# DATABASE SYSTEMS

## DQL

By Sana Faiz
Sana.faiz.muet83@gmail.com

## CATEGORIES OF SQL STATEMENTS

1. **Data Definition Languages (DDL).**
2. **Data Query Languages (DQL).**
3. **Data Manipulation Languages (DML).**
4. **Data Control Languages (DCL).**
5. **Transaction Control Languages (TCL).**

# DATA QUERY LANGUAGES (DQL)

- DQL statements are used for performing queries on the data within schema objects.
- Following is the command included in this category:

**1. SELECT**

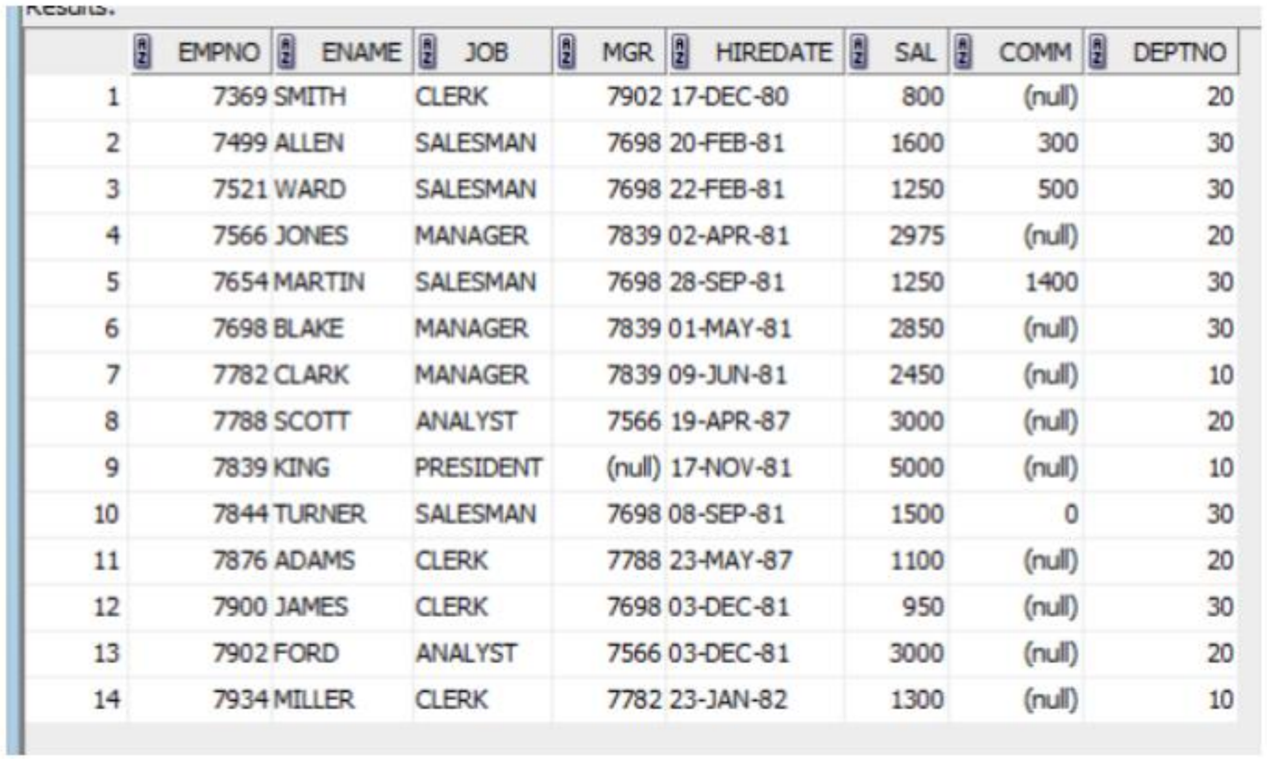
## EMP TABLE

| Empno (PK) | Ename | job | Mgr | hiredate | sal | comm | Deptno (FK) |
|---|---|---|---|---|---|---|---|
| 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 | BLAKE | MANAGER | 7839 | 01-MAY-81 | 2850 | | 30 |

| Deptno (PK) | dname | loc |
|---|---|---|
| 10 | ACCOUNTING | NEW YORK |
| 20 | RESEARCH | DALLAS |
| 30 | SALES | CHICAGO |

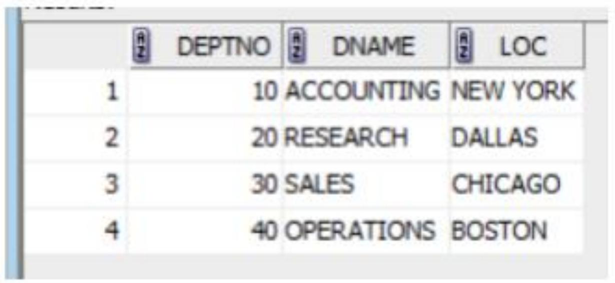
## DEPT TABLE

## EMP TABE



| | EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|---|---|---|---|---|---|---|---|---|
| 1 | 7369 | SMITH | CLERK | 7902 | 17-DEC-80 | 800 | (null) | 20 |
| 2 | 7499 | ALLEN | SALESMAN | 7698 | 20-FEB-81 | 1600 | 300 | 30 |
| 3 | 7521 | WARD | SALESMAN | 7698 | 22-FEB-81 | 1250 | 500 | 30 |
| 4 | 7566 | JONES | MANAGER | 7839 | 02-APR-81 | 2975 | (null) | 20 |
| 5 | 7654 | MARTIN | SALESMAN | 7698 | 28-SEP-81 | 1250 | 1400 | 30 |
| 6 | 7698 | BLAKE | MANAGER | 7839 | 01-MAY-81 | 2850 | (null) | 30 |
| 7 | 7782 | CLARK | MANAGER | 7839 | 09-JUN-81 | 2450 | (null) | 10 |
| 8 | 7788 | SCOTT | ANALYST | 7566 | 19-APR-87 | 3000 | (null) | 20 |
| 9 | 7839 | KING | PRESIDENT | (null) | 17-NOV-81 | 5000 | (null) | 10 |
| 10 | 7844 | TURNER | SALESMAN | 7698 | 08-SEP-81 | 1500 | 0 | 30 |
| 11 | 7876 | ADAMS | CLERK | 7788 | 23-MAY-87 | 1100 | (null) | 20 |
| 12 | 7900 | JAMES | CLERK | 7698 | 03-DEC-81 | 950 | (null) | 30 |
| 13 | 7902 | FORD | ANALYST | 7566 | 03-DEC-81 | 3000 | (null) | 20 |
| 14 | 7934 | MILLER | CLERK | 7782 | 23-JAN-82 | 1300 | (null) | 10 |

## DEPT TABE



| | DEPTNO | DNAME | LOC |
|---|---|---|---|
| 1 | 10 | ACCOUNTING | NEW YORK |
| 2 | 20 | RESEARCH | DALLAS |
| 3 | 30 | SALES | CHICAGO |
| 4 | 40 | OPERATIONS | BOSTON |

## SELECT STATEMENT

- SELECT statement retrieves information from the database. Using a SELECT statement, you can do the following:

1. **Projection:** You can use the projection capability in SQL to choose the columns in a table that you want to be returned by your query. You can choose as few or as many columns in the table as you require.
2. **Selection:** You can use the selection capability in SQL to choose the rows in a table that you want to be returned by a query.
   - You can use various criteria to restrict the rows that you see.
3. **Joining:** You can use the join capability in SQL to bring together data that is stored in different tables by creating a link between them.

**SYNTAX:**

```
SELECT * | [ DISTINCT | UNIQUE] (column_ name [ AS alias ] ,arithmetic
expr)
FROM              table _ name  [,……………]
[ WHERE           condition      ]
[ GROUP BY        column_list    ]
[ HAVING          condition      ]
[ ORDER BY        column_list    ] ;
```

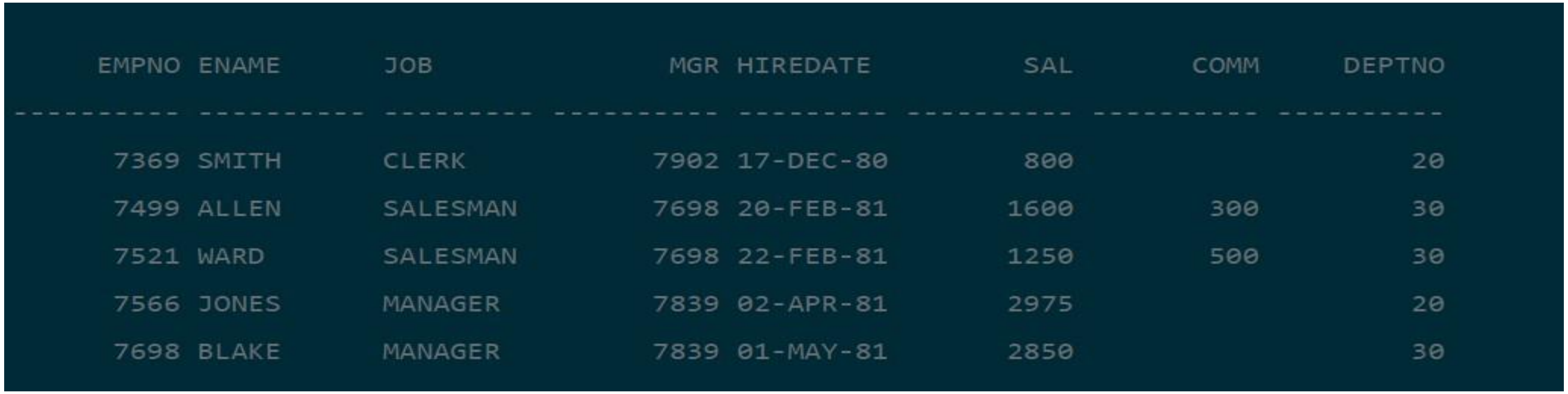
## RETRIVING THE COMPLETE TABLE

```
SELECT * | [ DISTINCT | UNIQUE] (column_ name [ AS alias ] ,arithmetic
expr) FROM table _ name  [,……………]
```

**EXAMPLE A:**

```
SELECT *
    FROM emp ;
```

**OUTPUT:**



```
     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
      7698 BLAKE      MANAGER         7839 01-MAY-81       2850                    30
```

**8 COLUMNS , 14 ROWS**

## RETRIVING SPECIFIC COLUMNS

```
SELECT * I [ DISTINCT I UNIQUE] (column_ name [ AS alias ] ,arithmetic expr)
FROM    table _ name  [,……………]
```

**EXAMPLE B:**

```
SELECT empno, ename, sal, deptno
FROM emp ;
```

**OUTPUT:**

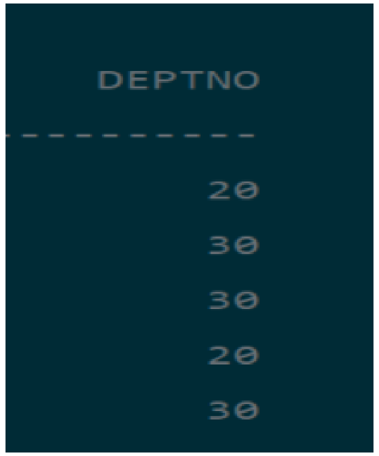
**4 COLUMNS , 14 ROWS**

**EXAMPLE C:**

```
SELECT deptno
FROM emp ;
```

**OUTPUT:**

**1 COLUMN, 14 ROWS**



```
    DEPTNO
----------
        20
        30
        30
        20
        30
```

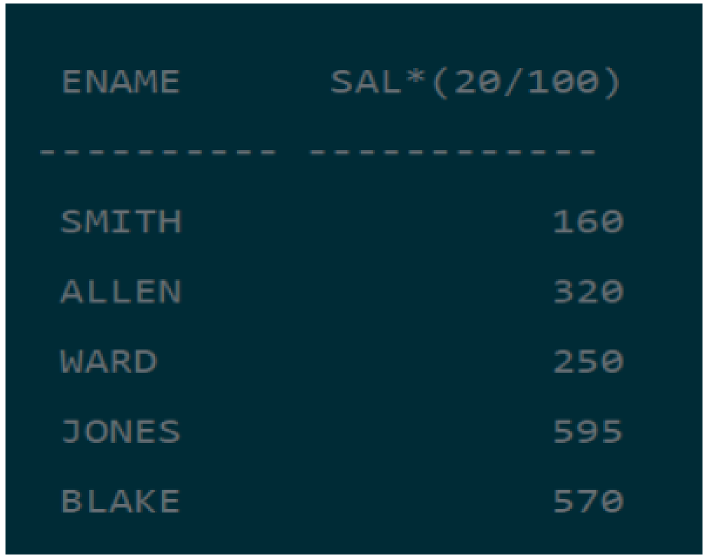
## USING ARITHMETIC EXPRESSIONS

```
SELECT * I [ DISTINCT I UNIQUE] (column_ name [ AS alias ] ,arithmetic expr)
FROM      table _ name  [,...............]
```

**EXAMPLE D:**

```
SELECT ename , sal * (20/100)
FROM emp ;
```

**OUTPUT:**



```
ENAME       SAL*(20/100)
----------  ------------
SMITH                160
ALLEN                320
WARD                 250
JONES                595
BLAKE                570
```

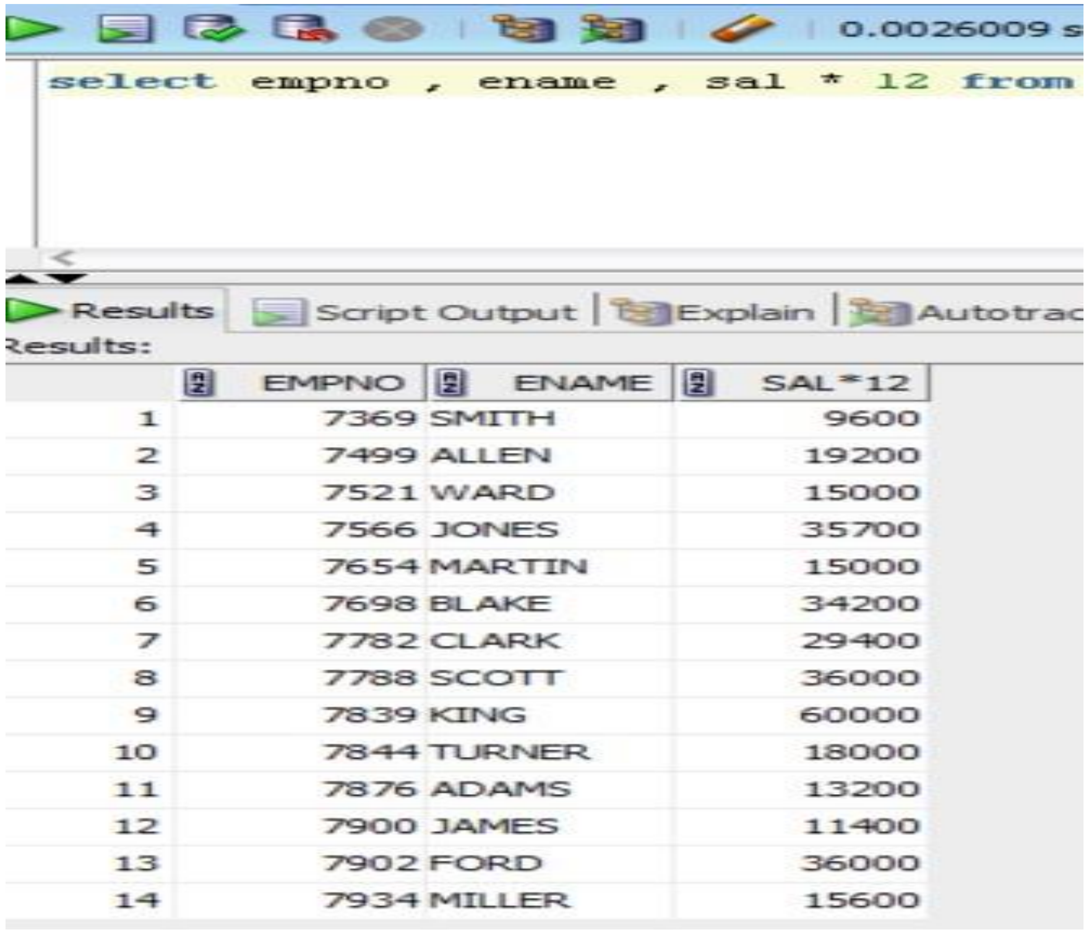
# TASK A

- Display the Annual Salary of all the employees.

