) suppliers from orders;

SUPPLIERS
4

4. Get maximum and Minimum Quantity for part p2.

SQL> select max(quantity), min(quantity) from orders where pid='p2';

MAX(QUANTITY)	MIN(QUANTITY)
400	200

5. For each part supply gets the part no. and total shipment quantity.

```
SQL> select pid, sum(quantity) from orders group by pid;

PID    SUM(QUANTITY)
------------  ----------------------------------------
  p1              600
  p2             1000
  p3              400
  p4              500
  p5              500
  p6              100

6 rows selected.
```

6. Get part number for all parts supplied by more than one supplier.

```
SQL> select pid from orders group by pid having count(sid)>1;

PID
---------
p1
p2
p4
p5
```

7. Get supplier names for suppliers who supplied at least one red part.

```
SQL> select sname from supplier where sid in
  2 (select distinct(sid) from orders where pid in
  3 (select pid from parts where color='red'));

SNAME
----------------------------------
smith
jones
clark
```

8. Get the names of the suppliers who supply the part p2.

```
SQL> select sname from supplier where sid in
  2 (select distinct(sid)from orders where pid='p2');

SNAME
----------------------
```

smith
jones
blake
clark

9. Get the supplier names for suppliers who do not supply part p2.

```
SQL> select sname from supplier where sid not in
  2 (select sid from orders where pid!='p2');

SNAME
----------
blake
adams
```

10. Get snames who supply all parts.

```
SQL> select sname from supplier where sid in
  2 (select sid from orders group by sid having count(pid)=6);

SNAME
----------
smith
```

11. Get pname for parts supplied by xyz supplier and by no one else.

```
SQL> select pname from parts where pid in
  2 (select pid from orders group by pid having count(sid)=1);

PNAME
----------
screw
      cog
```

12. Find the sid of suppliers who supply red or green parts.

```
SQL> select distinct(sid) from orders where pid in
  2 (select pid from parts where color= 'red' or color='green');

SID
---
s1
s2
s3
```

s4

13. Find the sid of suppliers who supply both red and green parts.

```
SQL> select distinct(sid) from orders where pid in
  2 (select pid from parts where color= 'red' and sid in
  3 (select sid from orders where pid in
  4 (select pid from parts where color= 'green')));

SID
---
s1
s2
        s4
```

EMPLOYEE DATABASE

/* TABLES INVOLVED */

1. EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO).
2. DEPT (DEPTNO, DNAME, LOC).
3. SALGRADE (GRADE, LOSAL, HISAL).

/* DDL STATEMENTS*/

```
SQL> create table dept (deptno number(6) primary key, dname varchar(20),
          loc varchar(20)) ;

         Table created.

SQL> create table emp (empno number(5) primary key,  ename varchar(10),
          job  varchar(10),mgr number(4), hiredate date, sal number(4), comm
number(4),                deptno number(6), foreign key (deptno) references
dept(deptno)) ;

         Table created.

SQL> create table salgrade (grade number(5) , losal number(6), hisal number(6));

         Table created.

SQL> describe emp;

         EMP Table

         Name                    Null?                    Type
         ----------------------- -------------------------------------------------------------------------
         EMPNO                   NOT NULL                 NUMBER(5)
         ENAME                                            VARCHAR2(10)
         JOB                                              VARCHAR2(9)
         MGR                                              NUMBER(4)
         HIREDATE                                         DATE
         SAL                                              NUMBER(4)
         COMM                                             NUMBER(4)
         DEPTNO                                           NUMBER(6)
```

SQL> describe dept;

DEPT Table

```
 Name                      Null?                    Type
 ------------------------- -------- ------------------------------
 DEPTNO                    NOT NULL                 NUMBER(6)
 DNAME                                              VARCHAR2(20)
 LOC                                                VARCHAR2(20)
```

SQL> describe salgrade;

```
SALGRADE Table

Name                      Null?            Type
 ------------             ----------       -------------------------
 GRADE                                              NUMBER(5)
 LOSAL                                              NUMBER(6)
   HISAL                                           NUMBER(6)
```

/*DML STATEMENTS */

SQL> insert into dept values (10, 'ACCOUNTING' , ' NEW YORK');

1 row created.

SQL> insert into dept values (20, 'RESEARCH', ' DALLAS');

1 row created.

SQL> select * from dept;

/* DEPT TABLE */

```
DEPTNO      DNAME                          LOC
 ----------------  ------------------------------  ----------------------------------------
    10          ACCOUNTING                 NEW YORK
    20          RESEARCH                   DALLAS
    30          SALES                      CHICAGO
    40          OPERATIONS                      BOSTON
```

4 rows selected.

SQL> insert into emp values (7369, 'SMITH', 'CLERK', 7902, '17-DEC-80',
1200, NULL, 20);

1 row created.

SQL> insert into emp values (7499, 'ALLEN' , 'SALESMAN' , 7698, '20-FEB-81',
1600, 300, 30);

1 row created.

SQL> select * from emp;

/* EMP TABLE */

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000	0	20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500		30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23- JAN-82	1300		10

SQL> insert into salgrade values (1, 700, 1200);

1 row created.

SQL> insert into salgrade values (1, 700, 1200);

