**Sample Table – Worker**

| **WORKER\_ID** | **FIRST\_NAME** | **LAST\_NAME** | **SALARY** | **JOINING\_DATE** | **DEPARTMENT** |
| --- | --- | --- | --- | --- | --- |
| 001 | Monika | Arora | 100000 | 2021-02-20 09:00:00 | HR |
| 002 | Niharika | Verma | 80000 | 2021-06-11 09:00:00 | Admin |
| 003 | Vishal | Singhal | 300000 | 2021-02-20 09:00:00 | HR |
| 004 | Amitabh | Singh | 500000 | 2021-02-20 09:00:00 | Admin |
| 005 | Vivek | Bhati | 500000 | 2021-06-11 09:00:00 | Admin |
| 006 | Vipul | Diwan | 200000 | 2021-06-11 09:00:00 | Account |
| 007 | Satish | Kumar | 75000 | 2021-01-20 09:00:00 | Account |
| 008 | Geetika | Chauhan | 90000 | 2021-04-11 09:00:00 | Admin |

**Sample Table – Bonus**

| **WORKER\_REF\_ID** | **BONUS\_DATE** | **BONUS\_AMOUNT** |
| --- | --- | --- |
| 1 | 2023-02-20 00:00:00 | 5000 |
| 2 | 2023-06-11 00:00:00 | 3000 |
| 3 | 2023-02-20 00:00:00 | 4000 |
| 1 | 2023-02-20 00:00:00 | 4500 |
| 2 | 2023-06-11 00:00:00 | 3500 |

**Sample Table – Title**

| **WORKER\_REF\_ID** | **WORKER\_TITLE** | **AFFECTED\_FROM** |
| --- | --- | --- |
| 1 | Manager | 2023-02-20 00:00:00 |
| 2 | Executive | 2023-06-11 00:00:00 |
| 8 | Executive | 2023-06-11 00:00:00 |
| 5 | Manager | 2023-06-11 00:00:00 |
| 4 | Asst. Manager | 2023-06-11 00:00:00 |
| 7 | Executive | 2023-06-11 00:00:00 |
| 6 | Lead | 2023-06-11 00:00:00 |
| 3 | Lead | 2023-06-11 00:00:00 |

To prepare the sample data, you can run the following queries

**SQL Script to Seed Sample Data.**

CREATE DATABASE ORG;

SHOW DATABASES;

USE ORG;

CREATE TABLE Worker (

WORKER\_ID INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

FIRST\_NAME CHAR(25),

LAST\_NAME CHAR(25),

SALARY INT(15),

JOINING\_DATE DATETIME,

DEPARTMENT CHAR(25)

);

INSERT INTO Worker

(WORKER\_ID, FIRST\_NAME, LAST\_NAME, SALARY, JOINING\_DATE, DEPARTMENT) VALUES

(001, 'Monika', 'Arora', 100000, '21-02-20 09.00.00', 'HR'),

(002, 'Niharika', 'Verma', 80000, '21-06-11 09.00.00', 'Admin'),

(003, 'Vishal', 'Singhal', 300000, '21-02-20 09.00.00', 'HR'),

(004, 'Amitabh', 'Singh', 500000, '21-02-20 09.00.00', 'Admin'),

(005, 'Vivek', 'Bhati', 500000, '21-06-11 09.00.00', 'Admin'),

(006, 'Vipul', 'Diwan', 200000, '21-06-11 09.00.00', 'Account'),

(007, 'Satish', 'Kumar', 75000, '21-01-20 09.00.00', 'Account'),

(008, 'Geetika', 'Chauhan', 90000, '21-04-11 09.00.00', 'Admin');

CREATE TABLE Bonus (

WORKER\_REF\_ID INT,

BONUS\_AMOUNT INT(10),

BONUS\_DATE DATETIME,

FOREIGN KEY (WORKER\_REF\_ID)

REFERENCES Worker(WORKER\_ID)

ON DELETE CASCADE

);