**QUERY:** SELECT empno , ename , sal*12

FROM emp ;

**OUTPUT:**



```
select empno , ename , sal * 12 from
```

Results:

| | EMPNO | ENAME | SAL*12 |
|---|---|---|---|
| 1 | 7369 | SMITH | 9600 |
| 2 | 7499 | ALLEN | 19200 |
| 3 | 7521 | WARD | 15000 |
| 4 | 7566 | JONES | 35700 |
| 5 | 7654 | MARTIN | 15000 |
| 6 | 7698 | BLAKE | 34200 |
| 7 | 7782 | CLARK | 29400 |
| 8 | 7788 | SCOTT | 36000 |
| 9 | 7839 | KING | 60000 |
| 10 | 7844 | TURNER | 18000 |
| 11 | 7876 | ADAMS | 13200 |
| 12 | 7900 | JAMES | 11400 |
| 13 | 7902 | FORD | 36000 |
| 14 | 7934 | MILLER | 15600 |

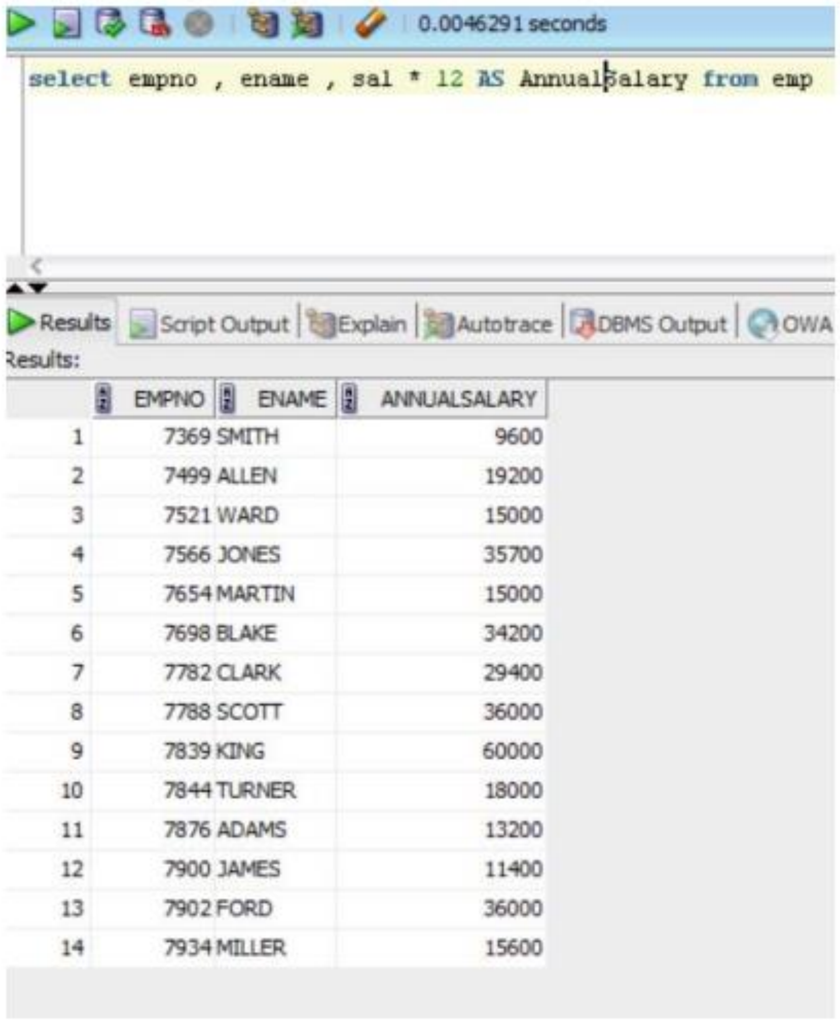
## USING ALIAS

SELECT * I [ DISTINCT I UNIQUE] (column_ name [ AS alias ] ,arithmetic expr)
FROM table _ name [,..............]

**EXAMPLE E:**

SELECT empno , ename , sal*12 AS AnnualSalary
FROM emp ;

**OUTPUT EXAMPLE E:**



```
select empno , ename , sal * 12 AS AnnualSalary from emp
```

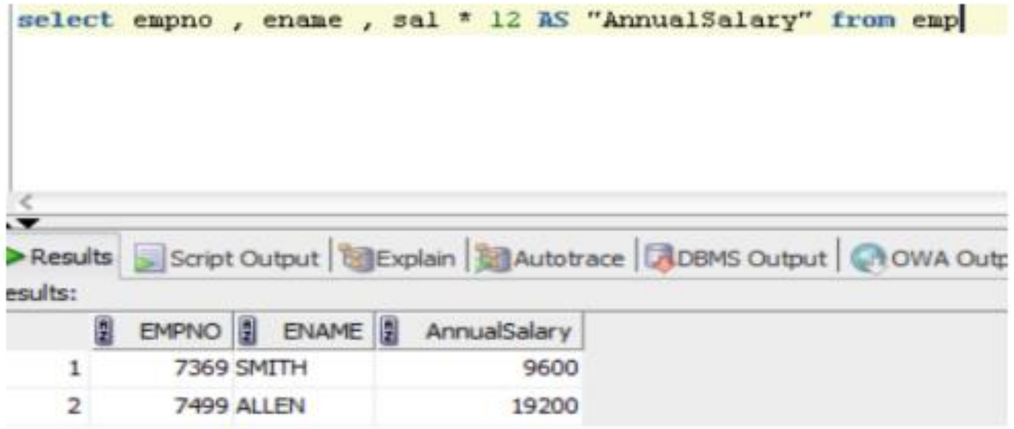
Results:

| | EMPNO | ENAME | ANNUALSALARY |
|---|---|---|---|
| 1 | 7369 | SMITH | 9600 |
| 2 | 7499 | ALLEN | 19200 |
| 3 | 7521 | WARD | 15000 |
| 4 | 7566 | JONES | 35700 |
| 5 | 7654 | MARTIN | 15000 |
| 6 | 7698 | BLAKE | 34200 |
| 7 | 7782 | CLARK | 29400 |
| 8 | 7788 | SCOTT | 36000 |
| 9 | 7839 | KING | 60000 |
| 10 | 7844 | TURNER | 18000 |
| 11 | 7876 | ADAMS | 13200 |
| 12 | 7900 | JAMES | 11400 |
| 13 | 7902 | FORD | 36000 |
| 14 | 7934 | MILLER | 15600 |

**EXAMPLE F:**

SELECT empno , ename , sal*12 AS “AnnualSalary”
FROM emp ;

**OUTPUT:**



```
select empno , ename , sal * 12 AS "AnnualSalary" from emp
```

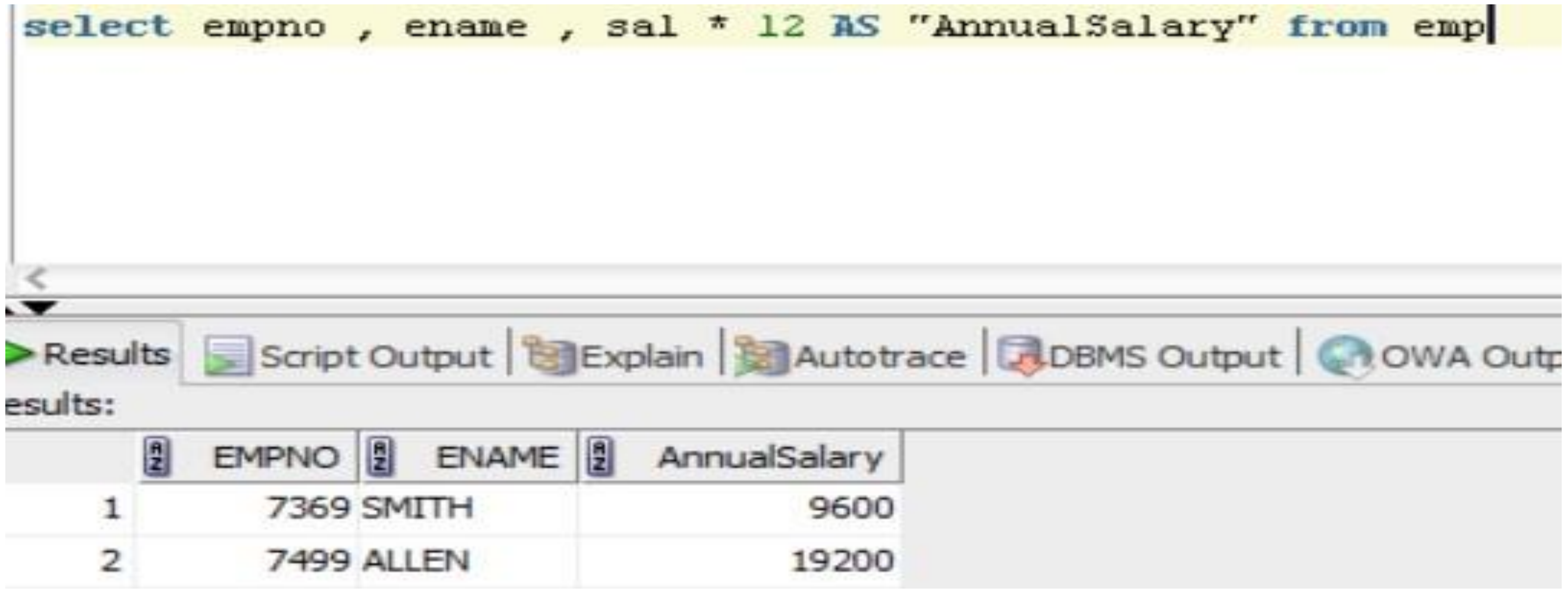
esults:

| | EMPNO | ENAME | AnnualSalary |
|---|---|---|---|
| 1 | 7369 | SMITH | 9600 |
| 2 | 7499 | ALLEN | 19200 |

**EXAMPLE F:**

SELECT empno , ename , sal*12 AS “AnnualSalary” FROM emp ;

**OUTPUT:**



```
select empno , ename , sal * 12 AS "AnnualSalary" from emp
```

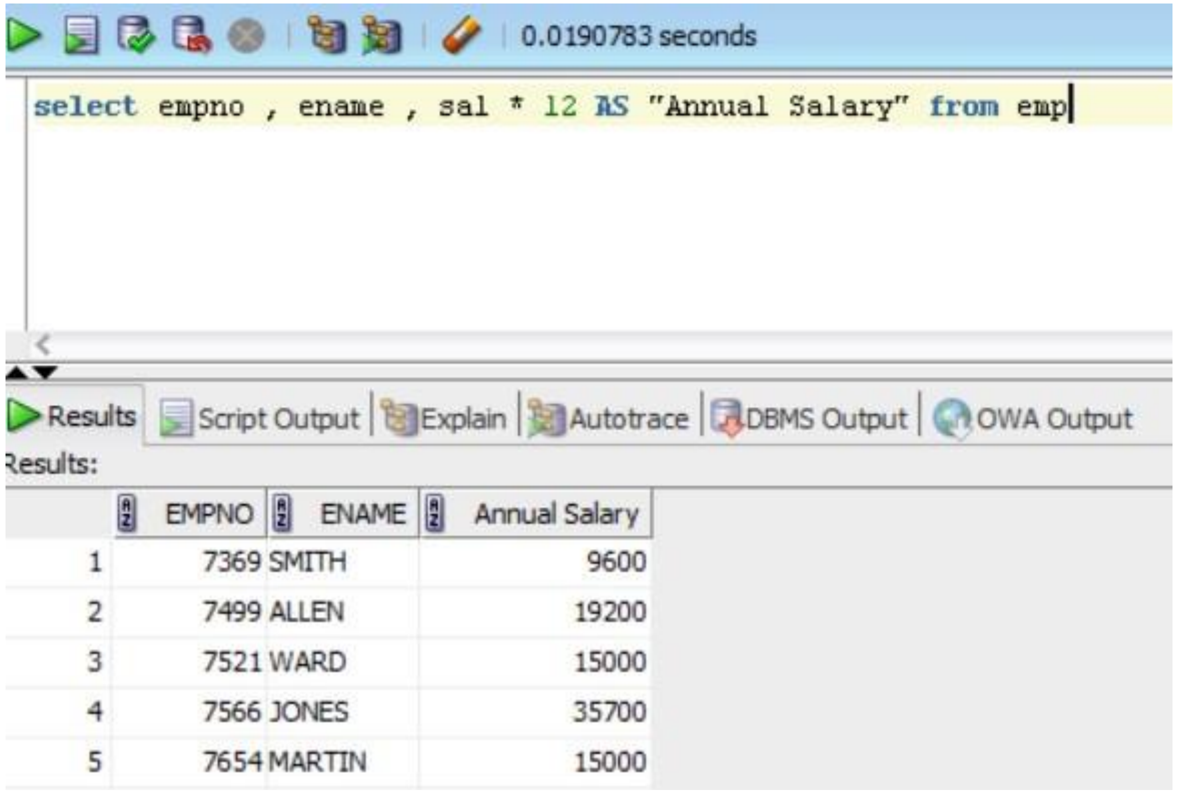
esults:

| | EMPNO | ENAME | AnnualSalary |
|---|---|---|---|
| 1 | 7369 | SMITH | 9600 |
| 2 | 7499 | ALLEN | 19200 |

## EXAMPLE G:

SELECT empno , ename , sal*12 AS “Annual Salary” FROM  emp ;

## OUTPUT:

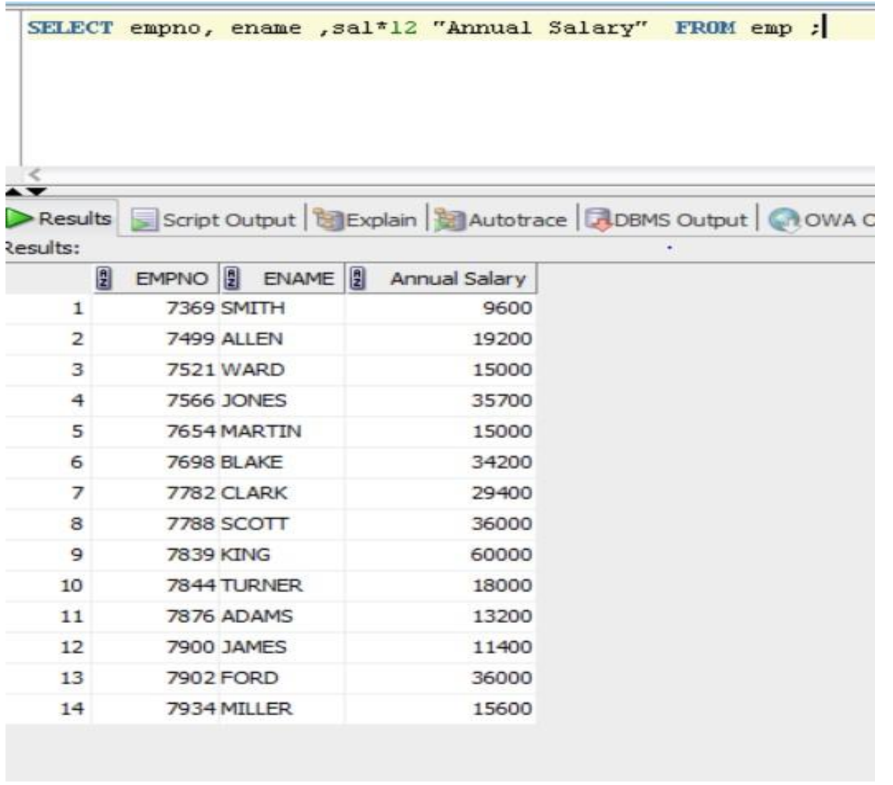


```
select empno , ename , sal * 12 AS "Annual Salary" from emp
```

Results:

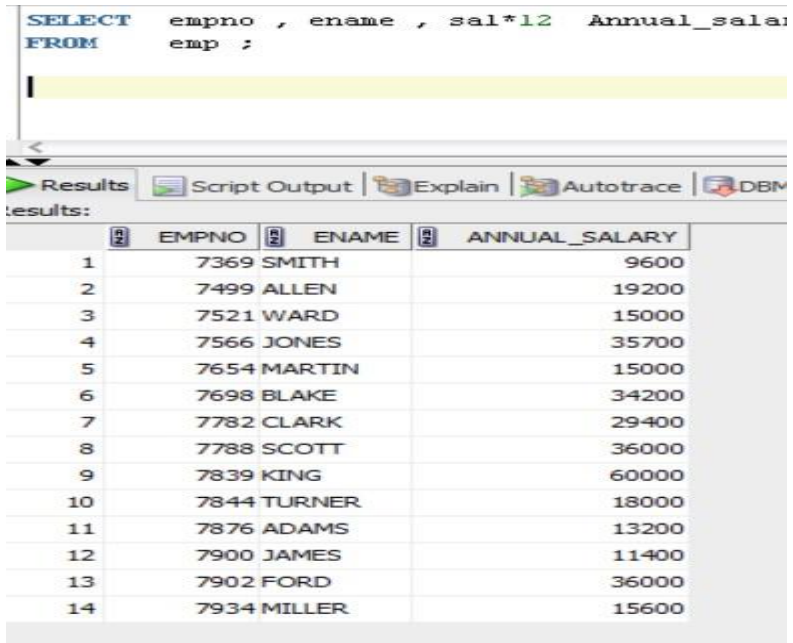
| | EMPNO | ENAME | Annual Salary |
|---|---|---|---|
| 1 | 7369 | SMITH | 9600 |
| 2 | 7499 | ALLEN | 19200 |
| 3 | 7521 | WARD | 15000 |
| 4 | 7566 | JONES | 35700 |
| 5 | 7654 | MARTIN | 15000 |