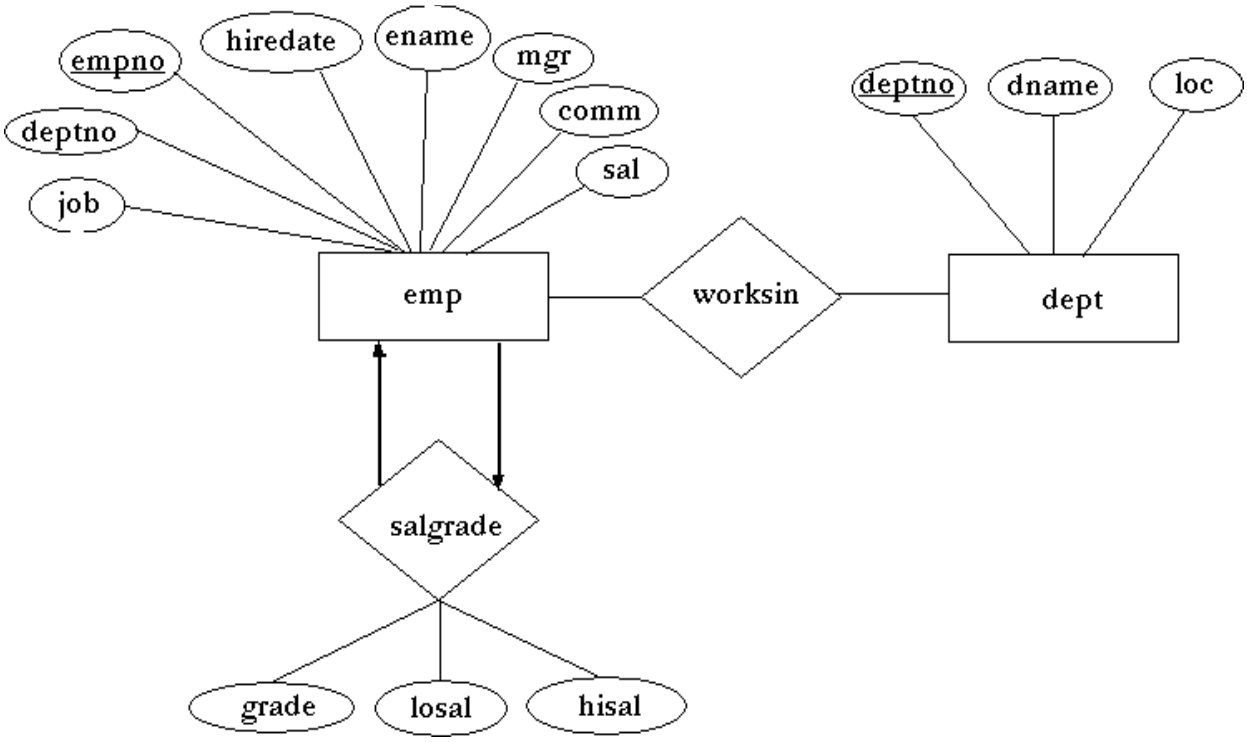
1 row created.

SQL> select * from salgrade;

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

5 rows selected.

ER DIAGRAM FOR EMPLOYEE DATABASE



/* QUERIES */

1. Display all different job types.

SQL> select distinct(job) from emp;

JOB

ANALYST

CLERK

MANAGER

PRESIDENT

SALESMAN

2. List the details of all employees in deptno 10 and 20 in alphabetical order.

SQL> select * from emp where deptno in(10,20) order by ename;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	-800		20
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000	0	20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

8 rows selected

.

3. List the names of employees who have “th” or “ll” in their names.

SQL> select ename from emp where ename like ('%TH%') or ename like('%LL%');

ENAME

SMITH

ALLEN

MILLER

4. List the names, jobs and salaries of all employees who have a manager.

SQL> select ename, job, sal from emp where mgr is not null;

ENAME	JOB	SAL
SMITH	CLERK	800
ALLEN	SALESMAN	1600
WARD	SALESMAN	1250
JONES	MANAGER	2975
MARTIN	SALESMAN	1250
BLAKE	MANAGER	2850
CLARK	MANAGER	2450
SCOTT	ANALYST	3000
KING	PRESIDENT	5000
TURNER	SALESMAN	1500
ADAMS	CLERK	1100
JAMES	CLERK	950
FORD	ANALYST	3000
MILLER	CLERK	1300

13 rows selected.

5. Give name and remuneration of all employees.

SQL> select ename, (sal + nvl(comm,0)) remuneration from emp;

ENAME	REMUNERATION

SMITH	800
ALLEN	1900
WARD	1750
JONES	2975
MARTIN	2650
BLAKE	2850
CLARK	2450
SCOTT	3000
KING	5000
TURNER	1500
ADAMS	1100
JAMES	950
FORD	3000
MILLER	1300

14 rows selected.

6. List name and salary increased by 15% of all employees.

SQL> select ename, sal*115/100 increased_salary from emp;

ENAME	INCREASED_SALARY
SMITH	920
ALLEN	1840
WARD	1437.5
JONES	3421.25
MARTIN	1437.5
BLAKE	3277.5
CLARK	2817.5
SCOTT	3450
KING	5750
TURNER	1725
ADAMS	1265
JAMES	1092.5
FORD	3450
MILLER	1495

14 rows selected.

7. Find all the employees who were hired during 1982.

```
SQL> select ename from emp where hiredate like('%82');

ENAME
-------
MILLER
```

8. Display name, annual salary, commission of all salesmen whose monthly salary is greater than commission.

```
SQL> select ename, sal*12 anual_salary, comm*12 anual_commission from emp
  2  where job='SALESMAN' and sal> nvl(comm , 0);
```

ENAME	ANUAL_SALARY	ANUAL_COMMISSION
ALLEN	19200	3600
WARD	15000	6000
TURNER	18000	0

9. Produce the output as “Smith has held the position of clerk in department 20 since 17-dec-80”.

```
SQL>   select ename || ' HAS HELD THE POSITION OF ' || job || ' IN
       DEPARTMENT '  2  || deptno || ' SINCE ' || hiredate details_of_employees
              from emp;

DETAILS_OF_EMPLOYEES
--------------------------------------------------
SMITH HAS HELD THE POSITION OF CLERK IN DEPARTMENT 20
SINCE 17-DEC-80

ALLEN HAS HELD THE POSITION OF SALESMAN IN DEPARTMENT 30
SINCE 20-FEB-81

WARD HAS HELD THE POSITION OF SALESMAN IN DEPARTMENT 30
SINCE 22-FEB-81

JONES HAS HELD THE POSITION OF MANAGER IN DEPARTMENT 20
SINCE 02-APR-81

MARTIN HAS HELD THE POSITION OF SALESMAN IN DEPARTMENT30
SINCE 28-SEP-81

BLAKE HAS HELD THE POSITION OF MANAGER IN DEPARTMENT 30
SINCE  01-MAY-81

CLARK HAS HELD THE POSITION OF MANAGER IN DEPARTMENT 10
SINCE 09-JUN-81

SCOTT HAS HELD THE POSITION OF ANALYST IN DEPARTMENT 20
SINCE  19-APR-87

KING HAS HELD THE POSITION OF PRESIDENT IN DEPARTMENT 10
SINCE  17-NOV-81

TURNER HAS HELD THE POSITION OF SALESMAN IN DEPARTMENT 30
SINCE  08-SEP-81
```

ADAMS HAS HELD THE POSITION OF CLERK IN DEPARTMENT 20 SINCE 23-MAY-87

JAMES HAS HELD THE POSITION OF CLERK IN DEPARTMENT 30 SINCE 03-DEC-81

FORD HAS HELD THE POSITION OF ANALYST IN DEPARTMENT 20 SINCE 03-DEC-81

MILLER HAS HELD THE POSITION OF CLERK IN DEPARTMENT 10 SINCE 23-JAN-82

10. Find average salary and average total remuneration of all employees other than sales man

SQL> select avg(sal), avg(sal + nvl(comm, 0)) avg_renumeration from emp;

