**DBMS PROGRAMS**

1. Get The Description Of Emp Table.

SQL>desc emp;

|  |  |  |
| --- | --- | --- |
| **Name**  | **Null?**  | **Type**  |
| EMPNO  | NOT NULL  | NUMBER(4)  |
| ENAME  |    | VARCHAR2(10)  |
| JOB  |    | VARCHAR2(9)  |
| MGR  |    | NUMBER(4)  |
| HIREDATE  |    | DATE  |
| SAL  |    | NUMBER(7,2)  |
| COMM  |    | NUMBER(7,2)  |
| DEPTNO  |    | NUMBER(2)  |

2.List All Employee Details.

SQL>select \* 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  | 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  |    | 20  |
| 7839  | KING  | PRESIDENT  |    | 17-NOV-81  | 5000  |    | 10  |
| 7844  | TURNER  | SALESMAN  | 7698  | 08-SEP-81  | 1500  | 0  | 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  |

14 rows selected.

3. List All Jobs Available In Employee Table.

SQL>select distinct job from emp;

|  |
| --- |
| **JOB**  |
| ANALYST  |
| CLERK  |
| MANAGER  |
| PRESIDENT  |
| SALESMAN  |

5. List All Employees Who Belongs To The Department 10 Or 20.

INPUT SQL>select ename from emp where deptno in (10,20);

|  |
| --- |
| **ENAME**  |
| SMITH  |
| JONES  |
| CLARK  |
| SCOTT  |
| KING  |
| ADAMS  |
| FORD  |
| MILLER  |

8 rows selected.

6. List All Employee Names , Salary And 15% Rise In Salary.

INPUT SQL>select ename , sal , sal+0.15\* sal from emp;

|  |  |  |
| --- | --- | --- |
| **ENAME**  | **SAL**  | **SAL+0.15\*SAL**  |
| SMITH  | 800  | 920  |
| ALLEN  | 1600  | 1840  |
| WARD  | 1250  | 1437.5  |
| JONES  | 2975  | 3421.25  |
| MARTIN  | 1250  | 1437.5  |
| BLAKE  | 2850  | 3277.5  |
| CLARK  | 2450  | 2817.5  |
| SCOTT  | 3000  | 3450  |
| KING  | 5000  | 5750  |
| TURNER  | 1500  | 1725  |
| ADAMS  | 1100  | 1265  |
| JAMES  | 950  | 1092.5  |
| FORD  | 3000  | 3450  |
| MILLER  | 1300  | 1495  |

14 rows selected.

7. List Minimum , Maximum , Average Salaries Of Employee.

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

|  |  |  |
| --- | --- | --- |
| **MIN(SAL)**  | **MAX(SAL)**  | **AVG(SAL)**  |
| 800  | 5000  | 2073.21429  |

8. Find How Many Job Titles Are Available In Employee Table.

SQL>select count (distinct job) from emp;

|  |
| --- |
| **COUNT(DISTINCTJOB)**  |
| 5  |

9. What is the difference between maximum and minimum salaries of

employees in the organization?

INPUT SQL>select max(sal)-min(sal) from emp;

|  |
| --- |
| **MAX(SAL)-MIN(SAL)**  |
| 4200  |

10.List all employee names and their salaries, whose salary lies between

1500/- and 3500/- both inclusive.

INPUT

SQL>select ename from emp where sal between 1500 and 3500;

|  |
| --- |
| **ENAME**  |
| ALLEN  |
| JONES  |
| BLAKE  |
| CLARK  |
| SCOTT  |
| TURNER  |
| FORD  |

7 rows selected.

11. Find how much amount the company is spending towards salaries.

INPUT SQL>select sum (sal) from emp;

|  |
| --- |
| **SUM(SAL)**  |
| 29025  |

12. Display name of the dept. with deptno 20.

INPUT SQL>select ename from emp where deptno = 20;

|  |
| --- |
| **ENAME**  |
| SMITH  |
| JONES  |
| SCOTT  |
| ADAMS  |
| FORD  |

13. List ename whose commission is NULL.

INPUT SQL>select ename from emp where comm is null;

|  |
| --- |
| **ENAME**  |
| SMITH  |
| JONES  |
| BLAKE  |
| CLARK  |
| SCOTT  |
| KING  |
| ADAMS  |
| JAMES  |
| FORD  |
| MILLER  |

10 rows selected.