INSERT INTO Bonus

(WORKER\_REF\_ID, BONUS\_AMOUNT, BONUS\_DATE) VALUES

(001, 5000, '23-02-20'),

(002, 3000, '23-06-11'),

(003, 4000, '23-02-20'),

(001, 4500, '23-02-20'),

(002, 3500, '23-06-11');

CREATE TABLE Title (

WORKER\_REF\_ID INT,

WORKER\_TITLE CHAR(25),

AFFECTED\_FROM DATETIME,

FOREIGN KEY (WORKER\_REF\_ID)

REFERENCES Worker(WORKER\_ID)

ON DELETE CASCADE

);

INSERT INTO Title

(WORKER\_REF\_ID, WORKER\_TITLE, AFFECTED\_FROM) VALUES

(001, 'Manager', '2023-02-20 00:00:00'),

(002, 'Executive', '2023-06-11 00:00:00'),

(008, 'Executive', '2023-06-11 00:00:00'),

(005, 'Manager', '2023-06-11 00:00:00'),

(004, 'Asst. Manager', '2023-06-11 00:00:00'),

(007, 'Executive', '2023-06-11 00:00:00'),

(006, 'Lead', '2023-06-11 00:00:00'),

(003, 'Lead', '2023-06-11 00:00:00');

**Start practice with 50 SQL query interview questions.**

**Q-1. Write an SQL query to fetch “FIRST\_NAME” from the Worker table using the alias name <WORKER\_NAME>.**

**Ans.**

The required query is:

Select FIRST\_NAME AS WORKER\_NAME from Worker;

**Q-2. Write an SQL query to fetch “FIRST\_NAME” from the Worker table in upper case.**

**Ans.**

The required query is:

Select upper(FIRST\_NAME) from Worker;

**Q-3. Write an SQL query to fetch unique values of DEPARTMENT from the Worker table.**

**Ans.**

The required query is:

Select distinct DEPARTMENT from Worker;

**Q-4. Write an SQL query to print the first three characters of  FIRST\_NAME from the Worker table.**

**Ans.**

The required query is:

Select substring(FIRST\_NAME,1,3) from Worker;

**Q-5. Write an SQL query to find the position of the alphabet (‘a’) in the first name column ‘Amitabh’ from the Worker table.**

**Ans.**

The required query is:

Select INSTR(FIRST\_NAME, BINARY'a') from Worker where FIRST\_NAME = 'Amitabh';

**Q-6. Write an SQL query to print the FIRST\_NAME from the Worker table after removing white spaces from the right side.**

**Ans.**

The required query is:

Select RTRIM(FIRST\_NAME) from Worker;

**Q-7. Write an SQL query to print the DEPARTMENT from the Worker table after removing white spaces from the left side.**

**Ans.**

The required query is:

Select LTRIM(DEPARTMENT) from Worker;

**Q-8. Write an SQL query that fetches the unique values of DEPARTMENT from the Worker table and prints its length.**

**Ans.**

The required query is:

Select distinct length(DEPARTMENT) from Worker;

**Q-9. Write an SQL query to print the FIRST\_NAME from the Worker table after replacing ‘a’ with ‘A’.**

**Ans.**

The required query is:

Select REPLACE(FIRST\_NAME,'a','A') from Worker;

**Q-10. Write an SQL query to print the FIRST\_NAME and LAST\_NAME from the Worker table into a single column COMPLETE\_NAME. A space char should separate them.**

**Ans.**

The required query is:

Select CONCAT(FIRST\_NAME, ' ', LAST\_NAME) AS 'COMPLETE\_NAME' from Worker;

**Q-11. Write an SQL query to print all Worker details from the Worker table order by FIRST\_NAME Ascending.**

**Ans.**

The required query is:

Select \* from Worker order by FIRST\_NAME asc;

**Q-12. Write an SQL query to print all Worker details from the Worker table order by FIRST\_NAME Ascending and DEPARTMENT Descending.**

**Ans.**

The required query is:

Select \* from Worker order by FIRST\_NAME asc,DEPARTMENT desc;

**Q-13. Write an SQL query to print details for Workers with the first names “Vipul” and “Satish” from the Worker table.**

**Ans.**

The required query is:

Select \* from Worker where FIRST\_NAME in ('Vipul','Satish');

**Q-14. Write an SQL query to print details of workers excluding first names, “Vipul” and “Satish” from the Worker table.**

**Ans.**

The required query is:

Select \* from Worker where FIRST\_NAME not in ('Vipul','Satish');

**Q-15. Write an SQL query to print details of Workers with DEPARTMENT name as “Admin”.**

**Ans.**

The required query is:

Select \* from Worker where DEPARTMENT like 'Admin%';

**Q-16. Write an SQL query to print details of the Workers whose FIRST\_NAME contains ‘a’.**

**Ans.**

The required query is:

Select \* from Worker where FIRST\_NAME like '%a%';

**Q-17. Write an SQL query to print details of the Workers whose FIRST\_NAME ends with ‘a’.**

**Ans.**

The required query is:

Select \* from Worker where FIRST\_NAME like '%a';

**Q-18. Write an SQL query to print details of the Workers whose FIRST\_NAME ends with ‘h’ and contains six alphabets.**

**Ans.**

The required query is:

Select \* from Worker where FIRST\_NAME like '\_\_\_\_\_h';

**Q-19. Write an SQL query to print details of the Workers whose SALARY lies between 100000 and 500000.**

**Ans.**

The required query is:

Select \* from Worker where SALARY between 100000 and 500000;

**Q-20. Write an SQL query to print details of the Workers who joined in Feb 2021.**

**Ans.**

The required query is:

Select \* from Worker where year(JOINING\_DATE) = 2021 and month(JOINING\_DATE) = 2;

**Q-21. Write an SQL query to fetch the count of employees working in the department ‘Admin’.**

**Ans.**

The required query is:

SELECT COUNT(\*) FROM worker WHERE DEPARTMENT = 'Admin';

**Q-22. Write an SQL query to fetch worker names with salaries >= 50000 and <= 100000.**

**Ans.**

The required query is:

SELECT CONCAT(FIRST\_NAME, ' ', LAST\_NAME) As Worker\_Name, Salary

FROM worker

WHERE WORKER\_ID IN

(SELECT WORKER\_ID FROM worker

WHERE Salary BETWEEN 50000 AND 100000);

**Q-23. Write an SQL query to fetch the number of workers for each department in descending order.**

**Ans.**

The required query is:

SELECT DEPARTMENT, count(WORKER\_ID) No\_Of\_Workers

FROM worker

GROUP BY DEPARTMENT

ORDER BY No\_Of\_Workers DESC;

**Q-24. Write an SQL query to print details of the Workers who are also Managers.**

**Ans.**

The required query is:

SELECT DISTINCT W.FIRST\_NAME, T.WORKER\_TITLE

FROM Worker W

INNER JOIN Title T

ON W.WORKER\_ID = T.WORKER\_REF\_ID

AND T.WORKER\_TITLE in ('Manager');

**Q-25. Write an SQL query to fetch duplicate records having matching data in some fields of a table.**

**Ans.**

The required query is:

SELECT WORKER\_TITLE, AFFECTED\_FROM, COUNT(\*)

FROM Title

GROUP BY WORKER\_TITLE, AFFECTED\_FROM

HAVING COUNT(\*) > 1;

**Q-26. Write an SQL query to show only odd rows from a table.**

**Ans.**

The required query is:

SELECT \* FROM Worker WHERE MOD (WORKER\_ID, 2) <> 0;

**Q-27. Write an SQL query to show only even rows from a table.**

**Ans.**

The required query is:

SELECT \* FROM Worker WHERE MOD (WORKER\_ID, 2) = 0;

**Q-28. Write an SQL query to clone a new table from another table.**

**Ans.**

The general query to clone a table with data is:

SELECT \* INTO WorkerClone FROM Worker;

The general way to clone a table without information is:

SELECT \* INTO WorkerClone FROM Worker WHERE 1 = 0;

An alternate way to clone a table (for MySQL) without data is:

CREATE TABLE WorkerClone LIKE Worker;

**Q-29. Write an SQL query to fetch intersecting records of two tables.**

**Ans.**

The required query is:

(SELECT \* FROM Worker)

INTERSECT

(SELECT \* FROM WorkerClone);

**Q-30. Write an SQL query to show records from one table that another table does not have.**

**Ans.**

The required query is:

SELECT \* FROM Worker

MINUS

SELECT \* FROM Title;

**Q-31. Write an SQL query to show the current date and time.**

**Ans.**

The following MySQL query returns the current date:

SELECT CURDATE();

Whereas the following MySQL query returns the current date and time:

SELECT NOW();

Here is a SQL Server query that returns the current date and time:

SELECT getdate();

Find this Oracle query that also returns the current date and time:

SELECT SYSDATE FROM DUAL;

**Q-32. Write an SQL query to show the top n (say 10) records of a table.**

**Ans.**

MySQL query to return the top n records using the LIMIT method:

SELECT \* FROM Worker ORDER BY Salary DESC LIMIT 10;

SQL Server query to return the top n records using the TOP command:

SELECT TOP 10 \* FROM Worker ORDER BY Salary DESC;

Oracle query to return the top n records with the help of ROWNUM:

SELECT \* FROM (SELECT \* FROM Worker ORDER BY Salary DESC)

WHERE ROWNUM <= 10;

Now, that you should have a solid foundation in intermediate SQL, let’s take a look at some more advanced SQL query questions. These questions will require us to use more complex SQL syntax and concepts, such as nested queries, joins, unions, and intersects.

**Q-33. Write an SQL query to determine the nth (say n=5) highest salary from a table.**

**Ans.**

MySQL query to find the nth highest salary:

SELECT Salary FROM Worker ORDER BY Salary DESC LIMIT n-1,1;

SQL Server query to find the nth highest salary:

SELECT TOP 1 Salary

FROM (

SELECT DISTINCT TOP n Salary

FROM Worker

ORDER BY Salary DESC

)

ORDER BY Salary ASC;

**Q-34. Write an SQL query to determine the 5th highest salary without using the TOP or limit method.**

**Ans.**

The following query is using the correlated subquery to return the 5th highest salary:

SELECT Salary

FROM Worker W1

WHERE 4 = (

SELECT COUNT( DISTINCT ( W2.Salary ) )

FROM Worker W2

WHERE W2.Salary >= W1.Salary

);

Use the following generic method to find the nth highest salary without using TOP or limit.

SELECT Salary

FROM Worker W1

WHERE n-1 = (

SELECT COUNT( DISTINCT ( W2.Salary ) )

FROM Worker W2

WHERE W2.Salary >= W1.Salary

);

**Q-35. Write an SQL query to fetch the list of employees with the same salary.**

**Ans.**

The required query is:

Select distinct W.WORKER\_ID, W.FIRST\_NAME, W.Salary

from Worker W, Worker W1

where W.Salary = W1.Salary

and W.WORKER\_ID != W1.WORKER\_ID;

**Q-36. Write an SQL query to show the second-highest salary from a table.**

**Ans.**

The required query is:

Select max(Salary) from Worker

where Salary not in (Select max(Salary) from Worker);

**Q-37. Write an SQL query to show one row twice in the results from a table.**

**Ans.**

The required query is:

select FIRST\_NAME, DEPARTMENT from worker W where W.DEPARTMENT='HR'

union all

select FIRST\_NAME, DEPARTMENT from Worker W1 where W1.DEPARTMENT='HR';

**Q-38. Write an SQL query to fetch intersecting records of two tables.**

**Ans.**

The required query is:

(SELECT \* FROM Worker)

INTERSECT

(SELECT \* FROM WorkerClone);

**Q-39. Write an SQL query to fetch the first 50% of records from a table.**

**Ans.**

The required query is:

SELECT \*

FROM WORKER

WHERE WORKER\_ID <= (SELECT count(WORKER\_ID)/2 from Worker);

**Q-40. Write an SQL query to fetch the departments that have less than five people in them.**

**Ans.**

The required query is:

SELECT DEPARTMENT, COUNT(WORKER\_ID) as 'Number of Workers' FROM Worker GROUP BY DEPARTMENT HAVING COUNT(WORKER\_ID) < 5;