AVG(SAL)	AVG_RENUMERATION
2073.21429	2230.35714

11. Find maximum, minimum and average salaries in each department.

SQL> select deptno, max(sal), min(sal), avg(sal) from emp group by deptno;

DEPTNO	MAX(SAL)	MIN(SAL)	AVG(SAL)
10	5000	1300	2916.66667
20	3000	800	2175
30	2850	950	1566.66667

12. Find the maximum, minimum and average salaries of each job.

SQL> select job, max(sal), min(sal) , avg(sal) from emp group by job;

JOB	MAX(SAL)	MIN(SAL)	AVG(SAL)
ANALYST	3000	3000	3000
CLERK	1300	800	1037.5
MANAGER	2975	2450	2758.33333
PRESIDENT	5000	5000	5000
SALESMAN	1600	1250	1400

13. Find the departments which have more than 3 employees.

SQL> select deptno from emp group by deptno having count(empno)>3;

DEPTNO
20
30

14. Display employee names and their respective department numbers.

SQL> select ename name, deptno department from emp;

NAME	DEPARTMENT
SMITH	20
ALLEN	30
WARD	30
JONES	20
MARTIN	30
BLAKE	30
CLARK	10
SCOTT	20
KING	10
TURNER	30
ADAMS	20
JAMES	30
FORD	20
MILLER	10

14 rows selected.

15. Give the salary grades for all the employees.

```
SQL> select ename , grade from emp e, salgrade s
  2  where e.sal between s.losal and s.hisal;
```

ENAME	GRADE
SMITH	1
ADAMS	1
JAMES	1
WARD	2
MARTIN	2
MILLER	2
ALLEN	3
TURNER	3
JONES	4
BLAKE	4
CLARK	4
SCOTT	4
FORD	4
KING	5

14 ROWS SELECTED.

16. Display the employee names who earn highest salary in each job.

```
SQL> select ename, job from emp where sal in
  2  (select max(sal) from emp group by job);

ENAME      JOB
---------- ---------
MILLER     CLERK

ALLEN      SALESMAN

JONES      MANAGER

SCOTT      ANALYST

FORD       ANALYST

KING       PRESIDENT

6 rows selected.
```

17. Find the employee details whose salary is greater than Blake's salary.

```
SQL> select * from emp where sal>(select sal from emp where ename='BLAKE');

EMPNO ENAME  JOB       MGR HIREDATE  SAL  COMM DEPTNO
----- ------ --------- --- --------- ---- ---- ------
 7566 JONES  MANAGER  7839 02-APR-81 2975  500     20

 7698 BLAKE  MANAGER  7839 01-MAY-81 2850          30

 7788 SCOTT  ANALYST  7566 19-APR-87 3000    0     20

 7839 KING   PRESIDENT     17-NOV-81 5000          10

 7902 FORD   ANALYST  7566 03-DEC-81 3000          20

4 rows selected.
```

18. Find employee details of employees who have the same job and salary as that of Scott.

```
SQL> select * from emp where
  2 (sal=(select sal from emp where ename='SCOTT') and
  3  job=(select job from emp where ename='SCOTT'));

     EMPNO ENAME    JOB           MGR HIREDATE       SAL COMM
     DEPTNO
     ---------- ---------- --------- ---------- --------- ---------- ----------
     ----------

     7788 SCOTT      ANALYST   7566      19-APR-87       3000
     20

     7902 FORD       ANALYST   7566     03-DEC-81       3000
     20
```

19. Display the maximum salaries in accounting and research department.

```
SQL> select max(sal) from emp e, dept d
  2  where e.deptno=d.deptno and  d.dname in('ACCOUNTING','RESEARCH');

    MAX(SAL)
    ----------
       5000
```

20. Display salary grades of all employees except of those employees whose salary grade is 3 and 4.

```
SQL> select ename, grade from emp e, salgrade s where
  2  e.sal between s.losal and s.hisal
  3  and grade not in(3,4);

     ENAME      GRADE
     ---------- ----------
     SMITH           1
     ADAMS           1
     JAMES           1
     WARD            2
     MARTIN          2
     MILLER          2
     KING            5

7 rows selected.
```

21. Give the names and salaries of the employees whose salary is maximum in their respective departments.

```
SQL> select ename, deptno from emp where sal in
```

2 (select max(sal) from emp group by deptno);

NAME	DEPARTMENT
BLAKE	30
SCOTT	20
FORD	20
KING	10

22. List the employees whose salary is greater than the salaries of all employees who are working as salesmen.

```
SQL> select ename from emp

  2 where sal> all (select sal from emp where job='SALESMAN');

ENAME
-------
JONES
BLAKE
CLARK
SCOTT
KING
FORD

6 rows selected.
```

23. Write a query which will return the day of the week entered in the format of sysdate.

```
SQL> select to_char(sysdate, 'day') day from dual;
DAY
------
  Sunday
```

24. Find the difference between highest and lowest salaries.

```
SQL> select max(sal) - min(sal) difference from emp;
DIFFERENCE
--------------
    4200
```

25. Generate the output as Smith – clerk.

```
SQL>  select ename || ' - ' || job from emp where ename=’smith’;
```

```
ENAME||'-'||JOB
--------------------
SMITH - CLERK
```

26. Generate the output as Smith (Clerk).

```
SQL> select ename || ' ( ' || job || ' ) ' emp_job_details from emp;

EMP_JOB_DETAILS
----------------
SMITH ( CLERK )
```

27. Give the details of all employees those who are working as manager.

```
SQL> select * from emp where job = 'MANAGER';

 EMPNO ENAME      JOB              MGR HIREDATE        SAL       COMM     DEPTNO
------ ---------- --------- ---------- --------- ---------- ---------- ----------
  7566 JONES      MANAGER         7839 02-APR-81       2975        500         20
  7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
  7782 CLARK      MANAGER         7839 09-JUN-81       2450                    10
```

28. List the departments where there are no employees

```
SQL> select deptno, dname from dept
  2  where deptno not in (select deptno from emp group by deptno);

    DEPTNO DNAME
---------- --------------
        40 OPERATIONS
```

29. Generate the following list :
EMPLOYEE NAME JOB SAL GRADE

```
SQL> select ename, job, sal, grade from emp e, salgrade s
```

2 where e.sal between s.losal and s.hisal;

ENAME	JOB	SAL	GRADE
SMITH	CLERK	800	1
ADAMS	CLERK	1100	1
JAMES	CLERK	950	1
WARD	SALESMAN	1250	2
MARTIN	SALESMAN	1250	2
MILLER	CLERK	1300	2
ALLEN	SALESMAN	1600	3
TURNER	SALESMAN	1500	3
JONES	MANAGER	2975	4
BLAKE	MANAGER	2850	4
CLARK	MANAGER	2450	4
SCOTT	ANALYST	3000	4
FORD	ANALYST	3000	4
KING	PRESIDENT	5000	5

14 rows selected.