1. SELECT empno , ename , sal*12 “Annual Salary” FROM emp ; ✓
2. SELECT empno , ename , sal*12 AS Annual Salary FROM emp ; ✗
3. SELECT empno , ename , sal*12 Annual FROM emp ; ✓
4. SELECT empno , ename , sal*12 Annual_salary FROM emp ; ✓
5. SELECT empno , ename , sal*12 'Annual_salary’ FROM emp ; ✗



```
SELECT empno, ename ,sal*12 "Annual Salary"  FROM emp ;
```

| | EMPNO | ENAME | Annual Salary |
|---|---|---|---|
| 1 | 7369 | SMITH | 9600 |
| 2 | 7499 | ALLEN | 19200 |
| 3 | 7521 | WARD | 15000 |
| 4 | 7566 | JONES | 35700 |
| 5 | 7654 | MARTIN | 15000 |
| 6 | 7698 | BLAKE | 34200 |
| 7 | 7782 | CLARK | 29400 |
| 8 | 7788 | SCOTT | 36000 |
| 9 | 7839 | KING | 60000 |
| 10 | 7844 | TURNER | 18000 |
| 11 | 7876 | ADAMS | 13200 |
| 12 | 7900 | JAMES | 11400 |
| 13 | 7902 | FORD | 36000 |
| 14 | 7934 | MILLER | 15600 |



```
SELECT     empno  ,  ename  ,  sal*12    Annual_sala
FROM       emp  ;
```

| | EMPNO | ENAME | ANNUAL_SALARY |
|---|---|---|---|
| 1 | 7369 | SMITH | 9600 |
| 2 | 7499 | ALLEN | 19200 |
| 3 | 7521 | WARD | 15000 |
| 4 | 7566 | JONES | 35700 |
| 5 | 7654 | MARTIN | 15000 |
| 6 | 7698 | BLAKE | 34200 |
| 7 | 7782 | CLARK | 29400 |
| 8 | 7788 | SCOTT | 36000 |
| 9 | 7839 | KING | 60000 |
| 10 | 7844 | TURNER | 18000 |
| 11 | 7876 | ADAMS | 13200 |
| 12 | 7900 | JAMES | 11400 |
| 13 | 7902 | FORD | 36000 |
| 14 | 7934 | MILLER | 15600 |

## USING DISTINCT KEYWORD

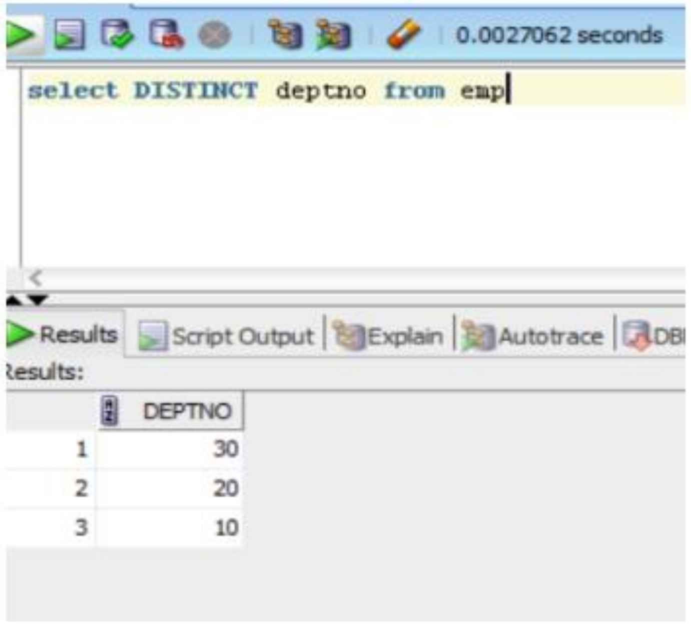
SELECT * I [ DISTINCT I UNIQUE] (column_ name [ AS alias ] ,arithmetic expr)
FROM table _ name [,...............]

**EXAMPLE H:**

SELECT DISTINCT deptno

FROM emp ;

**OUTPUT:**



```
select DISTINCT deptno from emp
```

0.0027062 seconds

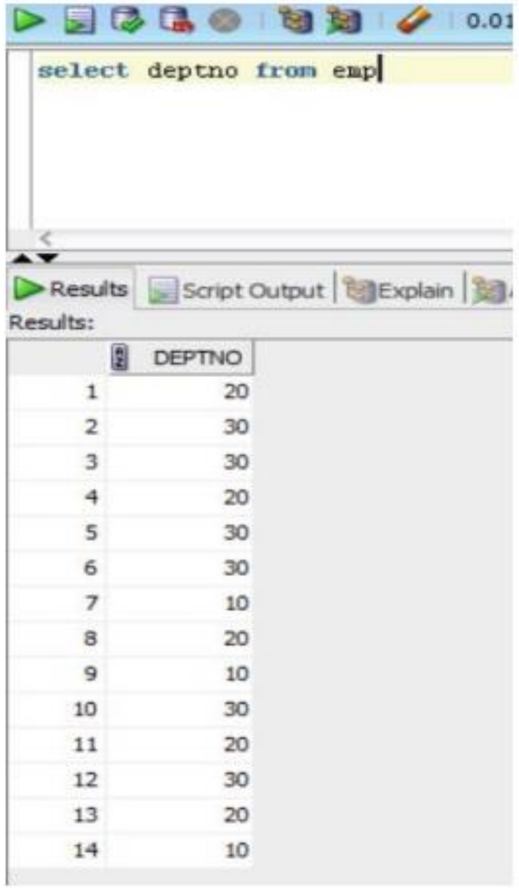
```
Results:
     DEPTNO
1        30
2        20
3        10
```

**EXAMPLE C:**

SELECT deptno

FROM emp ;

**OUTPUT:**



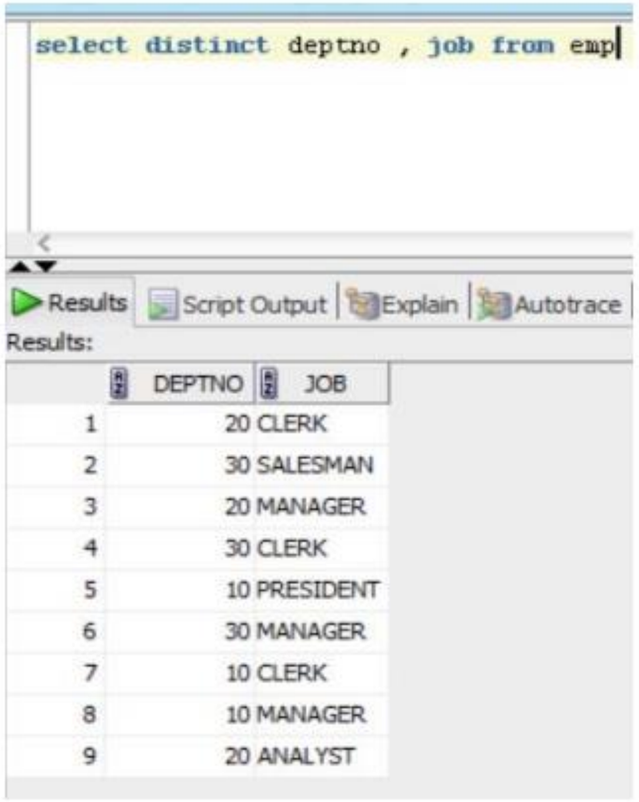
```
select deptno from emp
```

```
Results:
      DEPTNO
 1        20
 2        30
 3        30
 4        20
 5        30
 6        30
 7        10
 8        20
 9        10
10        30
11        20
12        30
13        20
14        10
```

## EXAMPLE H:

```
SELECT DISTINCT deptno, job
FROM    emp ;
```

**OUTPUT:**



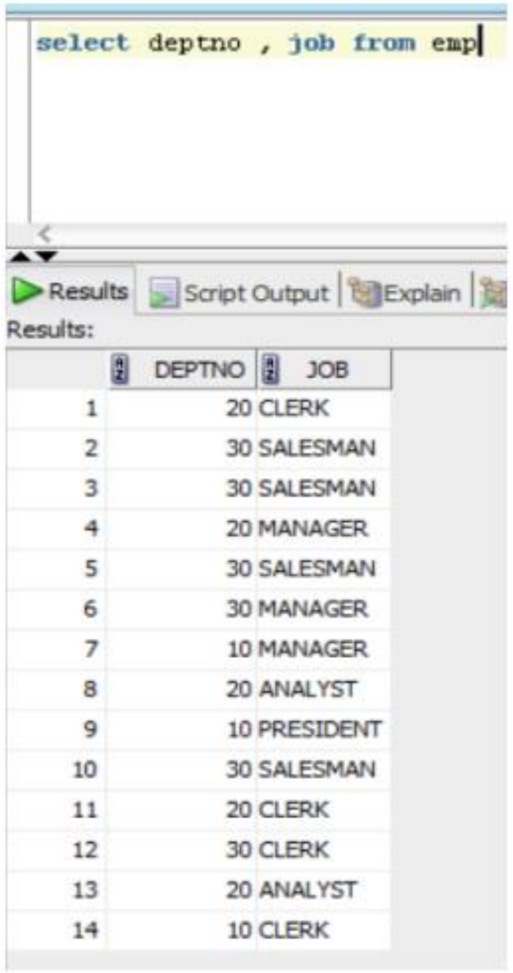
```
select distinct deptno , job from emp
```

```
      DEPTNO  JOB
  1       20  CLERK
  2       30  SALESMAN
  3       20  MANAGER
  4       30  CLERK
  5       10  PRESIDENT
  6       30  MANAGER
  7       10  CLERK
  8       10  MANAGER
  9       20  ANALYST
```

## EXAMPLE I:

```
SELECT deptno, job
FROM    emp ;
```

**OUTPUT:**



```
select deptno , job from emp
```

```
      DEPTNO  JOB
  1       20  CLERK
  2       30  SALESMAN
  3       30  SALESMAN
  4       20  MANAGER
  5       30  SALESMAN
  6       30  MANAGER
  7       10  MANAGER
  8       20  ANALYST
  9       10  PRESIDENT
 10       30  SALESMAN
 11       20  CLERK
 12       30  CLERK
 13       20  ANALYST
 14       10  CLERK
```

## CONCATENATION

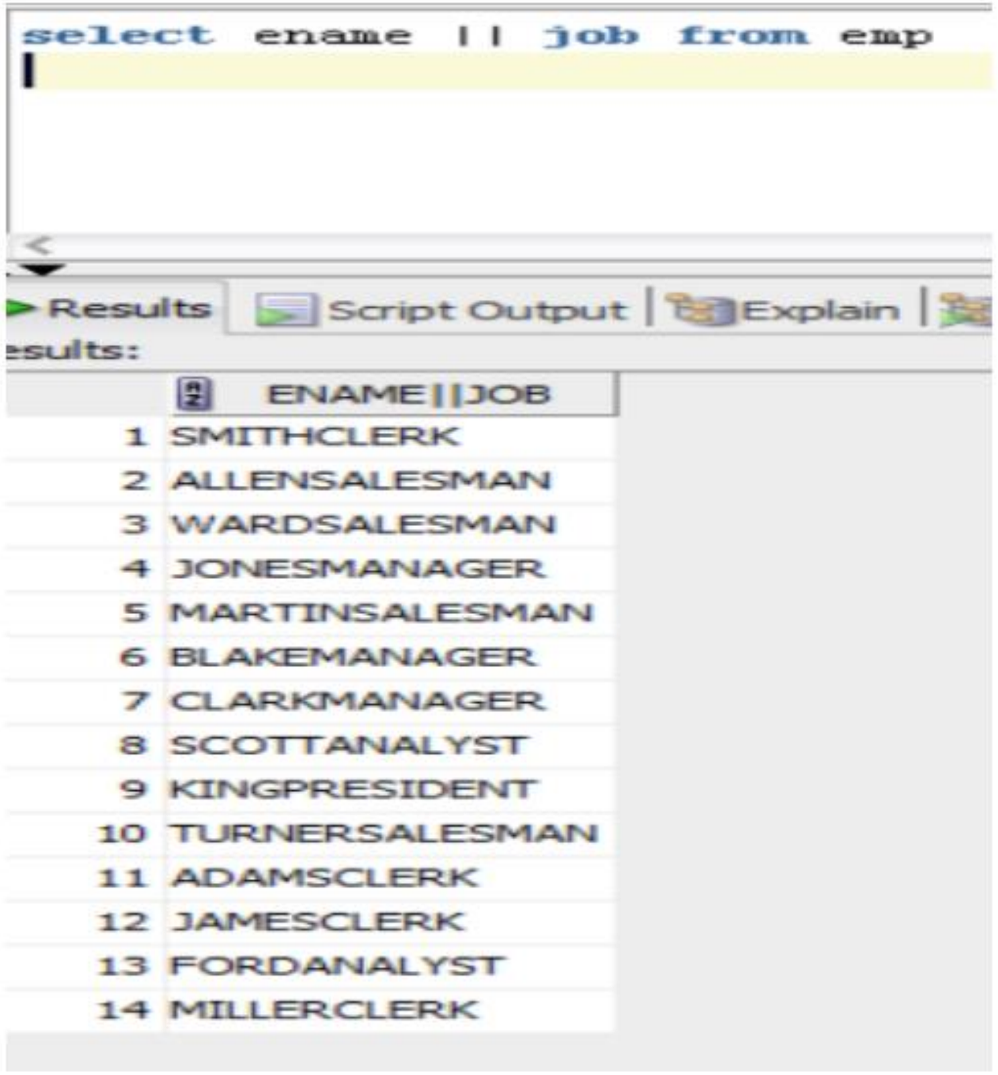
You can add your own statements in the output using **CONCAT** function or operators like **||** .