**Q-41. Write an SQL query to show all departments along with the number of people in there.**

**Ans.**

The following query returns the expected result:

SELECT DEPARTMENT, COUNT(DEPARTMENT) as 'Number of Workers' FROM Worker GROUP BY DEPARTMENT;

**Q-42. Write an SQL query to show the last record from a table.**

**Ans.**

The following query will return the last record from the Worker table:

Select \* from Worker where WORKER\_ID = (SELECT max(WORKER\_ID) from Worker);

**Q-43. Write an SQL query to fetch the first row of a table.**

**Ans.**

The required query is:

Select \* from Worker where WORKER\_ID = (SELECT min(WORKER\_ID) from Worker);

**###Q-44. Write an SQL query to fetch the last five records from a table.**

**Ans.**

The required query is:

SELECT \* FROM Worker WHERE WORKER\_ID <=5

UNION

SELECT \* FROM (SELECT \* FROM Worker W order by W.WORKER\_ID DESC) AS W1 WHERE W1.WORKER\_ID <=5;

**Q-45. Write an SQL query to print the names of employees having the highest salary in each department.**

**Ans.**

The required query is:

SELECT t.DEPARTMENT,t.FIRST\_NAME,t.Salary from(SELECT max(Salary) as TotalSalary,DEPARTMENT from Worker group by DEPARTMENT) as TempNew

Inner Join Worker t on TempNew.DEPARTMENT=t.DEPARTMENT

and TempNew.TotalSalary=t.Salary;

**Q-46. Write an SQL query to fetch three max salaries from a table.**

**Ans.**

The required query is:

SELECT distinct Salary from worker a WHERE 3 >= (SELECT count(distinct Salary) from worker b WHERE a.Salary <= b.Salary) order by a.Salary desc;

**Q-47. Write an SQL query to fetch three min salaries from a table.**

**Ans.**

The required query is:

SELECT distinct Salary from worker a WHERE 3 >= (SELECT count(distinct Salary) from worker b WHERE a.Salary >= b.Salary) order by a.Salary desc;

**Q-48. Write an SQL query to fetch nth max salaries from a table.**

**Ans.**

The required query is:

SELECT distinct Salary from worker a WHERE n >= (SELECT count(distinct Salary) from worker b WHERE a.Salary <= b.Salary) order by a.Salary desc;

**Q-49. Write an SQL query to fetch departments along with the total salaries paid for each of them.**

**Ans.**

The required query is:

 SELECT DEPARTMENT, sum(Salary) from worker group by DEPARTMENT;

**Q-50. Write an SQL query to fetch the names of workers who earn the highest salary.**

**Ans.**

The required query is:

SELECT FIRST\_NAME, SALARY from Worker WHERE SALARY=(SELECT max(SALARY) from Worker);

DECLARE

A NUMBER :=4;

B number :=3;

c number;

BEGIN

c:=a+b;

Dbms\_output.put\_line('this is variable a='||A);

Dbms\_output.put\_line('this is variable b='||B);

Dbms\_output.put\_line('this is variable c='||C);

end;

wap pl/sql block for the greater among the two variable

DECLARE

A NUMBER :=4;

B number :=3;

c number;

BEGIN

if(A>B)

then

Dbms\_output.put\_line('A is greater'||A);

else

Dbms\_output.put\_line('B is greater'||B);

end if;

end;

WAP pl/SQL block for greater than 3 number

DECLARE

A NUMBER :=4;

B number :=3;

c number :=10;

BEGIN

if(A>B) and (A>C)

then

Dbms\_output.put\_line('A is greater'||A);

else if(B>c) and (B>A) then

Dbms\_output.put\_line('B is greater'||B);

else

Dbms\_output.put\_line('C is greater'||C);

end if;

end if;

end;

DECLARE

age number :=1;

c varchar(20):=’Indian’;

BEGIN

if (age>18) and c:=’indian’

then

Dbms\_output.put\_line('Eligible for voting'||age);

else

Dbms\_output.put\_line('Not eligible for voting'||age);

end if;

end;