30. List the information of those employees in department number 10.

SQL> select * from emp

2 where deptno=10 and job not in('MANAGER', 'CLERK');

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT		17-NOV-81	5000		10

32. Find the department location of James.

SQL> select loc from dept d, emp e where

2 e.deptno=d.deptno and e.ename= 'JONES';

LOC

DALLAS

33. Get the manager of Jones.

SQL> select ename manager_of_Jones from emp where empno in
2 (select mgr from emp where ename = ‘JONES’);

MANAGER_OF_JONES

KING

34. Get the sub-ordinates of Jones.

SQL> select ename from emp where

2 mgr in(select empno from emp where ename='JONES');

ENAME

SCOTT
FORD

35. Write a query to calculate the length of time of all employees with the company.

SQL> select ename, trunc(months_between(sysdate, hiredate)) length_of_time from emp;

ENAME LENGTH_OF_TIME

SMITH	290
ALLEN	288
WARD	288
JONES	286
MARTIN	280
BLAKE	285
CLARK	284
SCOTT	214
KING	279
TURNER	281
ADAMS	213
JAMES	278
FORD	278
MILLER	277

14 rows selected.

36. List out name, job, salary, grade, department name of all employees who are not clerks.

```
SQL> select ename, job, sal, grade, dname from emp e, dept d, salgrade s
  2  where e.deptno=d.deptno and
  3  e.sal between s.losal and s.hisal and
  4  job not in('CLERK');
```

ENAME	JOB	SAL	GRADE	DNAME
WARD	SALESMAN	1250	2	SALES
MARTIN	SALESMAN	1250	2	SALES
ALLEN	SALESMAN	1600	3	SALES
TURNER	SALESMAN	1500	3	SALES
JONES	MANAGER	2975	4	RESEARCH
BLAKE	MANAGER	2850	4	SALES
CLARK	MANAGER	2450	4	ACCOUNTING
SCOTT	ANALYST	3000	4	RESEARCH
FORD	ANALYST	3000	4	RESEARCH
KING	PRESIDENT	5000	5	ACCOUNTING

10 rows selected.

37. List out name, job, salary, grade, department name of all employees who are clerks.

```
SQL> select ename, job, sal, grade, dname from emp e, dept d, salgrade s
  2  where e.deptno=d.deptno and
  3  e.sal between s.losal and s.hisal and
  4  job in('CLERK');

ENAME      JOB              SAL      GRADE          DNAME
----------  ---------- ----------  ----------  --------------
SMITH      CLERK            800           1           RESEARCH
ADAMS      CLERK           1100           1           RESEARCH
JAMES      CLERK            950           1           SALES
MILLER     CLERK           1300           2           ACCOUNTING
```

38. Display the most recently employed employee in each department.

```
SQL>  select ename, hiredate, deptno from emp
  2  where (deptno, hiredate) in (select deptno, max(hiredate) from emp
  3 group by deptno);

ENAME      HIREDATE      DEPTNO
---------- ---------- ----------
MILLER      23-JAN-82           10
ADAMS       23-MAY-87           20
JAMES       03-DEC-81           30
```

39. Display each employee with name, hiredate and review date – 1 year from now.

SQL> select ename, hiredate, add_months(sysdate,12) revdate from emp;

ENAME	HIREDATE	REVDATE
SMITH	17-DEC-80	27-FEB-06
ALLEN	20-FEB-81	27-FEB-06
WARD	22-FEB-81	27-FEB-06
JONES	02-APR-81	27-FEB-06
MARTIN	28-SEP-81	27-FEB-06
BLAKE	01-MAY-81	27-FEB-06
CLARK	09-JUN-81	27-FEB-06
SCOTT	19-APR-87	27-FEB-06
KING	17-NOV-81	27-FEB-06
TURNER	08-SEP-81	27-FEB-06
ADAMS	23-MAY-87	27-FEB-06
JAMES	03-DEC-81	27-FEB-06
FORD	03-DEC-81	27-FEB-06
MILLER	23-JAN-82	27-FEB-06

14 rows selected.

40. Display the year in which most people joined.

```
SQL> select to_char(hiredate,'yyyy') year from emp
  2  group by to_char(hiredate,'yy') having
  3  count(to_char(hiredate,'yy'))= (select
max(count(to_char(hiredate,'yy')))
  4  from emp group by to_char(hiredate,'yy'));
```

YEAR

1981

41. List out minimum salary of those employees under manager.

SQL> select ename, mgr, sal from emp

2 where sal in(select min(sal) from emp group by mgr having count(mgr)>=1);

ENAME	MGR	SAL
SMITH	7902	800
JAMES	7698	950
ADAMS	7788	1100
MILLER	7782	1300
CLARK	7839	2450
SCOTT	7566	3000
FORD	7566	3000

7 rows selected.

42. Print the employee names and List salary as:

a. ‘above target’ if salary is greater than 1500
b. ‘below target‘ if salary is lesser than 1500
c. ‘on the target‘ if salary is exactly 1500.

SQL> select ename, decode(sign(sal-1500),0,'on the target',

2 -1,'below target',

3 1,'above target')

4 details from emp;

ENAME DETAILS

```
--------------------
SMITH    below    target
ALLEN    above    target
WARD     below    target
JONES    above    target
MARTIN   below    target
BLAKE    above    target
CLARK    above    target
SCOTT    above    target
KING     above    target
TURNER   on the   target
ADAMS    below    target
JAMES    below    target
FORD     above    target
MILLER   below    target
```

14 rows selected.