14. Find no.of dept in employee table.

INPUT SQL>select count (distinct ename) from emp;

|  |
| --- |
| **COUNT(DISTINCTENAME)**  |
| 14  |

15. List ename whose manager is not NULL.

INPUT SQL>select ename from emp where mgr is not null;

|  |
| --- |
| **ENAME**  |
| SMITH  |
| ALLEN  |
| WARD  |
| JONES  |
| MARTIN  |
| BLAKE  |
| CLARK  |
| SCOTT  |
| TURNER  |
| ADAMS  |
| JAMES  |
| FORD  |
| MILLER  |

13 rows selected.

16. Display lowest paid employee details under each manager.

INPUT SQL>select job,sum (sal) from emp group by job;

|  |  |
| --- | --- |
| **JOB**  | **SUM(SAL)**  |
| ANALYST  | 6000  |
| CLERK  | 4150  |
| MANAGER  | 8275  |
| PRESIDENT  | 5000  |
| SALESMAN  | 5600  |

17. Display lowest paid employee details under each manager.

INPUT SQL>select ename, sal from emp where sal in (select min(sal) from emp group by mgr);

|  |  |
| --- | --- |
| **ENAME**  | **SAL**  |
| SMITH  | 800  |
| JAMES  | 950  |
| ADAMS  | 1100  |
| MILLER  | 1300  |
| CLARK  | 2450  |
| SCOTT  | 3000  |
| FORD  | 3000  |
| KING  | 5000  |

8 rows selected.

18. Display number of employees working in each department and their

department name.

INPUT SQL> select dname, count (ename) from emp, dept where emp.deptno=dept.deptno group by dname;

|  |  |
| --- | --- |
| **DNAME**  | **COUNT(ENAME)**  |
| ACCOUNTING  | 3  |
| RESEARCH  | 5  |
| SALES  | 6  |

**CREATING TABLE FOR VARIOUS RELATIONS(IN SQL).**

1.SQL>create table students(no number(5),name varchar2(20),rno varchar2(20) primary key);

Table created.

2.SQL>desc students;

|  |  |  |
| --- | --- | --- |
| **Name**  | **Null?**  | **Type**  |
| NO  |    | NUMBER(5)  |
| NAME  |    | VARCHAR2(20)  |
| RNO  | NOT NULL  | VARCHAR2(20)  |

3. SQL>desc marks;

|  |  |  |
| --- | --- | --- |
| **Name**  | **Null?**  | **Type**  |
| RNO  |    | VARCHAR2(20)  |
| ADS  |    | NUMBER(3)  |
| MFCS  |    | NUMBER(3)  |
| CO  |    | NUMBER(3)  |
| DBMS  |    | NUMBER(3)  |
| OS  |    | NUMBER(3)  |
| JAVA  |    | NUMBER(3)  |
| TOTAL  |    | NUMBER(4)  |

4.SQL> select \* from students;

|  |  |  |
| --- | --- | --- |
| **NO**  | **NAME**  | **RNO**  |
| 1  | kalyan  | mtech14  |
| 2  | sunitha  | mtech12  |
| 3  | padma  | mtech05  |
| 4  | siva  | mtech01  |
| 5  | varma  | mtech02  |
| 6  | divya  | mtech03  |
| 7  | khan  | mtech04  |
| 8  | rajesh  | mtech06  |
| 8  | prasad  | mtech07  |
| 9  | sandhya  | mtech13  |
| 10  | supriya  | mtech15  |

11 rows selected.

5.SQL> update students set name='varma' where no=5;

1 row updated.

6.SQL> delete from students where name='siva';

1 row deleted.