DECLARE

a number:=4;

b number:=5;

c number;

BEGIN

--Dbms\_output.put\_line('Before swapping'||a);

Dbms\_output.put\_line('Before swapping'||a);

Dbms\_output.put\_line('Before swapping'||b);

c:=a;

a:=b;

b:=c;

--.put\_line('After swapping');

Dbms\_output.put\_line('After swapping'||a);

Dbms\_output.put\_line('After swapping'||b);

end;

wap for to print the 1 to 20 using for loop;

DECLARE

i number;

n number:=20;

BEGIN

for i in 1..n

loop

Dbms\_output.put\_line(i);

end loop;

end;

wap for the print the 20 to 1 in pl/sql block;

DECLARE

i number;

n number:=20;

BEGIN

for i in reverse 1..n

loop

Dbms\_output.put\_line(i);

end loop;

end;

wap for the to print table of 7

DECLARE

i number ;

n number:=10;

BEGIN

for i in 1..n

loop

Dbms\_output.put\_line(7||'\*'||i||'='||7\*i);

end loop;

end;

wap for pl/sql to print pattern as follows

DECLARE

i number:=4 ;

j number;

n number:=4;

BEGIN

for i in 1..n

loop

for j in 1..i

loop

Dbms\_output.put('\*');

end loop;

Dbms\_output.new\_line;

end loop;

end;

\*  
\*\*  
\*\*\*  
\*\*\*\*

wap for pl/sql to print pattern as follows

DECLARE

i number:=4 ;

j number;

n number:=4;

BEGIN

for i in reverse 1..n

loop

for j in 1..i

loop

Dbms\_output.put('\*');

end loop;

Dbms\_output.new\_line;

end loop;

end;

Statement processed.  
\*\*\*\*  
\*\*\*  
\*\*  
\*

wap for pl/sql to print pattern as follows

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

DECLARE

i number:=4 ;

j number;

n number:=4;

BEGIN

for i in 1..n

loop

for j in 1..i

loop

Dbms\_output.put('\*');

end loop;

Dbms\_output.new\_line;

end loop;

for i in reverse 1..n

loop

for j in 1..i

loop

Dbms\_output.put('\*');

end loop;

Dbms\_output.new\_line;

end loop;

end;

4)\* \* \* \*

\* \* \*

\* \*

\*

DECLARE

N NUMBER := 4; -- Set the number of rows you want in the pattern

BEGIN

FOR I IN 1..N LOOP

FOR J IN 1..I - 1 LOOP

DBMS\_OUTPUT.PUT(' ');

END LOOP;

FOR J IN I..N LOOP

DBMS\_OUTPUT.PUT('\*');

END LOOP;

DBMS\_OUTPUT.NEW\_LINE;

END LOOP;

END;

5) \*

\* \*

\* \* \*

\* \* \* \*

6)

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

DECLARE

N NUMBER := 4; -- Set the number of rows you want in the pattern

BEGIN

FOR I IN 1..N LOOP

FOR J IN 1..N - I LOOP

DBMS\_OUTPUT.PUT(' ');

END LOOP;

FOR J IN 1..I LOOP

DBMS\_OUTPUT.PUT('\*');

END LOOP;

DBMS\_OUTPUT.NEW\_LINE;

END LOOP;

FOR I IN REVERSE 1..N-1 LOOP

FOR J IN 1..N - I LOOP

DBMS\_OUTPUT.PUT(' ');

END LOOP;

FOR J IN 1..I LOOP

DBMS\_OUTPUT.PUT('\*');

END LOOP;

DBMS\_OUTPUT.NEW\_LINE;

END LOOP;

END;

7)Wap for PL/SQL to reverse the number

DECLARE

rev number :=0;

r number;

n number:=12345;

BEGIN

while n>0

loop

r:=mod(n, 10);

rev:=(rev\*10)+r;

n:=trunc(n/10);

end loop;

Dbms\_output.put\_line('The reverse of number'||rev);

END;

8)wap FOR PL/sql BLOCK FOR PALINDROME NUMBER

DECLARE

rev number :=0;

r number;

n number:=121;

temp number;