43. Display ename, loc and dnames of the employees who earn more than 1500 salary.

```
SQL> select ename, loc, dname from emp e, dept d

  2  where e.deptno = d.deptno and e.sal>1500;

ENAME      LOC              DNAME
---------- ---------------- --------------

ALLEN      CHICAGO          SALES

JONES      DALLAS           RESEARCH

BLAKE      CHICAGO          SALES

CLARK      NEW YORK         ACCOUNTING

SCOTT      DALLAS           RESEARCH

KING       NEW YORK         ACCOUNTING

FORD       DALLAS           RESEARCH

7 rows selected.
```

44. Who are the top 3 earners in the company.

```
SQL> select e.ename, e.sal from emp e where

  2    3 > (select count(ename) from emp where emp.sal>e.sal);

ENAME        SAL
---------- ---------
SCOTT         3000
KING          5000
   FORD          3000
```

45. Write a query to display a '*' against the employee who joined most recently.

```
SQL> (select empno, ename||' * ' name , job , mgr, hiredate, deptno from emp

  2   where hiredate in (select max(hiredate) from emp)) union

  3  (select empno, ename, job, mgr, hiredate,  deptno from emp);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	-- 7902	17-DEC-80	-800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250		30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850	1400	30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN		08-SEP-81	1500	0	30
7876	ADAMS *	CLERK	7698	23-MAY-87	1100		20
7900	JAMES	CLERK	7788	03-DEC-81	950		30
7902	FORD	ANALYST	7698	03-DEC-81	3000		20
7934	MILLER	CLERK	7566 7782	23-JAN-82	1300		10

14 rows selected.

46. Find the most recently joined employee in the company in each department in order of hiredate.

```
SQL> select e.ename from emp e, emp m where

  2  e.mgr=m.empno and

  3  m.hiredate>e.hiredate;

ENAME
-------
SMITH

ALLEN

WARD

JONES

BLAKE

CLARK

6 rows selected.
```

47. Find all the employees who joined before their manager.

```
SQL> select e.ename from emp e, emp m

  2  where e.mgr= m.empno and m.hiredate > e.hiredate;

ENAME
-------
SMITH

ALLEN

WARD

JONES

BLAKE

CLARK

6 rows selected.
```

48. List jobs and department names of employees whose names are 5 letters long and must begin with ‘ALL’ and end with ‘N’.

```
SQL> select ename, job, deptno from emp where
  2  length(ename)=5 and
  3  ename like('ALL%N');

ENAME      JOB              DEPTNO
---------- ---------- ---------
ALLEN      SALESMAN         30
```

49. List employees whose commission is greater than 25% of their salary.

```
SQL> select ename from emp where nvl(comm,0)>(0.25 * sal);

ENAME
-------
WARD
MARTIN
```

50. List all the departments with at least 2 clerks.

```
SQL> select e.deptno from emp e,dept d where
  2  e.deptno = d.deptno and
  3  e.job='CLERK'
  4  group by e.deptno having count(e.job) >=2;

    DEPTNO
---------
      20
```

51. List employees in dept 30 with job not found in dept 10.

```
SQL> select ename from emp
  2  where deptno = 30 and
  3  job not in (select distinct(job) from emp where deptno= 10);

ENAME
-------
ALLEN
WARD
MARTIN
TURNER
```

52. Who earns the second highest salary?

```
SQL> select ename from emp where
  2 sal = (select max(sal) from emp where sal not in (select max(sal) from
emp));

ENAME
-------
SCOTT
FORD
```

53. Display the employee hiredate in the format of “Friday, 10-AUG-03”.

```
SQL> select to_char(hiredate, 'day, dd-mon-yy') hiredate from emp;

HIREDATE
-------------
Wednesday, 17-dec-80

Friday , 20-feb-81

Sunday , 22-feb-81

Thursday , 02-apr-81

Monday , 28-sep-81

Friday , 01-may-81

Tuesday , 09-jun-81

Sunday , 19-apr-87

Tuesday , 17-nov-81

Tuesday , 08-sep-81

Saturday , 23-may-87

Thursday , 03-dec-81

Thursday , 03-dec-81

Saturday , 23-jan-82

14 rows selected.
```

54. Count the people in department 30 who receive both salary and commission.

SQL> select count(empno) from emp where nvl(comm,0)!=0 and sal!=0 and deptno=30;

```
COUNT(EMPNO)
----------------
           3
```

55. List all the employees who were hired between " 1st January 1981" and "15th April 1985". In the order of their seniority.

SQL> select ename from emp where

2 hiredate between '01-JAN-1981' and '15-APR-1985' order by hiredate;

```
ENAME
-------
ALLEN

WARD

JONES

BLAKE

CLARK

TURNER

MARTIN

KING

JAMES

FORD

MILLER

11 rows selected.
```

56. Display name and period of service till date.

SQL> select ename, trunc(months_between(sysdate,hiredate))

```
2  || ' MONTHS '  service_period from emp order by hiredate;

ENAME      SERVICE_PERIOD
------------------------------
SMITH                290 MONTHS

ALLEN                288 MONTHS

WARD                 288 MONTHS

JONES                286 MONTHS

BLAKE                285 MONTHS

CLARK                284 MONTHS

TURNER               281 MONTHS

MARTIN               281 MONTHS

KING                 279 MONTHS

JAMES                278 MONTHS

FORD                 278 MONTHS

MILLER               277 MONTHS

SCOTT                214 MONTHS

ADAMS                213 MONTHS

14 rows selected.
```

PL/SQL PROGRAMS

1. Program to add and multiply 2 numbers.

SQL> ed add.sql;

```
declare
        a number :=&a;
        b number :=&b;
begin
        dbms_output.put_line(' Sum of ' || a||' and '|| b || ' is '||(a+b));
        dbms_output.put_line('Product of ' || a || 'and' || b || 'is' || a*b);
end;
```

SQL> @add.sql;

Input :-

```
Enter the value of a : 4
Enter the value of b : 5
```

Output :-

```
Sum of 4 and 5 is 9
Product of 4 and 5 is 20
```

PL/SQL procedure successfully completed

2. Program to find sum of first 'n' natural no.s.

SQL> ed sum.sql;

```
declare
        n number := &n;
        y number :=0;
begin
        for j in 1..n
        loop
                y := y + j;
        end loop;
        dbms_output.put_line(' Sum of first ' ||n|| ' Natural no.s is '|| y);
end;
```

SQL> @ sum.sql;

Input:-

Enter the value of n : 5

Output:-

Sum of first 5 natural no.s is 15.

PL/SQL procedure successfully completed

3. Program to print 10 numbers in the reverse order.

SQL> ed rev.sql;