**EXAMPLE J:**

SELECT ename || job

FROM emp ;

**OUTPUT:**



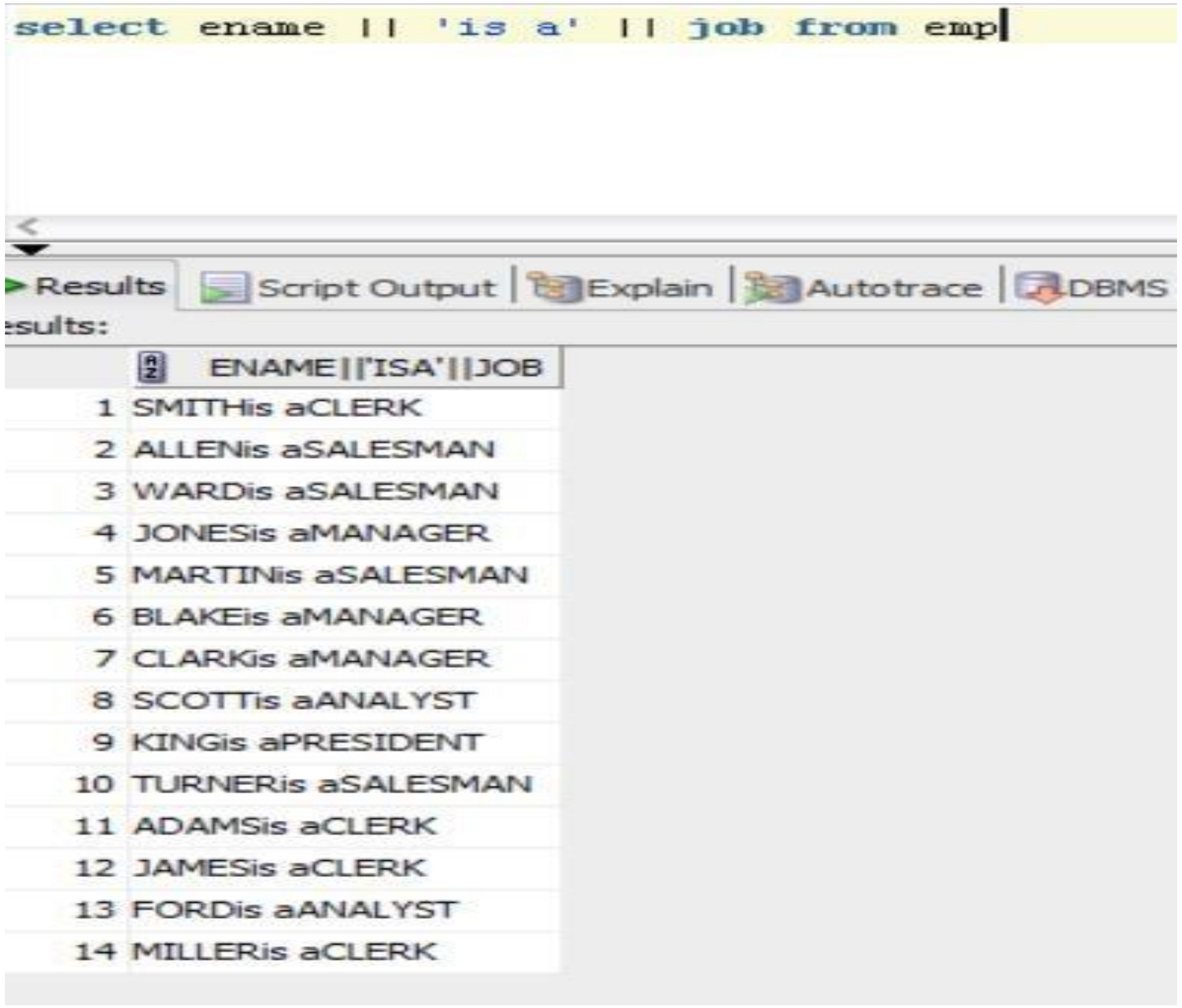
```
select ename || job from emp
```

```
   ENAME||JOB
 1 SMITHCLERK
 2 ALLENSALESMAN
 3 WARDSALESMAN
 4 JONESMANAGER
 5 MARTINSALESMAN
 6 BLAKEMANAGER
 7 CLARKMANAGER
 8 SCOTTANALYST
 9 KINGPRESIDENT
10 TURNERSALESMAN
11 ADAMSCLERK
12 JAMESCLERK
13 FORDANALYST
14 MILLERCLERK
```

## EXAMPLE K:

```
SELECT ename || ‘is a’ || job
FROM  emp ;
```

## OUTPUT:



```
select ename || 'is a' || job from emp
```

```
    ENAME||'ISA'||JOB
 1  SMITHis aCLERK
 2  ALLENis aSALESMAN
 3  WARDis aSALESMAN
 4  JONESis aMANAGER
 5  MARTINis aSALESMAN
 6  BLAKEis aMANAGER
 7  CLARKis aMANAGER
 8  SCOTTis aANALYST
 9  KINGis aPRESIDENT
10  TURNERis aSALESMAN
11  ADAMSis aCLERK
12  JAMESis aCLERK
13  FORDis aANALYST
14  MILLERis aCLERK
```

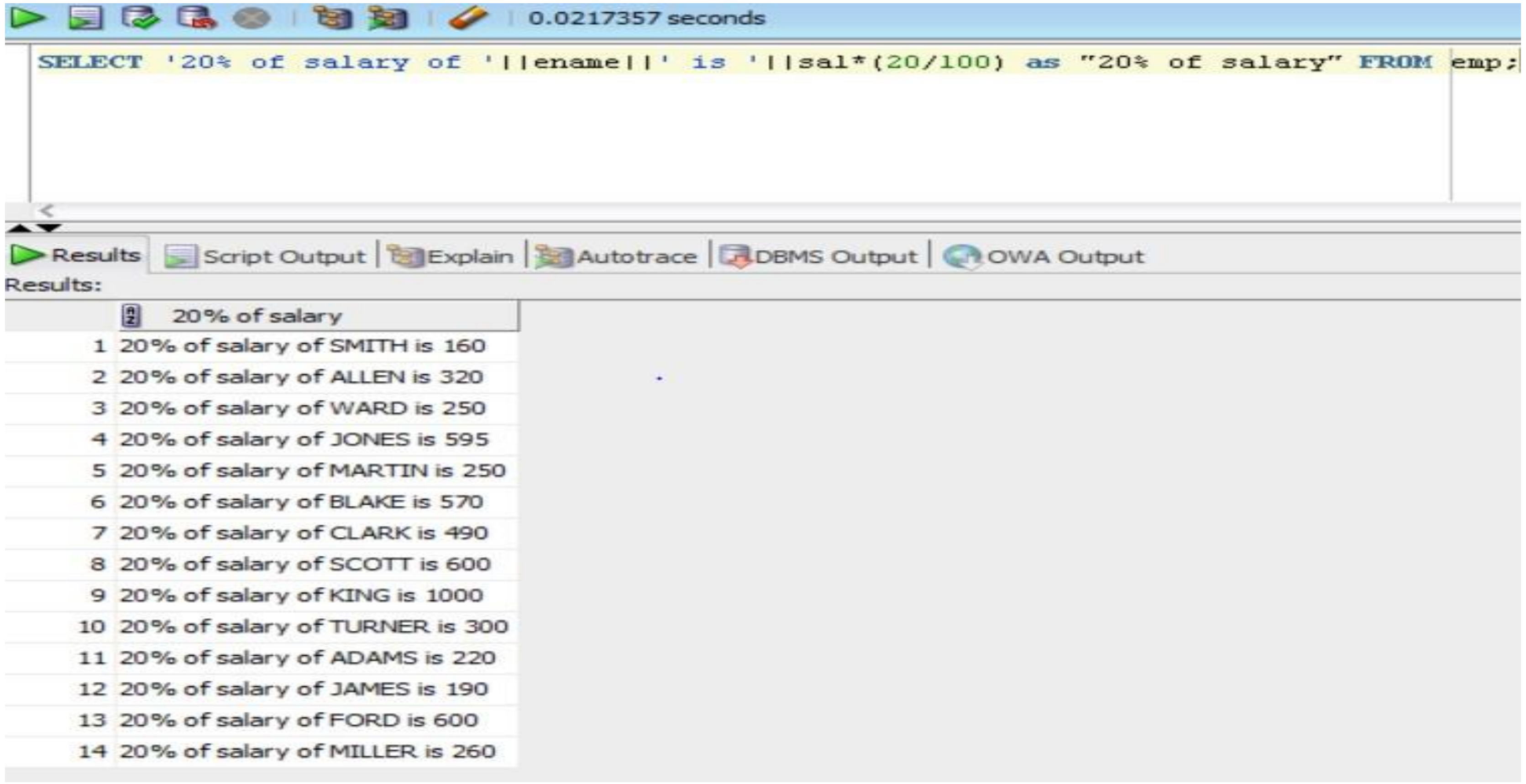
**EXAMPLE L:**

SELECT '20% of salary of '||ename||' is '||sal*(20/100) AS

"20% of salary"

FROM emp;

**OUTPUT:**



```
SELECT '20% of salary of '||ename||' is '||sal*(20/100) as "20% of salary" FROM emp;
```

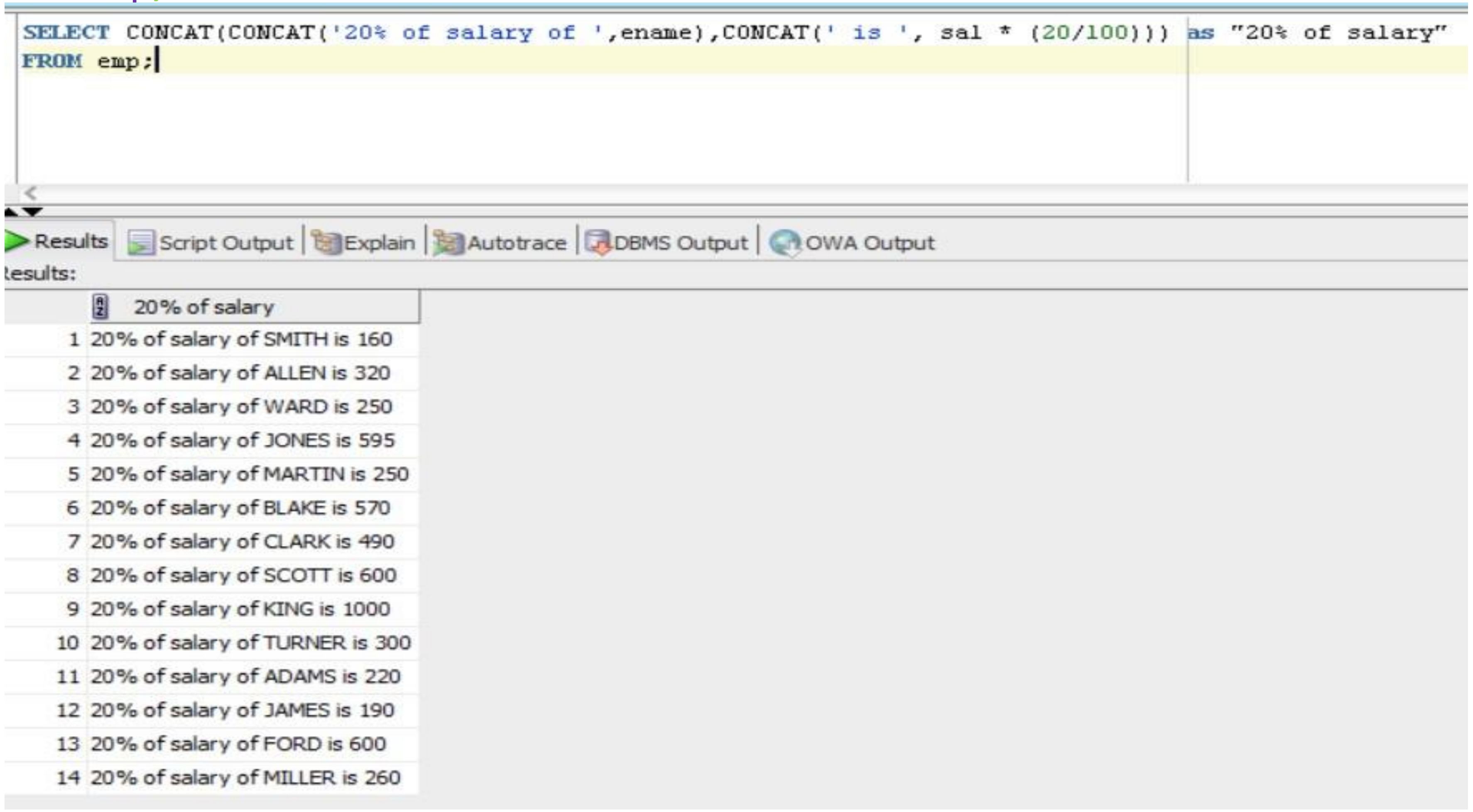
Results:

```
    20% of salary
 1  20% of salary of SMITH is 160
 2  20% of salary of ALLEN is 320
 3  20% of salary of WARD is 250
 4  20% of salary of JONES is 595
 5  20% of salary of MARTIN is 250
 6  20% of salary of BLAKE is 570
 7  20% of salary of CLARK is 490
 8  20% of salary of SCOTT is 600
 9  20% of salary of KING is 1000
10  20% of salary of TURNER is 300
11  20% of salary of ADAMS is 220
12  20% of salary of JAMES is 190
13  20% of salary of FORD is 600
14  20% of salary of MILLER is 260
```

**EXAMPLE M:**

SELECT CONCAT(CONCAT( '20% of salary of  ‘,ename), CONCAT(’ is ‘, sal*(20/100) )) AS "20% of salary"

FROM emp; **OUTPUT:**



```
SELECT CONCAT(CONCAT('20% of salary of ',ename),CONCAT(' is ', sal * (20/100))) as "20% of salary"
FROM emp;
```

```
    20% of salary
 1  20% of salary of SMITH is 160
 2  20% of salary of ALLEN is 320
 3  20% of salary of WARD is 250
 4  20% of salary of JONES is 595
 5  20% of salary of MARTIN is 250
 6  20% of salary of BLAKE is 570
 7  20% of salary of CLARK is 490
 8  20% of salary of SCOTT is 600
 9  20% of salary of KING is 1000
10  20% of salary of TURNER is 300
11  20% of salary of ADAMS is 220
12  20% of salary of JAMES is 190
13  20% of salary of FORD is 600
14  20% of salary of MILLER is 260
```

# TASK B

1. **Find errors:**

```
Select empno, ename sal x 12 Annual Salary
          From emp;
```

2. **Display employee's annual salary with one time bonus of $100.**
3. **Display annual compensation as monthly salary plus a monthly bonus of $100.**
4. **Display rows in following format:**

```
Monthly Salary
King: 1 month Salary = 5000
```

5. **Display kinds of Jobs available in employee table.**

# COMPARISION OPERATORS

## 1. Mathematical Operators

= > < < > or != or ^= <= >=

## 2. Logical Operators

NOT

AND

OR

## 3. Conditional Operators

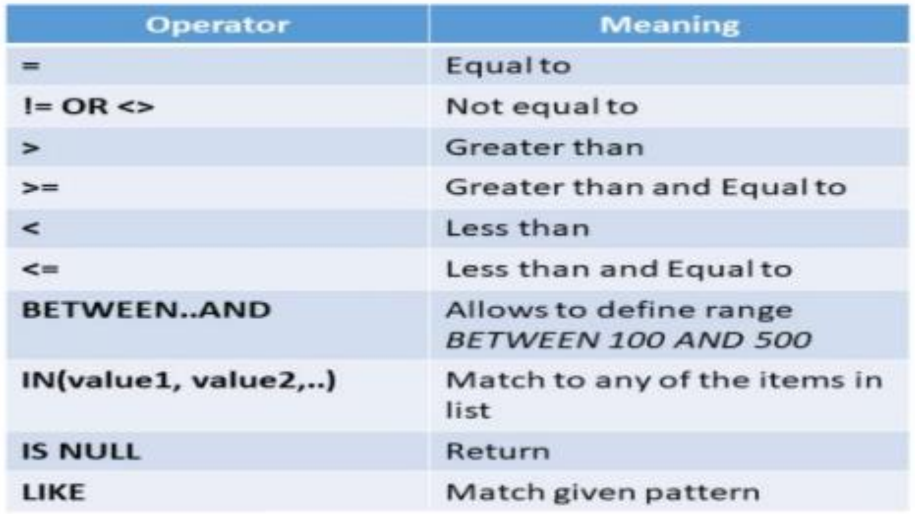
[ NOT ] BETWEEN lowerlimit AND upperlimit

[ NOT ] LIKE (Character Pattern )

[ NOT ] IN (x,y,z………)

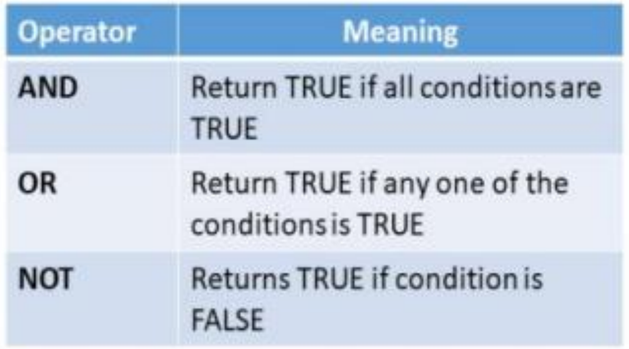
IS [ NOT ] NULL

**Conditional Operators**



| Operator | Meaning |
|---|---|
| = | Equal to |
| != OR <> | Not equal to |
| > | Greater than |
| >= | Greater than and Equal to |
| < | Less than |
| <= | Less than and Equal to |
| BETWEEN..AND | Allows to define range *BETWEEN 100 AND 500* |
| IN(value1, value2,..) | Match to any of the items in list |
| IS NULL | Return |
| LIKE | Match given pattern |

**Logical Conditional Operators**



| Operator | Meaning |
|---|---|
| AND | Return TRUE if all conditions are TRUE |
| OR | Return TRUE if any one of the conditions is TRUE |
| NOT | Returns TRUE if condition is FALSE |

# OPERATOR PRECEDENCE

1. Mathematical Operators
2. Logical Operators
   - NOT
   - AND
   - OR

# WHERE CLAUSE

**SYNTAX:**

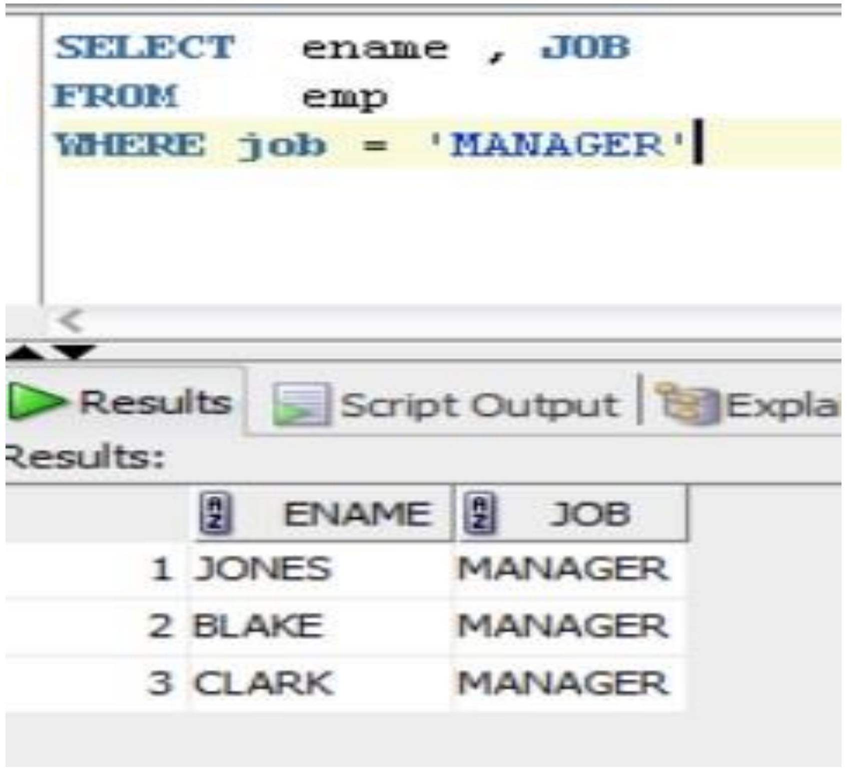
```
SELECT * I [ DISTINCT I UNIQUE] (column_ name [ AS alias ] ,arithmetic expr)
 FROM                              table _ name  [,...............]
 [ WHERE                          condition       ] ;
```

- **WHERE** clause is used to restrict rows in the output of the query.
- Only rows which meet the **WHERE** clause condition are displayed in the output.
- **WHERE** clause can be used to filter the records and fetching only the necessary records.
- The **WHERE** clause is not only used in the SELECT statement, but it is also used in the UPDATE, DELETE statement, etc.