7.SQL> select \* from students;

|  |  |  |
| --- | --- | --- |
| **NO**  | **NAME**  | **RNO**  |
| 1  | kalyan  | mtech14  |
| 2  | sunitha  | mtech12  |
| 3  | padma  | mtech05  |
| 5  | varma  | mtech02  |
| 6  | divya  | mtech03  |
| 7  | khan  | mtech04  |
| 8  | rajesh  | mtech06  |
| 8  | prasad  | mtech07  |
| 9  | sandhya  | mtech13  |
| 10  | supriya  | mtech15  |

10 rows selected.

**IMPLEMENTING SINGLE ROW FUNCTIONS**

**Single row functions:**

**1.Sqrt();**

**SQL>select sqrt(16) from dual;**

|  |
| --- |
| **SQRT(16)**  |
| 4  |

**2.power();**

**SQL>select power(3,2) from dual;**

|  |
| --- |
| **POWER(3,2)**  |
| 9  |

**3.Round();**

**SQL>select round(34.56732,2) from dual;**

|  |
| --- |
| **ROUND(34.56732,2)**  |
| 34.57  |

**4.cos();**

**SQL>Select cos(0) from dual;**

|  |
| --- |
| **COS(0)**  |
| 1  |

**5.floor();**

**SQL>select floor(34.7) from dual;**

|  |
| --- |
| **FLOOR(34.7)**  |
| 34  |

**6.ceil();**

**SQL>select ceil(34.3) from dual;**

|  |
| --- |
| **CEIL(34.3)**  |
| 35  |

**7.abs();**

**SQL>select abs(-7) from dual;**

|  |
| --- |
| **ABS(-7)**  |
| 7  |

**WRITING PL/SQL BLOCK FOR INSERTION INTO A TABLE.**

**1. TO WRITE A PL/SQL BLOCK TO CHECK WHETHER A GIVEN NUMBER IS EVEN OR ODD.**

**PROGRAM:**

**DECLARE**

**NUM NUMBER(5);**

**REM NUMBER(5);**

**BEGIN**

**NUM:=&NUM;**

**REM:=MOD(NUM,2);**

**IF (REM=0)THEN**

**DBMS\_OUTPUT.PUT\_LINE(NUM||' IS EVEN');**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE(NUM||' IS ODD');**

**END IF;**

**END;**

**OUTPUT:**

**ENTER VALUE FOR NUM: 3**

**OLD 6: NUM:=&NUM;
NEW 6: NUM:=3;
NUMBER 3 IS ODD**

**PL/SQL PROCEDURE SUCCESSFULLY COMPLETED.**

**2.TO WRITE A PL/SQL PROGRAM TO IMPLEMENT THE SCOPE OF THE VARIABLE**.

**PROGRAM:**

**DECLARE**

**X REAL:=205;**

**BEGIN**

**DBMS\_OUTPUT.PUT\_LINE('THE VALUE OF OUTER X:='||X);**

**DECLARE**

**X REAL:=405;**

**BEGIN**

**DBMS\_OUTPUT.PUT\_LINE('THE VALUE OF INNER X:='||X);**

**END;**

**DBMS\_OUTPUT.PUT\_LINE('THE VALUE OF OUTER X:='||X);**

**END;**

**OUTPUT:**

**THE VALUE OF OUTER X:=205**

**THE VALUE OF INNER X:=405**

**THE VALUE OF OUTER X:=205**

**3. WRITE PL/SQL BLOCK TO CHECK WHETHER GIVEN NUMBER IS ARMSTRONG OR NOT.**

**PROGRAM:**

**DECLARE**

**NUM NUMBER(5);**

**REM NUMBER(5);**

**S NUMBER(5):=0;**

**NUM1 NUMBER(5);**

**BEGIN**

**NUM:=&NUM;**

**NUM1:=NUM;**

**WHILE(NUM>0)**

**LOOP**

**REM:=MOD(NUM,10);**

**S:=S+POWER(REM,3);**

**NUM:=TRUNC(NUM/10);**

**END LOOP;**

**IF(S=NUM1)THEN**

**DBMS\_OUTPUT.PUT\_LINE(NUM1||'IS ARMSTRONG NUMBER');**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE(NUM1||'IS NOT ARMSTRONG NUMBER');**