```
begin
        for i in reverse 1..10
        loop
                dbms_output.put_line(i);
        end loop;
end;
```

SQL> @rev.sql

Output:-

```
10
9
8
7
6
5
4
3
2
1
PL/SQL procedure successfully completed
```

4. Program to print even numbers upto 100.

SQL> ed even.sql;

```
declare
        n number :=&n;
        c number;
begin
        for j in 1..n
        loop
                if j mod 2 =0 then
                        dbms_output.put_line(j);
                end if;
        end loop;
end;
```

SQL> @even.sql;

Input :-

Enter the value of n: 100

Output :-

2	22	42	62	82
4	24	44	64	84
6	26	46	66	86
8	28	48	68	88
10	30	50	70	90
12	32	52	72	92
14	34	54	74	94
16	36	56	76	96
18	38	58	78	98
20	40	60	80	100

PL/SQL procedure successfully completed

5. Program to find sum of digits of a number.

SQL> ed digit.sql;

```
declare
        n number := &n;
        s number := 0;
begin
        while n>0
        loop
                s := n mod 10 + s;
                n := floor(n/10);
        end loop;
        dbms_output.put_line('The Sum of digits of  given number is '||s);
end;
```

SQL> @digit.sql;

Input :-

Enter the value of n : 111

Output :-

The sum of digits of given number is 3.

PL/SQL procedure successfully completed

6. Program to print multiplication table of a given number.

SQL> ed multi.sql;

```
declare
        n number := &n;
        product number;

begin
        for j in 1..10
        loop
                product := n*j;
                dbms_output.put_line( n || ' * ' || j || ' = ' || product);
        end loop;
end;
```

SQL> @multi.sql;

Input :-

Enter the value of n: 5

Output :-

```
5 * 1  =  5
5 * 2  =  10
5 * 3  =  15
5 * 4  =  20
5 * 5  =  25
5 * 6  =  30
5 * 7  =  35
5 * 8  =  40
5 * 9  =  45
5 * 10 = 50
```

PL/SQL procedure successfully completed

7. Program to print n multiplication tables.

SQL> ed nmulti.sql;

```
declare
        n number := &n;
        product number;

begin
        for i in 1..n
        loop
                for j in 1..10
                loop
                        product := i * j;
                        dbms_output.put_line( i || ' * ' || j || ' = ' || product);
                end loop;
        end loop;
end;
```

SQL> @nmulti.sql;

Input :-

Enter the value of n : 5

Output:-

1 * 1 = 1	2 * 1 = 2	3 * 1 = 3	4 * 1 = 4	5 * 1 = 5
1 * 2 = 2	2 * 2 = 4	3 * 2 = 6	4 * 2 = 8	5 * 2 = 10
1 * 3 = 3	2 * 3 = 6	3 * 3 = 9	4 * 3 = 12	5 * 3 = 15
1 * 4 = 4	2 * 4 = 8	3 * 4 = 12	4 * 4 = 16	5 * 4 = 20
1 * 5 = 5	2 * 5 = 10	3 * 5 = 15	4 * 5 = 20	5 * 5 = 25
1 * 6 = 6	2 * 6 = 12	3 * 6 = 18	4 * 6 = 24	5 * 6 = 30
1 * 7 = 7	2 * 7 = 14	3 * 7 = 21	4 * 7 = 28	5 * 7 = 35
1 * 8 = 8	2 * 8 = 16	3 * 8 = 24	4 * 8 = 32	5 * 8 = 40
1 * 9 = 9	2 * 9 = 18	3 * 9 = 27	4 * 9 = 36	5 * 9 = 45
1 * 10 = 10	2 * 10 = 20	3 * 10 = 30	4 * 10 = 40	5 * 10 = 50

PL/SQL procedure successfully completed

8. Program to find reverse of a number.

SQL> ed reverse.sql;

```
declare
        n number := &n;
        r number;
        s number :=0;

begin
        while (n != 0)
        loop
                r  := mod(n , 10);
                s := (s*10) + r;
                n := floor(n/10);
        end loop;
        dbms_output.put_line( ' The reverse of given number is '|| s);
end;
```

SQL> @reverse.sql;

Input :-

Enter the value of n : 1234

Output :-

The reverse of given number is 4321.

PL/SQL procedure successfully completed

9. Program to check whether a given number is prime or not.

SQL> ed prime.sql;

```
declare
        a number := &a;
        r number := 0;
begin
        for i in 2..a-1
        loop
                if a mod i = 0  then
                        r:=1;
                end if;
        end loop;
        if r = 0 then
                dbms_output.put_line(' It is a prime number ' );
        else
                dbms_output.put_line(' It is not a prime number ' );
        end if;
end;
```

SQL> @prime.sql;

Input :-

Enter the value of n : 13

Output :-

It is a prime number.

PL/SQL procedure successfully completed

10. Program to generate Fibonacci series.

SQL> ed fibonacci.sql ;

```
declare
        a number :=0;
        b number :=1;
        c number;
begin
        while b<=1000
        loop
                c :=a+b;
        dbms_output.put_line(b);
        dbms_output.put_line(c);
                a:=c;
                b := b+c;
        end loop;
end;
```

SQL> @fibonacci.sql;

Output :-

```
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
```

PL/SQL procedure successfully completed

11. Program to generate prime numbers upto 100.

```
SQL> ed prime.sql;

declare
        flag number :=1;
        i number;
begin
        for j in 2..100
        loop
                flag:=1;
                i :=2;
                while i<=(j/2)
                loop
                        if j mod i =0 then
                                flag :=0;
                        end if;
                        i:=i+1;
                end loop;
                if flag =1 then
                        dbms_output.put_line(j);
                end if;
        end loop;
end;

SQL> @ prime.sql;
```

Output:-

```
1               23              59
2               29              61
3               31              67
5               37              71
7               41              73
11              43              79
13              47              83
17              53              89
19                              97
PL/SQL procedure successfully completed
```

12. Program to find factorial of a given number.

SQL> ed factorial.sql;

```
declare
        n number := &n;
        p number :=1;
begin
        for j in 1..n
        loop
                p := p * j;
        end loop;
        dbms_output.put_line( ' Factorial of '|| n || ' is ' || p);
end;
```

SQL> @ factorial.sql;

Input:-

Enter the value of n : 5

Output:-

Factorial of 5 is 120.

PL/SQL procedure successfully completed

13. Program to find the area of a circle given the radius.

SQL> ed acircle.sql;

```
declare
        r number := &radius;
        area number;
begin
        area := 3.141 * r*r;
        dbms_output.put_line(area);
end;
```

SQL> @ acircle.sql;

Input :-

Enter the value of radius : 5

Output:-

78.525

PL/SQL procedure successfully completed

14. Insert the values of areas of a circle into a table called areas taking radius values from 2 to 8.