**EXAMPLE L:**

```
SELECT ename, job
FROM  emp
WHERE job = 'MANAGER' ;
```

**OUTPUT:**



```
SELECT   ename , JOB
FROM     emp
WHERE job = 'MANAGER'
```

```
Results:
     ENAME    JOB
   1 JONES    MANAGER
   2 BLAKE    MANAGER
   3 CLARK    MANAGER
```

1. SELECT ename , job
   FROM emp
   WHERE job = 'manager’;

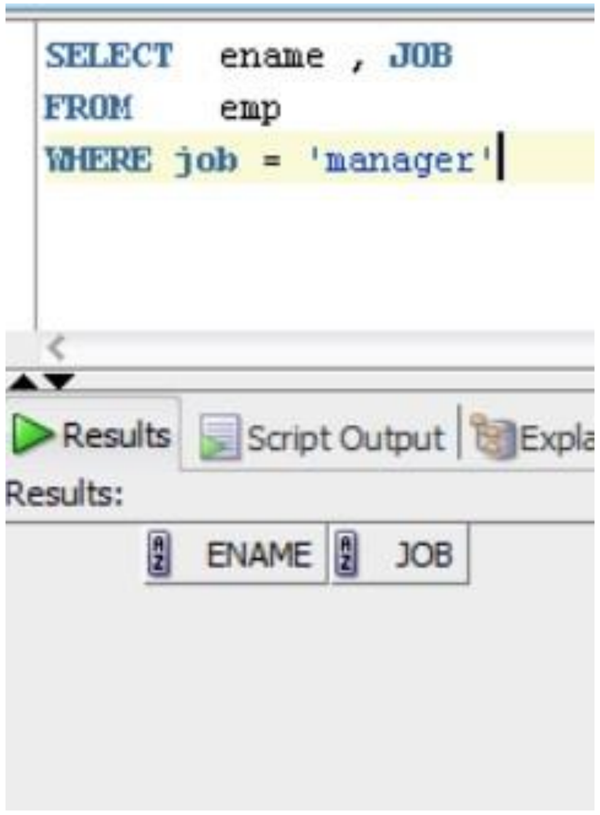
✓

2. SELECT ename , JOB
   FROM emp
   WHERE job = MANAGER;

✗

3. SELECT ename , JOB
   FROM emp
   WHERE job = "MANAGER"

✗

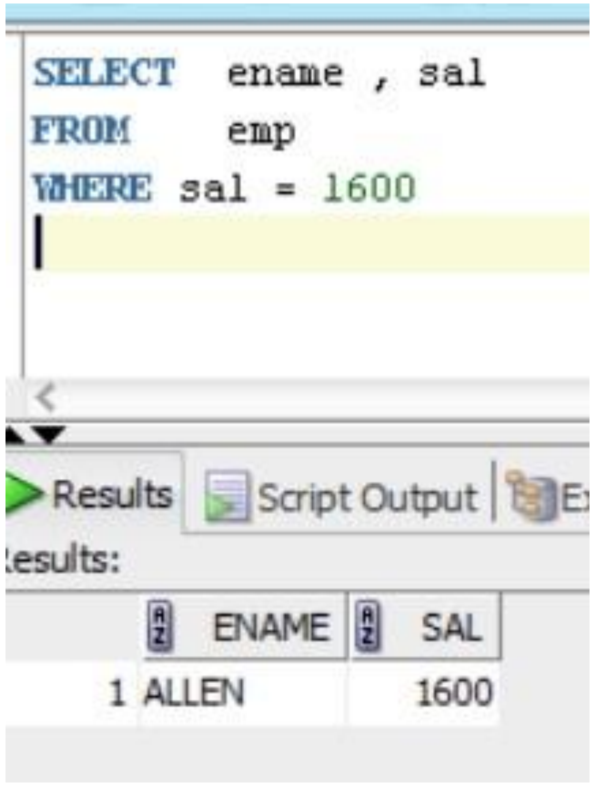


```
SELECT   ename , JOB
FROM     emp
WHERE job = 'manager'
```

## EXAMPLE M:

```
SELECT ename, sal
FROM  emp
WHERE sal = 1600 ;
```

**OUTPUT:**

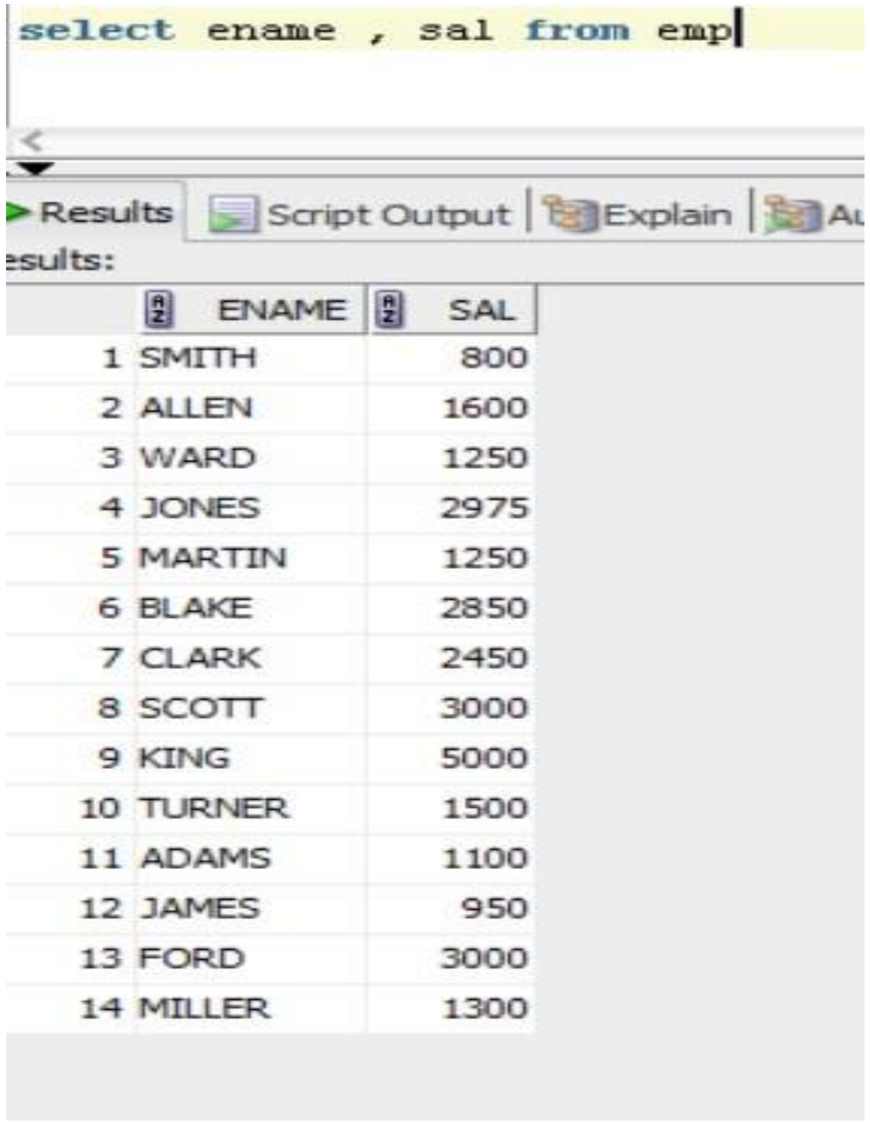


```
SELECT   ename , sal
FROM     emp
WHERE sal = 1600
```

Results | Script Output | E

Results:

|   | ENAME | SAL |
|---|---|---|
| 1 | ALLEN | 1600 |



```
select ename , sal from emp
```

Results | Script Output | Explain | Au

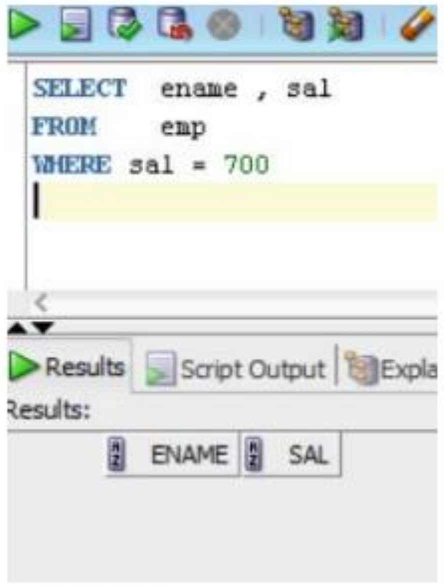
esults:

|   | ENAME | SAL |
|---|---|---|
| 1 | SMITH | 800 |
| 2 | ALLEN | 1600 |
| 3 | WARD | 1250 |
| 4 | JONES | 2975 |
| 5 | MARTIN | 1250 |
| 6 | BLAKE | 2850 |
| 7 | CLARK | 2450 |
| 8 | SCOTT | 3000 |
| 9 | KING | 5000 |
| 10 | TURNER | 1500 |
| 11 | ADAMS | 1100 |
| 12 | JAMES | 950 |
| 13 | FORD | 3000 |
| 14 | MILLER | 1300 |

**EXAMPLE N:**

```
SELECT ename, sal
FROM   emp
WHERE sal = 700 ;
```

**OUTPUT N:**



```
SELECT  ename , sal
FROM    emp
WHERE sal = 700
```

Results | Script Output | Expla

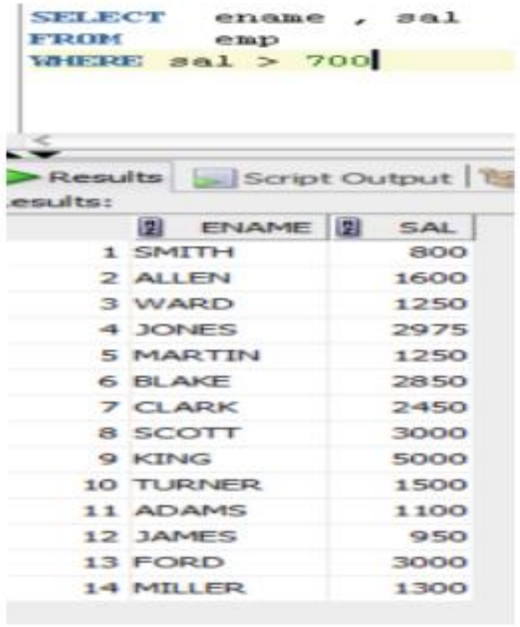
Results:

```
ENAME  SAL
```

**EXAMPLE O:**

```
SELECT ename, sal
FROM   emp
WHERE sal > 700 ;
```

**OUTPUT O:**



```
SELECT   ename , sal
FROM     emp
WHERE sal > 700
```

Results | Script Output

esults:

```
     ENAME     SAL
  1  SMITH     800
  2  ALLEN    1600
  3  WARD     1250
  4  JONES    2975
  5  MARTIN   1250
  6  BLAKE    2850
  7  CLARK    2450
  8  SCOTT    3000
  9  KING     5000
 10  TURNER   1500
 11  ADAMS    1100
 12  JAMES     950
 13  FORD     3000
 14  MILLER   1300
```

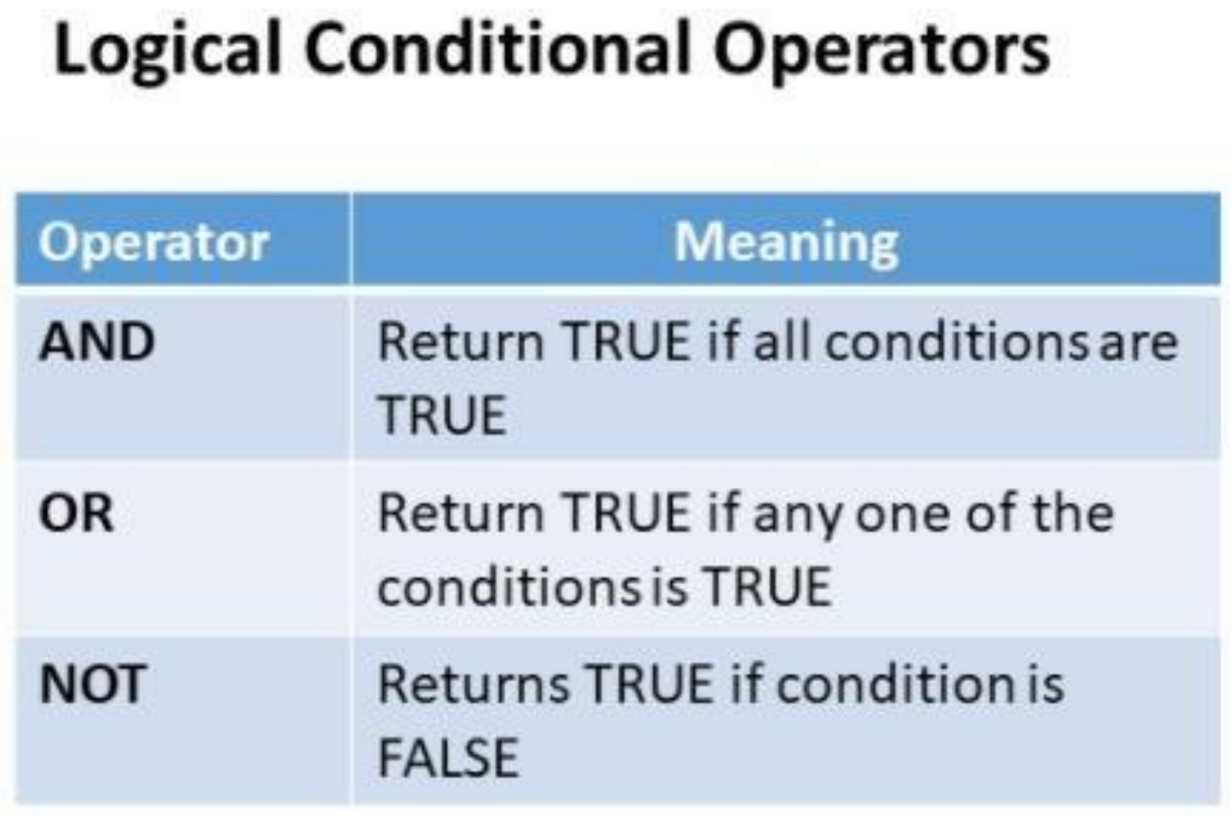
# LOGICAL OPERATORS

**SYNTAX (AND):**

```
SELECT column1, column2,..
FROM table_name
WHERE condition1 AND condition2 ;
```

**SYNTAX (OR):**

```
SELECT column1, column2,..
FROM table_name
WHERE condition1 OR condition2 ;
```



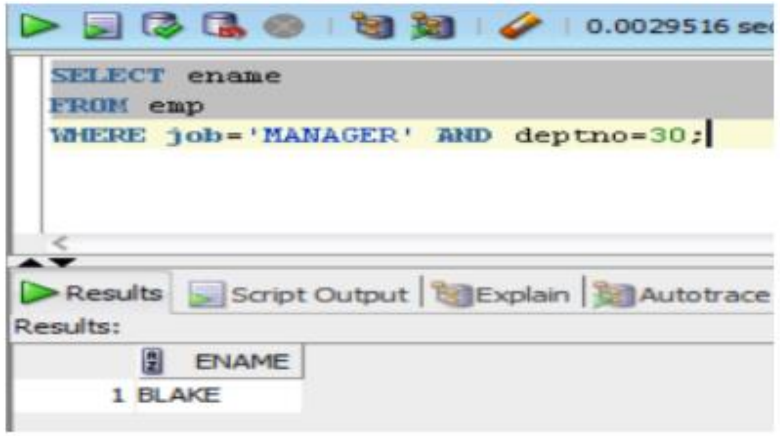
**Logical Conditional Operators**

| Operator | Meaning |
|---|---|
| AND | Return TRUE if all conditions are TRUE |
| OR | Return TRUE if any one of the conditions is TRUE |
| NOT | Returns TRUE if condition is FALSE |