**END IF;**

**END;**

**OUTPUT:**

**ENTER VALUE FOR NUM: 153**

**OLD 7: NUM:=&NUM;**

**NEW 7: NUM:=153;**

**153 IS ARMSTRONG NUMBER**

**PL/SQL PROCEDURE SUCCESSFULLY COMPLETED.**

**SQL> /**

**ENTER VALUE FOR NUM: 123**

**OLD 7: NUM:=&NUM;**

**NEW 7: NUM:=123;**

**123 IS NOT ARMSTRONG NUMBER**

**PL/SQL PROCEDURE SUCCESSFULLY COMPLETED.**

**4.TO WRITE A PL/SQL BLOCK TO CHECK THE GIVEN STRING IS PALINDROME OR NOT.**

**PROGRAM:**

**DECLARE**

**NAME1 VARCHAR2(20);**

**NAME2 VARCHAR2(20);**

**L NUMBER(5);**

**BEGIN**

**NAME1:='&NAME1';**

**L:=LENGTH(NAME1);**

**WHILE L>0 LOOP**

**NAME2:=NAME2||SUBSTR(NAME1,L,1);**

**L:=L-1;**

**END LOOP;**

**DBMS\_OUTPUT.PUT\_LINE('REVERSE OF STRING IS:'||NAME2);**

**IF(NAME1=NAME2) THEN**

**DBMS\_OUTPUT.PUT\_LINE(NAME1||'IS PALINDROME');**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE(NAME1||'IS NOT PALINDROME');**