```
SQL> create table areas ( radius number, area number);

        Table created.

SQL> ed areas.sql

        declare
                area number;
        begin
                for radius in 2..8
                loop
                        area := 3.14* radius*radius;
                        insert into areas values(radius,area);
                end loop;
        end;


SQL> @ areas.sql;

        PL/SQL procedure successfully completed.

SQL> select * from areas;

    RADIUS       AREA
----------      ----------
         2         12.56
         3         28.26
         4         50.24
         5         78.5
         6         113.04
         7         153.86
         8          200.96

        7 rows selected.
```

15. Program to find whether the given number is an Armstrong number or not.

SQL> ed armstrong.sql;

```
declare
        a number := &a;
        s number :=0;
        n number :=a;
        re number;

begin
        while a>0
        loop
                re := a mod 10;
                re := re*re*re;
                s := s+re;
                a := floor(a/10);
        end loop;
        if n = s then
                dbms_output.put_line(' It is  an armstrong number ');
        else
                dbms_output.put_line(' It is not an armstrong number ');

        end if;
end;
```

SQL> @ armstrong.sql;

Input :-

Enter the value of a : 153

Output :-

It is an Armstrong number.

PL/SQL procedure successfully completed.

16. Program to generate Armstrong numbers below 1000.

SQL> ed armstrongn.sql;

```
declare
        s number :=0;
        a number ;
        re number;

begin
        for n in 1..1000
        loop
                s:=0;
                a :=n;
                while a>0
                loop
                        re := a mod 10;
                        re := re*re*re;
                        s := s+re;
                        a := floor(a/10);
                end loop;
                if s = n then
                        dbms_output.put_line(n);
                end if;
        end loop;
end;
```

SQL> @armstrongn.sql;

Output:-

```
 1
153
370
371
407
```

PL/SQL procedure successfully completed.

17. Program to find greatest of 2 numbers.

SQL> ed greatest.sql;

```
declare
        a number := &a;
        b number := &b;
begin
        if a>b then
                dbms_output.put_line(a || ' is greater than ' || b);

        else
                dbms_output.put_line(b || ' is greater than ' || a);
        end if;
end;
```

SQL> @ greatest.sql;

Input:-

Enter the value of a: 5
Enter the value of b: 15

Output:-

15 is greater than 5

PL/SQL procedure successfully completed.

18. Program to check whether the given number is palindrome number or not.

SQL> ed palindrome.sql;

```
declare
        n number := &n;

       a number := n ;
        r number;
        s number :=0;

begin
        while (n != 0)
        loop
                r  := mod(n , 10);
                s := (s*10) + r;
                n := floor(n/10);
        end loop;
        if(s=a) then dbms_output.put_line( ' It is a palindrome number');
        else dbms_output.put_line(' It is not a palindrome number');
        end if;
end;
```

SQL> @palindrome.sql;

Input :- Enter the value of n : 121

Output :- It is a palindrome number.

PL/SQL Procedure successfully completed.

19. Program to find the greatest of 3 numbers.

SQL> ed great.sql;

```
declare
        a number :=&a;
        b number :=&b;
        c number :=&c;
begin
          if a>b then
                if(a>c) then dbms_output.put_line(a||' is the greatest of the
    given numbers. ' );
                else  dbms_output.put_line(c||' is the greatest of the given
    numbers. ' );
                end if;
          else
                if(c>b) then  dbms_output.put_line(c||' is the greatest of the
      given numbers. ' );
                else  dbms_output.put_line(b||' is the greatest of the given
      numbers. ' );
                end if;
        end if;
end;
```

SQL> @ great.sql;

Input :-

```
Enter the value of a: 5
Enter the value of b: 7
Enter the value of  c:  4
```

Output :-

7 is the greatest of given numbers.

PL/SQL Procedure successfully completed.

20. Program to check whether the given number is Adams number or not!

```
SQL> ed adams.sql;

declare
        a number:=&a;
        re1 number;
        re2 number;
        n number ;
        s1 number :=0;
        s2 number :=0;
begin
        n:=a;
        n:=n*n;
        while (n!=0)
        loop
                re1 :=n mod 10;
                s1 := re1 + (10*s1);
                n := floor(n/10);
        end loop;
        s1 := sqrt(s1);
        while  (s1!=0)
        loop
                re2 :=s1 mod 10;
                s2 := re2 + (10*s2);
                s1 := floor(s1/10);
        end loop;
        if s2=a then dbms_output.put_line(' It is ADAMS number ' );
        else dbms_output.put_line(' It is not an ADAMS number ');
        end if;
end;

SQL> @adams.sql;
```

Input :- Enter the value of a: 12

Output :- It is an adams number.

PL/SQL Procedure successfully completed.

CURSOR'S PROGRAMS:

1. Write a cursor program to retrieve the details of all the employees using cursors.

```
SQL> ed cursor1.sql;
declare
        cursor cemp is select empno, ename, job, sal  from emp;
        vemp cemp%rowtype;
begin
        open cemp;
        dbms_output.put_line(' Details of the employees :- ');
        loop
                fetch cemp into vemp ;
                exit when (cemp% notfound);
                dbms_output.put_line(vempno.empno || '   ' || vempno.ename || '   ' ||
                                                        vemp.job || '  ' || vempno.sal);
        end loop;
        close cemp;
end;
```

```
SQL> @cursor1.sql;
```

7369	SMITH	CLERK	800
7499	ALLEN	SALESMAN	1600
7521	WARD	SALESMAN	1250
7566	JONES	MANAGER	2975
7654	MARTIN	SALESMAN	1250
7698	BLAKE	MANAGER	2850
7782	CLARK	MANAGER	2450
7788	SCOTT	ANALYST	3000
7839	KING	PRESIDENT	5000
7844	TURNER	SALESMAN	1500
7876	ADAMS	CLERK	1100
7900	JAMES	CLERK	950
7902	FORD	ANALYST	3000
7934	MILLER	CLERK	1300

PL/SQL Procedure successfully completed.

2. Write a PL/SQL block to find the employees in department 20.

SQL> ed cursor2.sql;