## EXAMPLE O:

Find names of employees whose job is MANAGER and belong to department 30.

```
SELECT ename
FROM emp
WHERE job = 'MANAGER' AND deptno = 30 ;
```



```
SELECT ename
FROM emp
WHERE job='MANAGER' AND deptno=30;
```

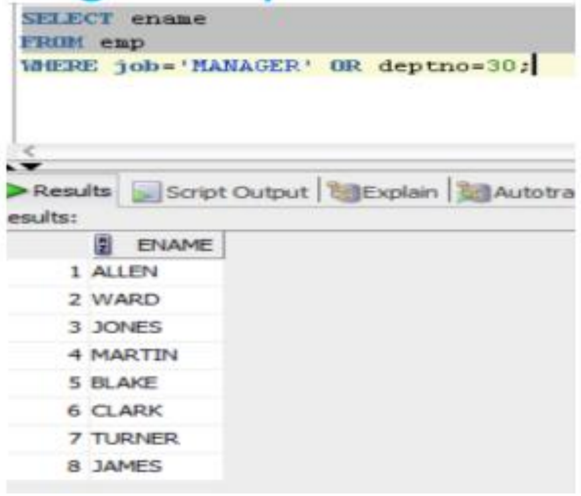
Results:

```
  ENAME
1 BLAKE
```

## EXAMPLE P:

Find names of employees whose job is MANAGER or belong to department 30.

```
SELECT ename
FROM emp
WHERE job = 'MANAGER' OR deptno = 30 ;
```



```
SELECT ename
FROM emp
WHERE job='MANAGER' OR deptno=30;
```

esults:

```
  ENAME
1 ALLEN
2 WARD
3 JONES
4 MARTIN
5 BLAKE
6 CLARK
7 TURNER
8 JAMES
```

**SYNTAX (NOT):**

SELECT column1 , column2, ...

FROM table_name WHERE NOT condition ; **EXAMPLE**

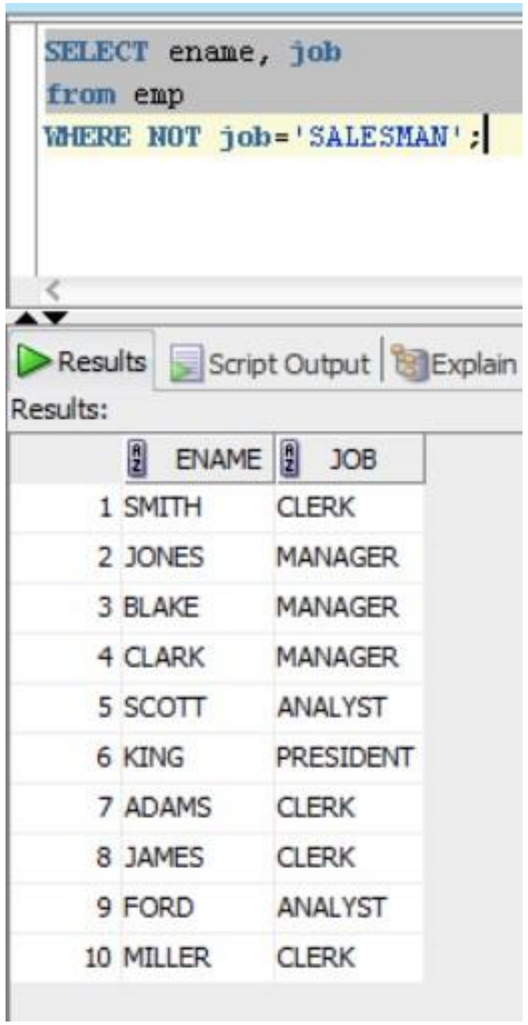
**Q:**

Find all the employees whose job is not SALESMAN

SELECT ename , job

FROM emp

WHERE NOT job = 'SALESMAN';



```
SELECT ename, job
from emp
WHERE NOT job='SALESMAN';
```

Results | Script Output | Explain

Results:

|  | ENAME | JOB |
|---|---|---|
| 1 | SMITH | CLERK |
| 2 | JONES | MANAGER |
| 3 | BLAKE | MANAGER |
| 4 | CLARK | MANAGER |
| 5 | SCOTT | ANALYST |
| 6 | KING | PRESIDENT |
| 7 | ADAMS | CLERK |
| 8 | JAMES | CLERK |
| 9 | FORD | ANALYST |
| 10 | MILLER | CLERK |

# TASK C

- **Find all employees whose job is not CLERK and belong to department 20.**

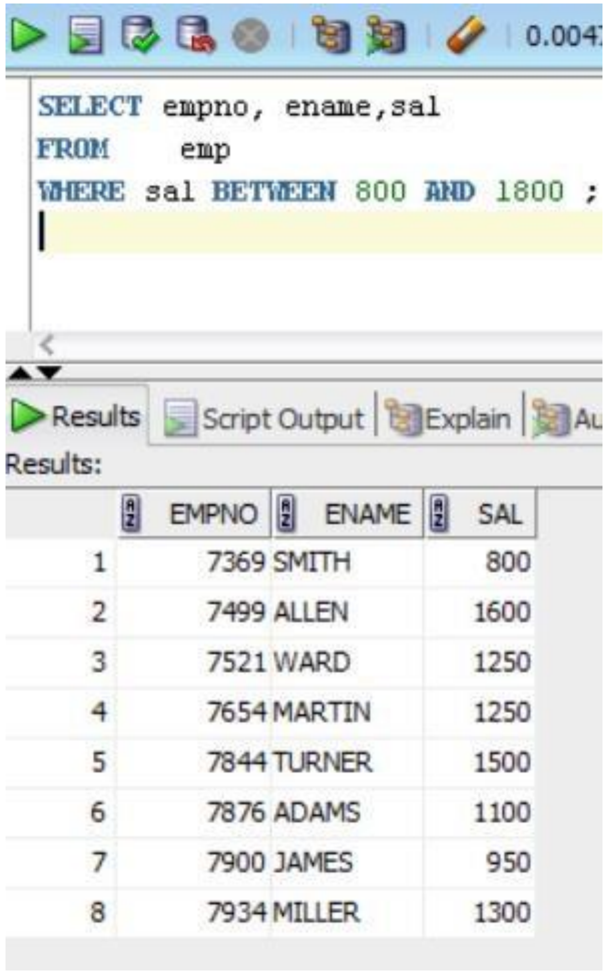
- **Display the employee's name , job title & salary based on the following criteria:**
  a) If the employee is a salesman, then he should be included in the O/P
  b) If the employee is a manager, then his salary package must be above 2450.

- **Display employee's name , Job titles & salary if the employee is either a salesman or a manager & earns more than 2450.**

[NOT] BETWEEN lower_range AND upper_range

## EXAMPLE R:

```
SELECT empno, ename, sal
FROM  emp
WHERE sal BETWEEN 800 AND 1800 ;
```



```
SELECT empno, ename,sal
FROM    emp
WHERE sal BETWEEN 800 AND 1800 ;
```

Results:

| | EMPNO | ENAME | SAL |
|---|---|---|---|
| 1 | 7369 | SMITH | 800 |
| 2 | 7499 | ALLEN | 1600 |
| 3 | 7521 | WARD | 1250 |
| 4 | 7654 | MARTIN | 1250 |
| 5 | 7844 | TURNER | 1500 |
| 6 | 7876 | ADAMS | 1100 |
| 7 | 7900 | JAMES | 950 |
| 8 | 7934 | MILLER | 1300 |

## [NOT] LIKE (CHARACTER PATTERN)

LIKE uses two wildcards such as percentage (%) and underscore (_) to represent the number of characters in the pattern.

Patterns are case-sensitive.

**% means any zero, one, or multiple characters**

- %M% Match any string having M in any position
- M% Match value having M at start
- %M Match value having M at end
- M%A Start with M and end with A

**_ specifies the number of unknown characters before or after the known character. One underscore is one character.**

- _r% Match value having r in the second position

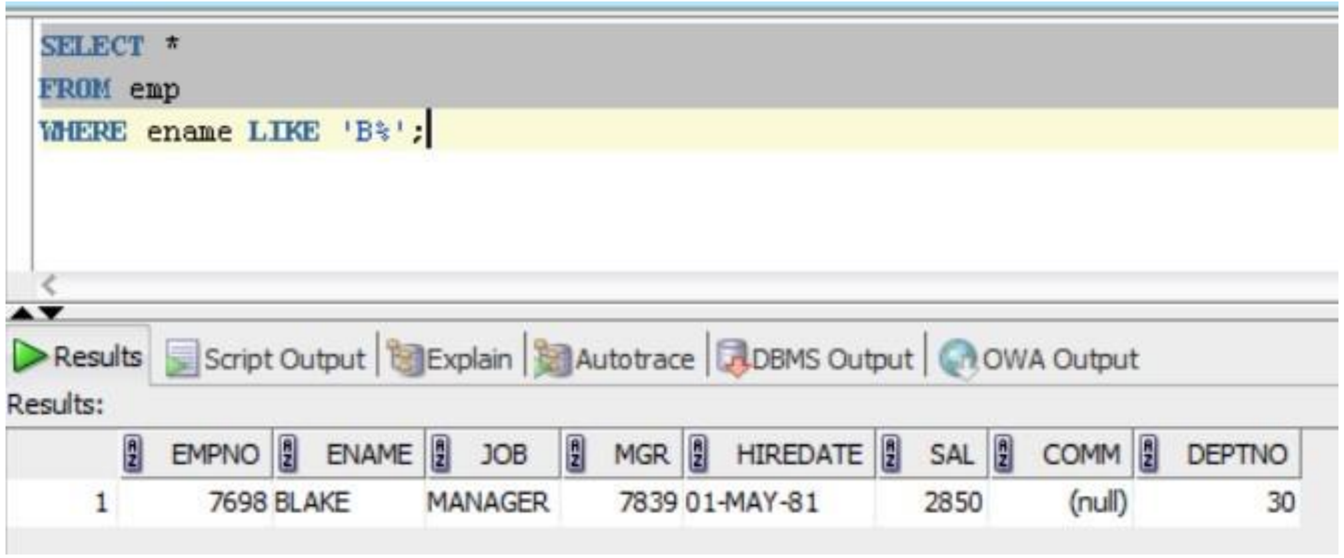
## EXAMPLE S:

Get names of all employees whose names start with 'B’.

SELECT *

FROM emp

WHERE ename LIKE ‘B%’ ;



```
SELECT *
FROM emp
WHERE ename LIKE 'B%';
```

Results:

```
     EMPNO  ENAME    JOB       MGR   HIREDATE   SAL   COMM    DEPTNO
1     7698  BLAKE    MANAGER   7839  01-MAY-81  2850  (null)      30
```

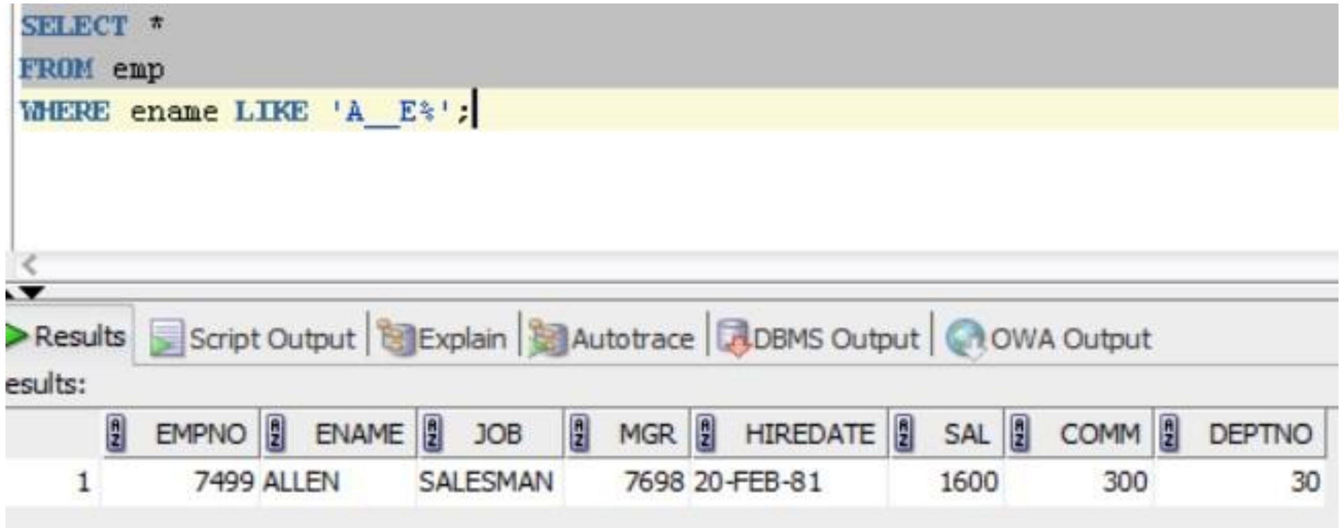
## EXAMPLE T:

Get names of all employees whose names start with an 'A' and has 'E' in the fourth position.