**END IF;**

**END;**

**OUTPUT:**

**ENTER VALUE FOR NAME1: MADAM**

**OLD 6: NAME1:='&NAME1';**

**NEW 6: NAME1:='MADAM';**

**REVERSE OF STRING IS:MADAM**

**MADAM IS PALINDROME**

**5.TO WRITE A PL/SQL BLOCK TO FIND SUM OF DIGITS OF A GIVEN NUMBER.**

**PROGRAM:**

**DECLARE**

**NUM NUMBER(5);**

**REM NUMBER(5);**

**SM NUMBER(5):=0;**

**NUM1 NUMBER(5);**

**BEGIN**

**NUM:=&NUM;**

**NUM1:=NUM;**

**WHILE(NUM>0)**

**LOOP**

**REM:=MOD(NUM,10);**

**SM:=SM+REM;**

**NUM:=TRUNC(NUM/10);**

**END LOOP;**

**DBMS\_OUTPUT.PUT\_LINE(NUM1||' SUM OF DIGITS OF A GIVEN NUMBERIS: '||SM);**

**END;**

**OUTPUT:**

**ENTER VALUE FOR NUM: 123**

**OLD 7: NUM:=&NUM;**

**NEW 7: NUM:=123;**

**SUM OF DIGITS OF 123 IS: 6**

**PL/SQL PROCEDURE SUCCESSFULLY COMPLETED.**

**6.TO WRITE A PL/SQL BLOCK TO GENERATE FIBONACCI SERIES**

**PROGRAM:**

**DECLARE**

**NUM NUMBER(5);**

**F1 NUMBER(5):=0;**

**F2 NUMBER(5):=1;**

**F3 NUMBER(5);**

**I NUMBER(5):=3;**

**BEGIN**

**NUM:=&NUM;**

**DBMS\_OUTPUT.PUT\_LINE('THE FIBONACCI SERIES IS:');**

**DBMS\_OUTPUT.PUT\_LINE(F1);**

**DBMS\_OUTPUT.PUT\_LINE(F2);**

**WHILE(I<=NUM)**

**LOOP**

**F3:=F1+F2;**

**DBMS\_OUTPUT.PUT\_LINE(F3);**

**F1:=F2;**

**F2:=F3;**

**I:=I+1;**

**END LOOP;**

**END;**

**OUTPUT:**

**ENTER VALUE FOR NUM: 10**

**OLD 8: NUM:=&NUM;
NEW 8: NUM:=10;
THE FIBONACCI SERIES IS:**

**0**

**1**

**1**

**2**

**3**

**5**

**8**

**13**

**21**

**34**

**PL/SQL PROCEDURE SUCCESSFULLY COMPLETED.**

**7. WRITE PL/SQL PROGRAM TO FIND FACTORIAL VALUE USING REVERSE FOR LOOP**

**PROGRAM:**

**DECLARE**

**V\_NUMFACT NUMBER:=&GIVENNUMBER;**

**V\_FACTORIAL NUMBER:=1;**

**BEGIN**

**FOR INDEXI IN REVERSE 1..V\_NUMFACT LOOP**

**V\_FACTORIAL:=V\_FACTORIAL\*INDEXI;**

**DBMS\_OUTPUT.PUT\_LINE('THE INDEXI NUMBER IS '||INDEXI);**

**END LOOP;**

**DBMS\_OUTPUT.PUT\_LINE('THE FACTORIAL OF '||V\_NUMFACT||'IS :'||V\_FACTORIAL);**

**END;**

**OUTPUT:**

**GIVEN NUMBER 8**

**OLD 2: V\_NUMFACT NUMBER:=&GIVENNUMBER;
NEW 2: V\_NUMFACT NUMBER:=9;**

**THE INDEXI NUMBER IS 8**

**THE INDEXI NUMBER IS7**

**THE INDEXI NUMBER IS6**

**THE INDEXI NUMBER IS5**

**THE INDEXI NUMBER IS4**

**THE INDEXI NUMBER IS3**

**THE INDEXI NUMBER IS2**

**THE INDEXI NUMBER IS1**

**THE FACTORIAL OF 9 IS 51840**

**PL/SQL PROCEDURE SUCCESSFULLY COMPLETED.**

**8.TO WRITE A TRIGGER TO ENSURE THAT DEPT TABLE DOESN’T CONTAIN DUPLICATE OF NULL VALUES IN DEPTNO COLUMN.**

**PROGRAM:**

**CREATE OR REPLACE TRIGGER TRIG1 BEFORE INSERT ON DEPT FOR EACH ROW**

**DECLARE**

**A NUMBER;**

**BEGIN**

**IF(:NEW.DEPTNO IS NULL)THEN**

**RAISE\_APPLICATION\_ERROR(-20001,'ERROR::DEPTNO CANNOT BE NULL');**

**ELSE**

**SELECT COUNT(\*) INTO A FROM DEPT WHERE DEPTNO=:NEW.DEPTNO;**

**IF(A=1) THEN**

**RAISE\_APPLICATION\_ERROR(-20002,'ERROR:: CANNOT HAVE DUPLICATE DEPTNO');**

**END IF;**

**END IF;**

**END;**

**OUTPUT:**

**TRIGGER CREATED.**

**SQL>SELECT \* FROM DEPT;**

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

**SQL> INSERT INTO DEPT VALUES(&DEPTNP,'&DNAME','&LOC');**

**OLD 1: INSERT INTO DEPT VALUES(&DEPTNP,'&DNAME','&LOC')
NEW 1: INSERT INTO DEPT VALUES(50,'MARKETING','HYDERABAD')
1 ROW CREATED**.

**SQL>SELECT \* FROM DEPT;**

|  |  |  |
| --- | --- | --- |
| **DEPTNO**  | **DNAME**  | **LOC**  |
| 10  | ACCOUNTING  | NEW YORK  |
| 20  | RESEARCH  | DALLAS  |
| 30  | SALES  | CHICAGO  |
| 40  | OPERATIONS  | BOSTON  |
| 50  | MARKETING  | HYDERABAD  |

**9.WRITE A PL/SQL TRIGGER TO TRANSATIONAL AUDITING ON SALARY COLUMN IN EMP TABLE**

**PROGRAM:**

**CREATE OR REPLACE TRIGGER CHKSALARY**

**BEFORE UPDATE OF SAL ON EMP**

**FOR EACH ROW**

**WHEN(NEW.SAL<OLD.SAL)**

**BEGIN**

**RAISE\_APPLICATION\_ERROR(-20500,'SALARY CANNOT BE DECREASED..');**

**END;**

**OUTPUT:**

**TRIGGER CREATED.**

**UPDATE EMP54 SET SAL=7000 WHERE SAL=8000;**

**'SALARY CANNOT BE DECREASED**