```
declare
        cursor cemp2 is select ename from emp where deptno=20;
        vemp cemp2%rowtype;
begin
        open cemp2;
        dbms_output.put_line('Employees in department 20:-');
        loop
                fetch cemp2 into vemp;
                exit when(cemp2% notfound);
                dbms_output.put_line(vemp.ename);
        end loop;
        close cemp2;
end;
```

SQL> @cursor2.sql;

Output :-

Employees in department 20:-

SMITH
JONES
SCOTT
ADAMS
FORD

PL/SQL procedure successfully completed.

3. Write a PL/SQL block to update the salary of each employee in department 20 by raisin the salary by 20% and insert a record in empraise table.

SQL> create table empraise(empno number(5), raisedate date, raiseamt number(5));

Table created.

SQL> ed cursor3.sql;

```
declare
        cursor cempr is select empno, sal from emp where deptno=20;
        vemp cempr%rowtype;
begin
        open cempr;
        loop

                fetch cempr into vemp;
                exit when cempr% notfound;
                update emp set sal = vemp.sal+(vemp.sal*.20) where
empno=vemp.empno;
                insert into empraise values(vemp.empno, sysdate, vemp.sal*0.20);
        end loop;
        commit;
        close cempr;
end;
```

SQL> @cursor3.sql;

PL/SQL procedure successfully completed.

SQL> select * from empraise;

EMPNO	RAISEDATE	RAISEAMT
7369	02-APR-05	160
7566	02-APR-05	595
7788	02-APR-05	600
7876	02-APR-05	220
7902	02-APR-05	600

4. Write a cursor program to count the number of employees in department 20.

SQL> ed cursor4.sql;

```
declare
        c number(5);
        cursor cemp3 is select count(empno) c from emp where deptno=20;
        vemp cemp3%rowtype;
begin
        open cemp3;
        dbms_output.put_line('Employees in department 20:-');
        loop
                fetch cemp3 into vemp;
                exit when(cemp3% notfound);
                dbms_output.put_line(vemp.c);
        end loop;
        close cemp3;
end;
```

SQL> @cursor4.sql;

Output :-

```
Employees in department 20:-
5
```

PL/SQL procedure successfully completed.

5. Write a PL/SQL block to count the no. of employees in each job type.

SQL> ed cursor5.sql;

```
declare
        c number(5);
        cursor cemp3 is select count(empno) c,job from emp group by job;
        vemp cemp3%rowtype;
begin
        open cemp3;
        loop
                fetch cemp3 into vemp;
                exit when(cemp3% notfound);
                dbms_output.put_line(vemp.job||'  - ' ||vemp.c);
        end loop;
        close cemp3;
end;
```

SQL> @cursor5.sql;

Output :-

```
ANALYST  - 2
CLERK  - 4
MANAGER  - 3
```

PRESIDENT - 1
SALESMAN - 4

PL/SQL procedure successfully completed.

TRIGGERS's PROGRAMS :

1. Trigger on Insertion.

```
SQL> create table log(access_date date,operation varchar2(40), table
varchar(10));

Table created.

SQL> create or replace trigger trigger2  after insert on class
  2  begin
  3        insert into log values(sysdate,' 1 row successfully inserted ',' CLASS ');
  4        dbms_output.put_line('CONGRATULATIONS!!!');
  5  end;
  6  /

Trigger created.

SQL> insert into class values(1260,'Vikram');
```

Output :-

CONGRATULATIONS!!!

1 row created.

SQL> select * from log;

ACCESS_DATE	OPERATION	TABLE
02 – APR – 2005	1 row successfully inserted	CLASS

2. Trigger on updation.

```
SQL> create  or replace trigger  trigger3  after update on class
  2   begin
  3         insert into log values(sysdate,' 1 row successfully  updated  ',' CLASS ');
  4         dbms_output.put_line('Success!!!');
  5   end;
  6  /
```

Trigger created.

SQL> update class set name=’Uday’ where roll=1260;

Output :-

Success!!!

1 row updated.

SQL> select * from log;

ACCESS_DATE	OPERATION	TABLE
02 – APR – 2005	1 row successfully updated	CLASS
02 – APR – 2005	1 row successfully inserted	CLASS

3. Trigger on updation (Statement Level Trigger).

```
SQL> create or replace trigger trigger3 after update on class
  2  begin
  3  dbms_output.put_line ('Success!!!');
  4   insert into log values(sysdate, 'One row successfully updated ',' CLASS ');
  5  end;
  6  /

 Trigger created.

SQL> update class set grade='A' where marks>90;
```

Output :-

```
Success!!!

3 rows updated.

SQL> select * from log;

ACCES_DATE      OPERATION                    TABLENAME
```

02 – APR – 2005	One row successfully updated	CLASS
02 – APR – 2005	1 row successfully updated	CLASS
02 – APR – 2005	1 row successfully inserted	CLASS

4. Trigger on updation (Row Level Trigger).

```
SQL> create or replace trigger trigger4 after update on class for each row
  2  begin
  3  dbms_output.put_line ('Success!!!');
  4   insert into log values(sysdate, '1 row  Updated ',' CLASS ');
  5  end;
  6  /

 Trigger created.

SQL> update class set grade='A' where marks>90;
```

Output :-

```
Success!!!
Success!!!
Success!!!

3 rows updated.

SQL> select * from log;
```

ACCES_DATE	OPERATION	TABLENAME

02 – APR – 2005	1 row updated	CLASS
02 – APR – 2005	1 row updated	CLASS
02 – APR – 2005	1 row updated	CLASS
02 – APR – 2005	One row successfully updated	CLASS
02 – APR – 2005	1 row successfully updated	CLASS
02 – APR – 2005	1 row successfully inserted	CLASS

5. Trigger on Deletion.

```
SQL> create or replace trigger trigger4 after update on class
  2  begin
  3  dbms_output.put_line ('A row deleted!!!');
  4   insert into log values(sysdate, '1 row  successfully deleted ',' CLASS ');
  5  end;
  6  /

 Trigger created.

SQL> delete from class where roll=1201;
```

Output :-

A row deleted!!!

1 row deleted**.**

SQL> select * from log;

ACCES_DATE	OPERATION	TABLENAME

02 – APR – 2005	1 row successfully deleted	CLASS
02 – APR – 2005	1 row updated	CLASS
02 – APR – 2005	1 row updated	CLASS
02 – APR – 2005	1 row updated	CLASS
02 – APR – 2005	One row successfully updated	CLASS
02 – APR – 2005	1 row successfully updated	CLASS
02 – APR – 2005	1 row successfully inserted	CLASS