**SELECT ***

**FROM** emp

**WHERE ename LIKE 'A __ E%' ;**



```
SELECT *
FROM emp
WHERE ename LIKE 'A__E%';
```

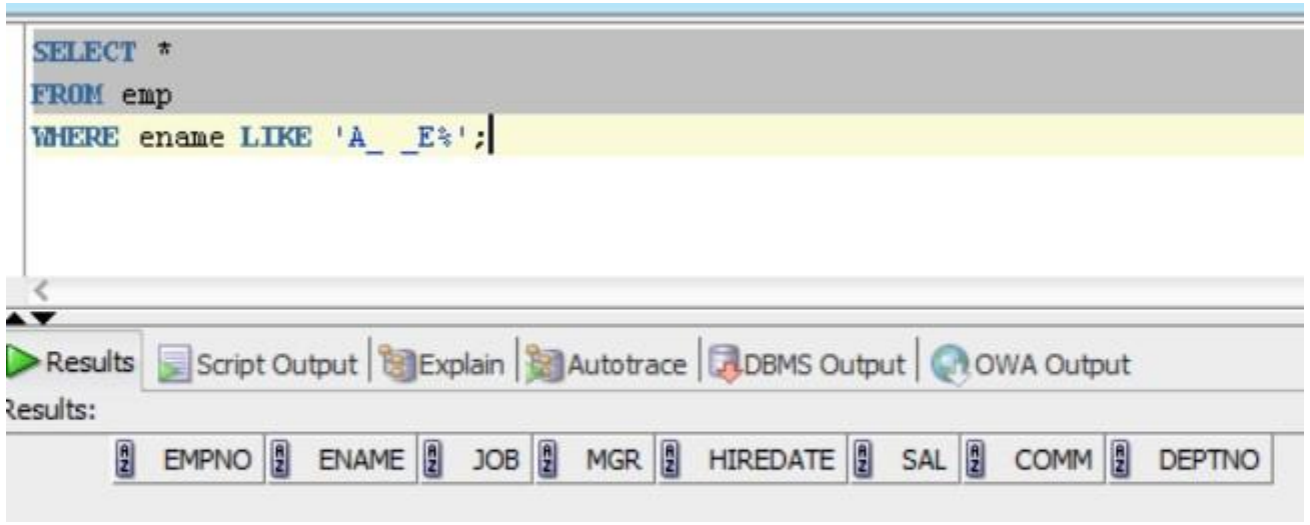
| | EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|---|---|---|---|---|---|---|---|---|
| 1 | 7499 | ALLEN | SALESMAN | 7698 | 20-FEB-81 | 1600 | 300 | 30 |

## EXAMPLE U:

SELECT *

FROM emp

WHERE ename LIKE 'A _ _ E%'



```
SELECT *
FROM emp
WHERE ename LIKE 'A_ _E%';
```

Results:

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

## TASK D

- **List the employees having at least two A’s in their names.**
- **List the employees whose names start with S and end at H.**
- **List the employee whose name has E as the second character.**
- **Display employ number and job title of all employees who have a job title that contain the string ‘MAN’ & earn more than 10,000.**

## [NOT] IN(value1,value2, value3,..)

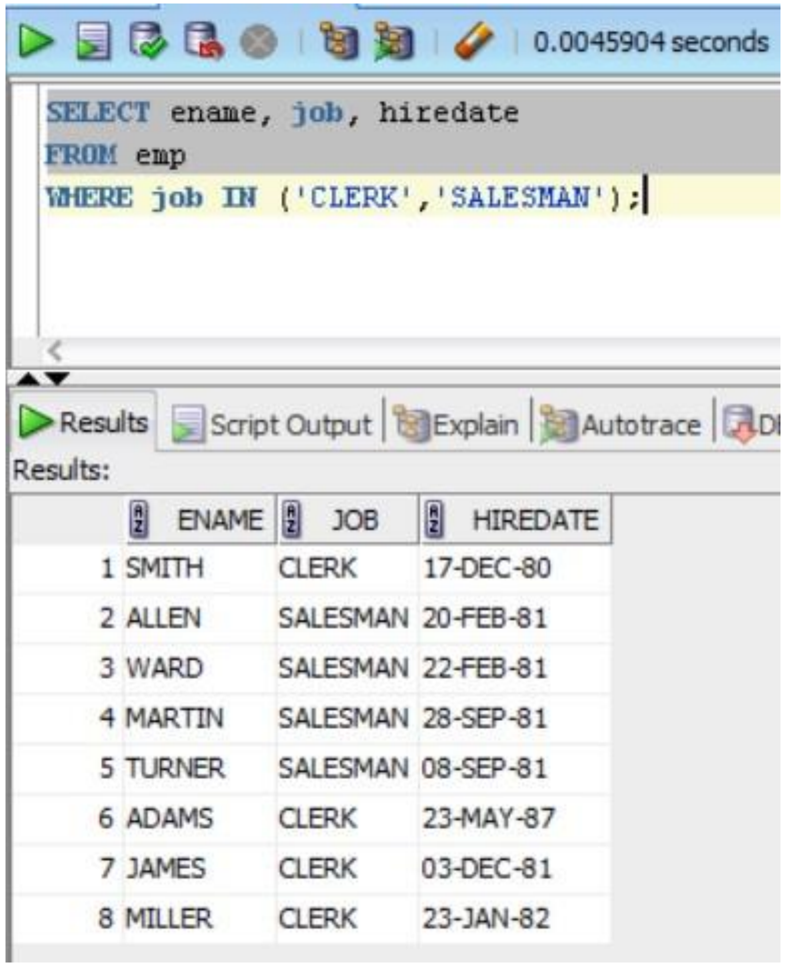
- The **IN** operator can take one, two or multiple values and allows you to match a column to the given values in parentheses in the WHERE clause.

## EXAMPLE V:

SELECT ename, job, hiredate

FROM emp

WHERE job IN ('CLERK','SALESMAN’) ;



```
SELECT ename, job, hiredate
FROM emp
WHERE job IN ('CLERK','SALESMAN');
```

```
   ENAME   JOB       HIREDATE
 1 SMITH   CLERK     17-DEC-80
 2 ALLEN   SALESMAN  20-FEB-81
 3 WARD    SALESMAN  22-FEB-81
 4 MARTIN  SALESMAN  28-SEP-81
 5 TURNER  SALESMAN  08-SEP-81
 6 ADAMS   CLERK     23-MAY-87
 7 JAMES   CLERK     03-DEC-81
 8 MILLER  CLERK     23-JAN-82
```

## TASK E

**Display list of employees who are either a clerk or an analyst & who do not earn 1000, 3000,5000.**

# IS [NOT] NULL

IS NULL is used to check for NULL values in a given attribute.

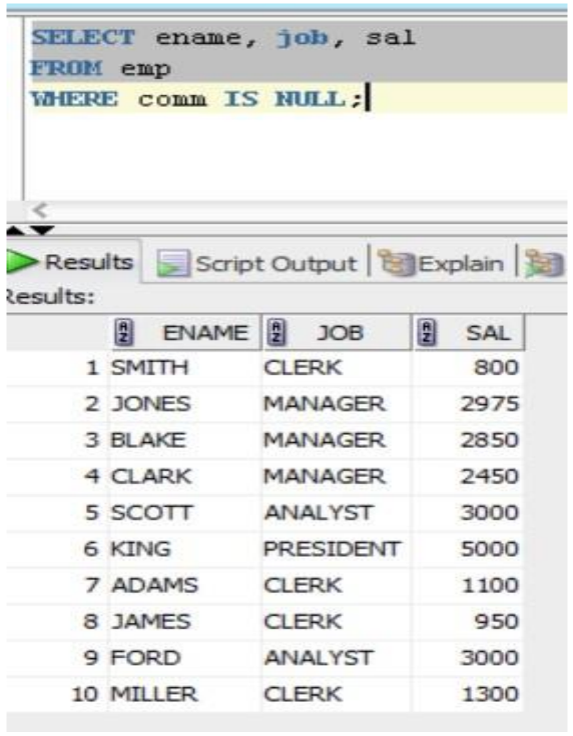
EXAMPLE W:

Find all employees who don't earn commission.

SELECT ename, job, sal

FROM emp

WHERE comm IS NULL ;



```
SELECT ename, job, sal
FROM emp
WHERE comm IS NULL;
```

Results:

| | ENAME | JOB | SAL |
|---|---|---|---|
| 1 | SMITH | CLERK | 800 |
| 2 | JONES | MANAGER | 2975 |
| 3 | BLAKE | MANAGER | 2850 |
| 4 | CLARK | MANAGER | 2450 |
| 5 | SCOTT | ANALYST | 3000 |
| 6 | KING | PRESIDENT | 5000 |
| 7 | ADAMS | CLERK | 1100 |
| 8 | JAMES | CLERK | 950 |
| 9 | FORD | ANALYST | 3000 |
| 10 | MILLER | CLERK | 1300 |

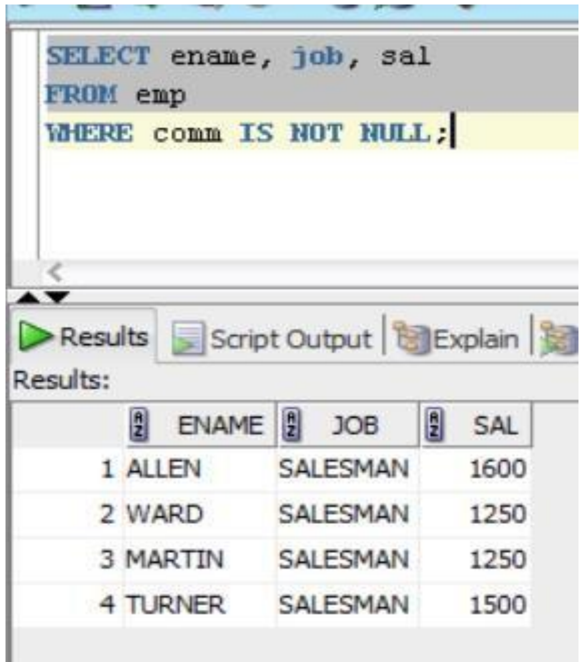
EXAMPLE X:

Find all employees who earn commission.

SELECT ename, job, sal

FROM emp

WHERE comm IS NOT NULL ;



```
SELECT ename, job, sal
FROM emp
WHERE comm IS NOT NULL;
```

Results:

| | ENAME | JOB | SAL |
|---|---|---|---|
| 1 | ALLEN | SALESMAN | 1600 |
| 2 | WARD | SALESMAN | 1250 |
| 3 | MARTIN | SALESMAN | 1250 |
| 4 | TURNER | SALESMAN | 1500 |

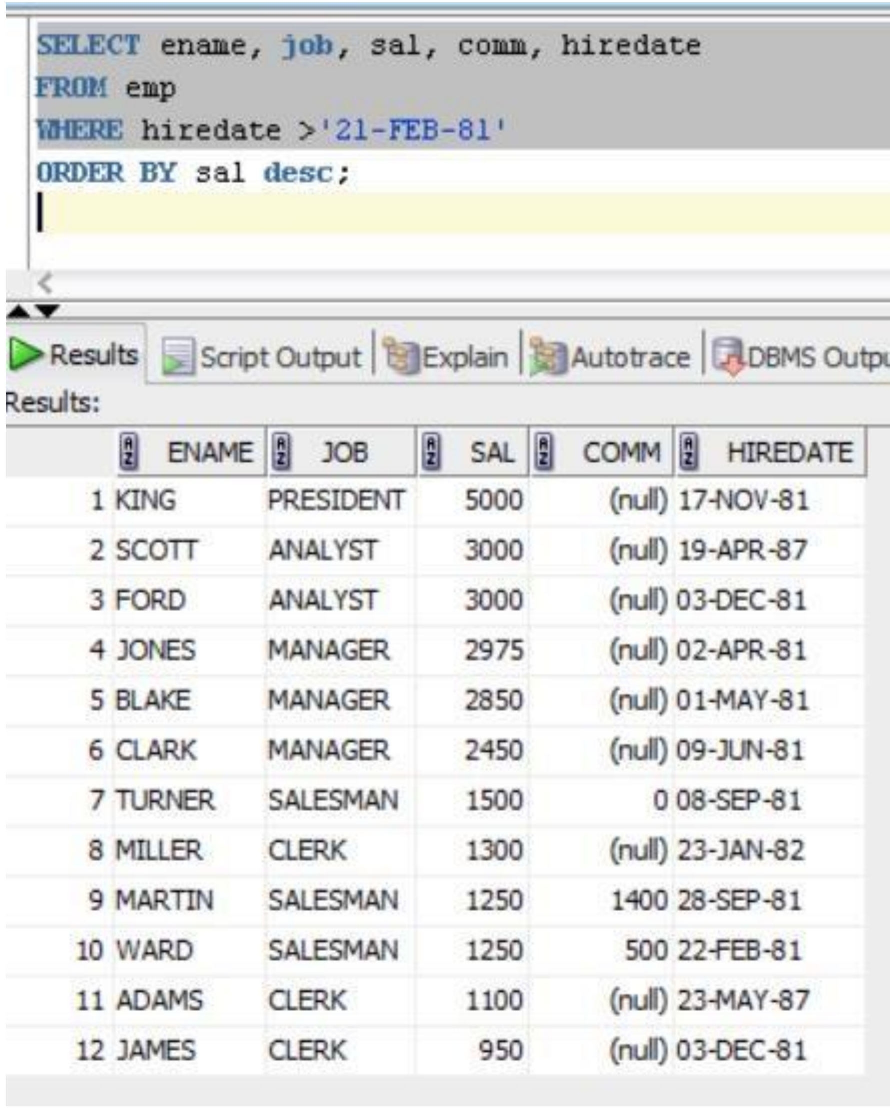
## ORDER BY CLAUSE

- **ORDER BY** clause is used for sorting the results of a query.
- Sorting can be done in ascending (ASC) or descending order (DESC).
- Default order is ascending **(0-10, A-Z, NULL in last)**.
- Descending order **(10-0, Z-A, NULL at first).**
- Sorting can be done using a single column or multiple columns.
- You can also order by aliases that you specify in SELECT clause.
- **ORDER BY** clause is always the last clause of the SELECT statement.

## EXAMPLE Y:

Find all employees who were hired after 21st Feb 1981 and display them based on highest pay scales.

```
SELECT ename, job, sal , comm , hiredate
FROM emp
WHERE hiredate > '21-FEB-81'
ORDER BY sal DESC ;
```



```
SELECT ename, job, sal, comm, hiredate
FROM emp
WHERE hiredate >'21-FEB-81'
ORDER BY sal desc;
```

Results:

| | ENAME | JOB | SAL | COMM | HIREDATE |
|---|---|---|---|---|---|
| 1 | KING | PRESIDENT | 5000 | (null) | 17-NOV-81 |
| 2 | SCOTT | ANALYST | 3000 | (null) | 19-APR-87 |
| 3 | FORD | ANALYST | 3000 | (null) | 03-DEC-81 |
| 4 | JONES | MANAGER | 2975 | (null) | 02-APR-81 |
| 5 | BLAKE | MANAGER | 2850 | (null) | 01-MAY-81 |
| 6 | CLARK | MANAGER | 2450 | (null) | 09-JUN-81 |
| 7 | TURNER | SALESMAN | 1500 | 0 | 08-SEP-81 |
| 8 | MILLER | CLERK | 1300 | (null) | 23-JAN-82 |
| 9 | MARTIN | SALESMAN | 1250 | 1400 | 28-SEP-81 |
| 10 | WARD | SALESMAN | 1250 | 500 | 22-FEB-81 |
| 11 | ADAMS | CLERK | 1100 | (null) | 23-MAY-87 |
| 12 | JAMES | CLERK | 950 | (null) | 03-DEC-81 |

**EXAMPLE Z:**

Sort employees first by their deptno in ascending order and then names in descending.