BEGIN

temp:=n;

while n>0

loop

r:=mod(n, 10);

rev:=(rev\*10)+r;

n:=trunc(n/10);

end loop;

if(temp=rev) then

Dbms\_output.put\_line('No is palindrome');

else

Dbms\_output.put\_line('No is not palindrome');

end if;

end;

9) wap FOR PL/sql BLOCK SUM OF ALL DIGIT ARE EVEN OR ODD

10) wap for PL/SQL block factorial of number

DECLARE

n number :=6;

i number;

fact number :=1;

BEGIN

for i in 1..n

loop

fact:=fact\*i;

end loop;

Dbms\_output.put\_line('factorial of number'||n||'is'||fact);

end;

11)WAP for PL/SQL block to print grade of system as follows

A-execellent

B-very good

c-good

D-Average

E-Pass

Fail

DECLARE

grade char(1):='c';

BEGIN

Case

when grade='A' then

Dbms\_output.put\_line('Excellent');

when grade='B' then

Dbms\_output.put\_line('Very good');

when grade='c' then

Dbms\_output.put\_line(' good');

when grade='D' then

Dbms\_output.put\_line('Average');

when grade='E' then

Dbms\_output.put\_line('Pass');

else

Dbms\_output.put\_line('Failed');

end Case;

end;

11)WAP for the find the billing system

DECLARE

taxi varchar(20):='CAR';

dist number:=200;

BEGIN

Case

when taxi='CAR' then

Dbms\_output.put\_line('RATE/KM'||8\*dist);

when taxi='VAN' then

Dbms\_output.put\_line('RATE/KM'||10\*dist);

when taxi='BUS' then

Dbms\_output.put\_line('RATE/KM'||20\*dist);

when taxi='BIKE' then

Dbms\_output.put\_line('RATE/KM'||5\*dist);

when taxi='SOOTY'then

Dbms\_output.put\_line('RATE/KM'||4\*dist);

end Case;

end;

12)wap print the number in character form

DECLARE

num int :=1234;

BEGIN

Case

when num=1 then

Dbms\_output.put\_line('one');

when num=2 then

Dbms\_output.put\_line('two');

when num=3 then

Dbms\_output.put\_line('three');

when num=4 then

Dbms\_output.put\_line('four');

when num=5 then

Dbms\_output.put\_line('five');

when num=6 then

Dbms\_output.put\_line('six');

when num=7 then

Dbms\_output.put\_line('seven');

when num=8 then

Dbms\_output.put\_line('eight');

when num=9 then

Dbms\_output.put\_line('nine');

end Case;

end;

13)WAP print the sum of digit for the given number if the number is given 234 then o/p

DECLARE

NUM number:=234;

s number:=0;

r number;

BEGIN

while num>0

loop

r:=mod(num, 10);

s := s+r;

num:=trunc(num/10);

end loop;

if mod(s,2) = 0 then

Dbms\_output.put\_line('Sum is even');

else

Dbms\_output.put\_line('Sum is odd');

end if;

END;

14) \*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

DECLARE

N NUMBER := 4; -- Set the number of rows you want in the pattern

BEGIN

-- Upper part of the pattern

FOR I IN 1..N LOOP

-- Print leading spaces

FOR J IN 1..N - I LOOP

DBMS\_OUTPUT.PUT(' ');

END LOOP;

-- Print asterisks

FOR J IN 1..(2\*I - 1) LOOP

DBMS\_OUTPUT.PUT('\*');

END LOOP;

-- Move to the next line

DBMS\_OUTPUT.NEW\_LINE;

END LOOP;

-- Lower part of the pattern

FOR I IN REVERSE 1..N-1 LOOP

-- Print leading spaces

FOR J IN 1..N - I LOOP

DBMS\_OUTPUT.PUT(' ');

END LOOP;

-- Print asterisks

FOR J IN 1..(2\*I - 1) LOOP

DBMS\_OUTPUT.PUT('\*');

END LOOP;

-- Move to the next line

DBMS\_OUTPUT.NEW\_LINE;

END LOOP;

END;