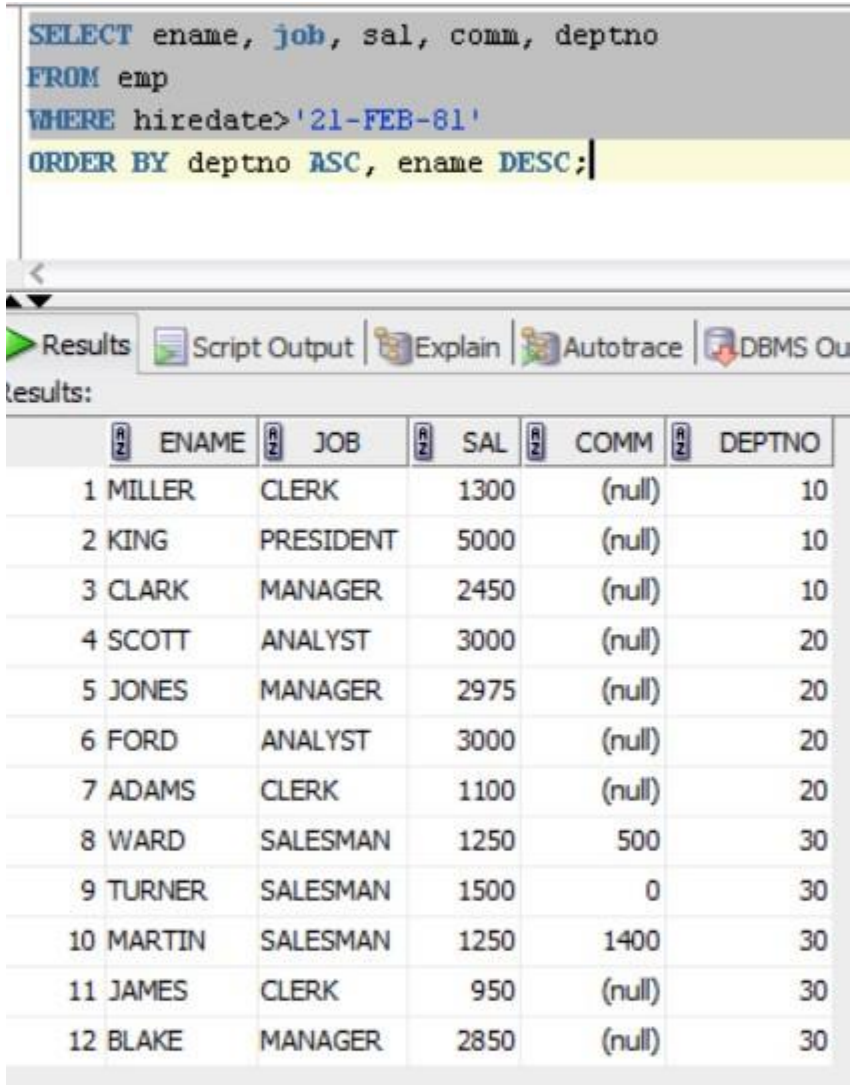
SELECT ename, job, sal , comm , deptno

FROM emp

WHERE hiredate > **’21-FEB-81’**

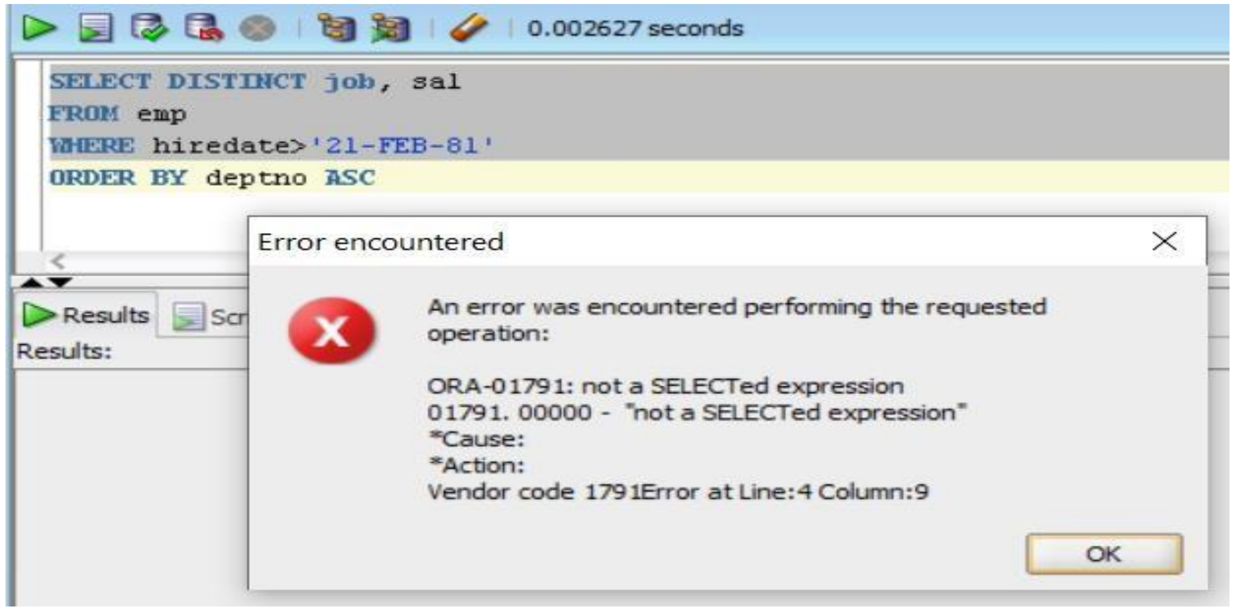
ORDER BY deptno ASC , ename DESC ;

- If multiple columns are listed in the order by clause, then the first listed column is called the PRIMARY SORT and the others are called SECONDARY SORT.
- Sort order applies to the column after which it was listed.
- If **UNIQUE** or **DISTINCT** is used, then sorting must be done with only those columns that are listed in the **SELECT** clause.



```
SELECT ename, job, sal, comm, deptno
FROM emp
WHERE hiredate>'21-FEB-81'
ORDER BY deptno ASC, ename DESC;
```

| | ENAME | JOB | SAL | COMM | DEPTNO |
|---|---|---|---|---|---|
| 1 | MILLER | CLERK | 1300 | (null) | 10 |
| 2 | KING | PRESIDENT | 5000 | (null) | 10 |
| 3 | CLARK | MANAGER | 2450 | (null) | 10 |
| 4 | SCOTT | ANALYST | 3000 | (null) | 20 |
| 5 | JONES | MANAGER | 2975 | (null) | 20 |
| 6 | FORD | ANALYST | 3000 | (null) | 20 |
| 7 | ADAMS | CLERK | 1100 | (null) | 20 |
| 8 | WARD | SALESMAN | 1250 | 500 | 30 |
| 9 | TURNER | SALESMAN | 1500 | 0 | 30 |
| 10 | MARTIN | SALESMAN | 1250 | 1400 | 30 |
| 11 | JAMES | CLERK | 950 | (null) | 30 |
| 12 | BLAKE | MANAGER | 2850 | (null) | 30 |

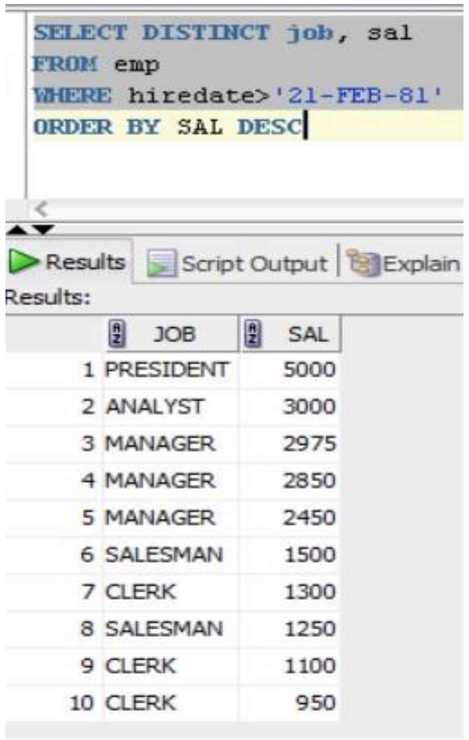


```
SELECT DISTINCT job, sal
FROM emp
WHERE hiredate>'21-FEB-81'
ORDER BY deptno ASC
```

Error encountered

An error was encountered performing the requested operation:

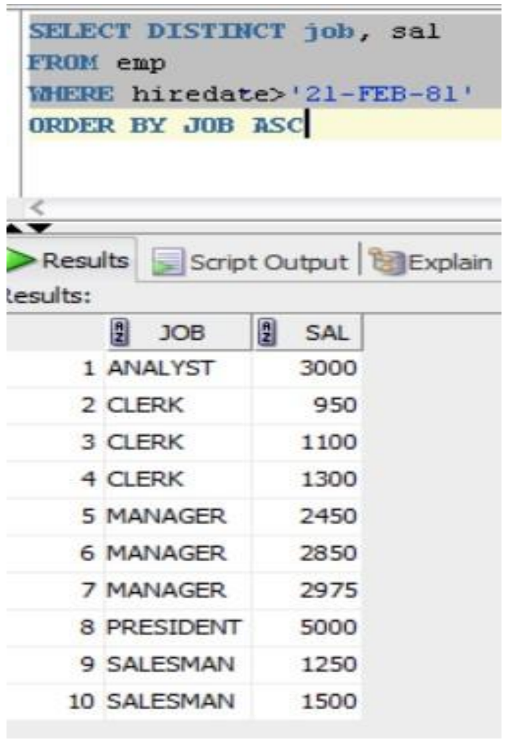
ORA-01791: not a SELECTed expression
01791. 00000 - "not a SELECTed expression"
*Cause:
*Action:
Vendor code 1791Error at Line:4 Column:9



```
SELECT DISTINCT job, sal
FROM emp
WHERE hiredate>'21-FEB-81'
ORDER BY SAL DESC
```

Results:

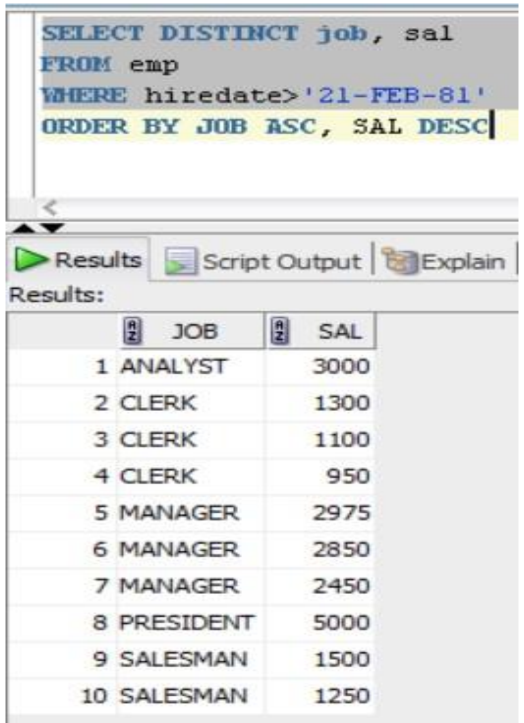
| | JOB | SAL |
|---|---|---|
| 1 | PRESIDENT | 5000 |
| 2 | ANALYST | 3000 |
| 3 | MANAGER | 2975 |
| 4 | MANAGER | 2850 |
| 5 | MANAGER | 2450 |
| 6 | SALESMAN | 1500 |
| 7 | CLERK | 1300 |
| 8 | SALESMAN | 1250 |
| 9 | CLERK | 1100 |
| 10 | CLERK | 950 |



```
SELECT DISTINCT job, sal
FROM emp
WHERE hiredate>'21-FEB-81'
ORDER BY JOB ASC
```

esults:

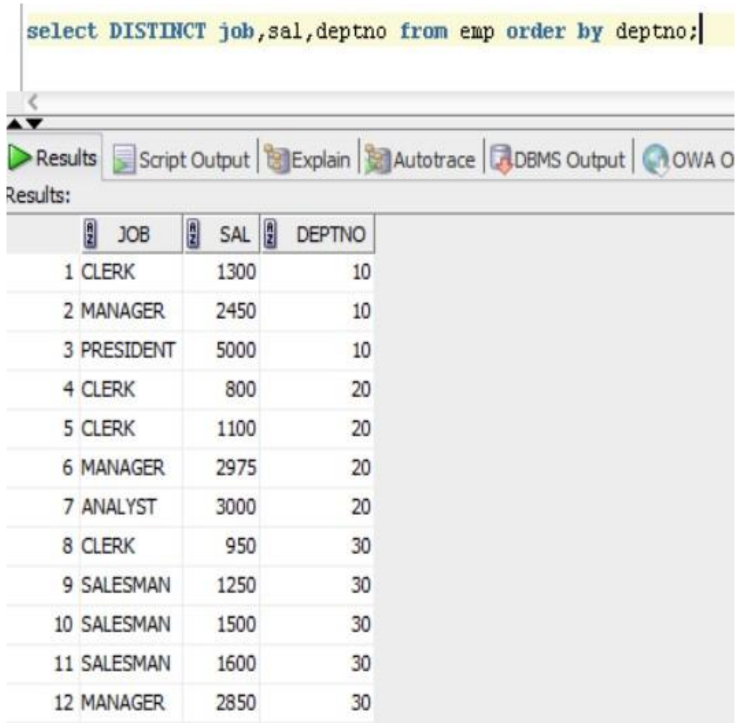
| | JOB | SAL |
|---|---|---|
| 1 | ANALYST | 3000 |
| 2 | CLERK | 950 |
| 3 | CLERK | 1100 |
| 4 | CLERK | 1300 |
| 5 | MANAGER | 2450 |
| 6 | MANAGER | 2850 |
| 7 | MANAGER | 2975 |
| 8 | PRESIDENT | 5000 |
| 9 | SALESMAN | 1250 |
| 10 | SALESMAN | 1500 |



```
SELECT DISTINCT job, sal
FROM emp
WHERE hiredate>'21-FEB-81'
ORDER BY JOB ASC, SAL DESC
```

Results:

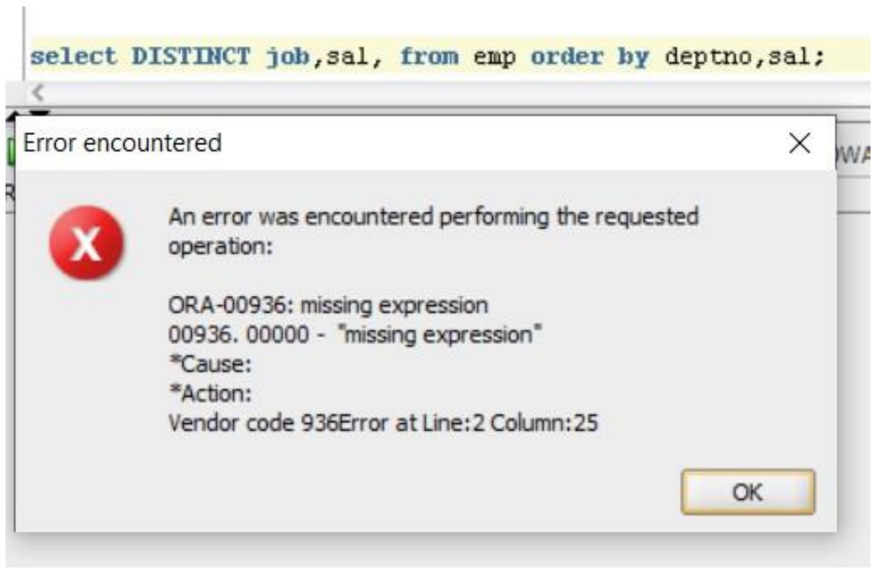
| | JOB | SAL |
|---|---|---|
| 1 | ANALYST | 3000 |
| 2 | CLERK | 1300 |
| 3 | CLERK | 1100 |
| 4 | CLERK | 950 |
| 5 | MANAGER | 2975 |
| 6 | MANAGER | 2850 |
| 7 | MANAGER | 2450 |
| 8 | PRESIDENT | 5000 |
| 9 | SALESMAN | 1500 |
| 10 | SALESMAN | 1250 |



```
select DISTINCT job,sal,deptno from emp order by deptno;
```

Results:

| | JOB | SAL | DEPTNO |
|---|---|---|---|
| 1 | CLERK | 1300 | 10 |
| 2 | MANAGER | 2450 | 10 |
| 3 | PRESIDENT | 5000 | 10 |
| 4 | CLERK | 800 | 20 |
| 5 | CLERK | 1100 | 20 |
| 6 | MANAGER | 2975 | 20 |
| 7 | ANALYST | 3000 | 20 |
| 8 | CLERK | 950 | 30 |
| 9 | SALESMAN | 1250 | 30 |
| 10 | SALESMAN | 1500 | 30 |
| 11 | SALESMAN | 1600 | 30 |
| 12 | MANAGER | 2850 | 30 |



```
select DISTINCT job,sal, from emp order by deptno,sal;
```

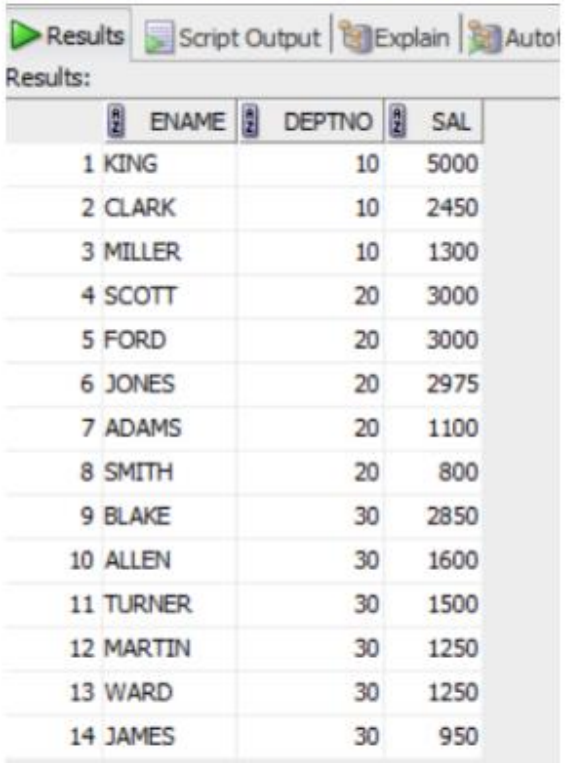
Error encountered

An error was encountered performing the requested operation:

ORA-00936: missing expression
00936. 00000 - "missing expression"
*Cause:
*Action:
Vendor code 936Error at Line:2 Column:25

## TASK F

- Display the names of employees according to their seniority.
- Display names and annual salary of all employees, also sort the result based on annual salary in descending order.
- Write a query which produces the following output.



Results:

|  | ENAME | DEPTNO | SAL |
|---|---|---|---|
| 1 | KING | 10 | 5000 |
| 2 | CLARK | 10 | 2450 |
| 3 | MILLER | 10 | 1300 |
| 4 | SCOTT | 20 | 3000 |
| 5 | FORD | 20 | 3000 |
| 6 | JONES | 20 | 2975 |
| 7 | ADAMS | 20 | 1100 |
| 8 | SMITH | 20 | 800 |
| 9 | BLAKE | 30 | 2850 |
| 10 | ALLEN | 30 | 1600 |
| 11 | TURNER | 30 | 1500 |
| 12 | MARTIN | 30 | 1250 |
| 13 | WARD | 30 | 1250 |
| 14 | JAMES | 30 | 950 |