Contents

[Introduction of DBMS 2](#_Toc88608300)

[DATA 2](#_Toc88608301)

[DATA BASE 2](#_Toc88608302)

[DATA-BASE MANAGEMENT SYSTEM (DBMS) 2](#_Toc88608303)

[Types of DBMS 2](#_Toc88608304)

[Arithmatic Operator – 9](#_Toc88608305)

[Use of “-”Arithmetic Operator 9](#_Toc88608306)

[Use of “\*”Arithmetic Operator:- 10](#_Toc88608307)

[Use of “/”Arithmetic Operator:- 10](#_Toc88608308)

[Literals 11](#_Toc88608309)

[Dual 12](#_Toc88608310)

[Concatenation 12](#_Toc88608311)

[Orderby 15](#_Toc88608312)

[RELATIONAL OPERATOR 17](#_Toc88608313)

[LOGICAL OPERATOR 17](#_Toc88608314)

[SPECIAL OPERATOR 19](#_Toc88608315)

[BETWEEN 20](#_Toc88608316)

[LIKE(Pattern matching) 21](#_Toc88608317)

[NULL 23](#_Toc88608318)

[FUNCTION 25](#_Toc88608319)

[Replace 27](#_Toc88608320)

[Multirow Function 32](#_Toc88608321)

[HAVING 37](#_Toc88608322)

[SUB-QURIES 38](#_Toc88608323)

[JOINS 40](#_Toc88608324)

[Note: 41](#_Toc88608325)

[DDL 42](#_Toc88608326)

[ALTER 44](#_Toc88608327)

***SQL CONTENTS: -***

* **Introduction of DBMS**
* **Types of DBMS**
* **DQL (Data Query Language)**
* Joint, Sub- Queries, Co- Related Sub-Queries
* **DDL (Data Definition Language)**
* Create, Alter, Rename, Drop, Truncate
* Data Type, Constraints
* **DML (Data Manipulation Language)**
* Insert, Update, Delete
* **TCL (Transaction Control Language)**
* Commit, Same point, Roll Back, Flashback, Purge
* **DCL (Data Control Language)**
* Grant, Revoke
* **Functions**
* Single Row Functions
* Multi Row Functions
* Data Functions
* **Normalization (1 NE, 2NE, 3NE)**

# Arrow Slight curveComputerIntroduction of DBMS



## DATA

* It may be defined as the useful information & all the data are stored in a centralized location is known as data.

## ***DATA BASE***

* It is defined as a useful information which is organized by certain set of rules is called database.

## ***DATA-BASE MANAGEMENT SYSTEM (DBMS)***-

* It is defined as a **collection of programs** which is written in order to **manage data base effectively** is known as DBMS.

## TYPES OF DBMS

1. Flat file DBMS
2. Heretical DBMS
3. Network DBMS
4. **RDBMS (Relational data-base management system)**

## RDBMS (Relational data-base management system)-

* It is defined as **collection of data** which is **organized** by following **some relational rule** is called RDBMS.
* In this type **data** will be **store in the form of tables** and its is also easy to **relate multiple tables** to fetch an output.

**Store Data with Rules and Table**

## DATA BASE VENDOR-

* Vendor are those who manufactures the database & sells it is called Data Base Vendor.

|  |  |
| --- | --- |
| **Name of Vendor** | **Product** |
| ORACLE | 10G,11G,12G |
| SAP | Sybase |
| IBM | DB2 |
| Open Source | My SQL |

## STRUCTURE AND NAMING OF DATABASE COMPONENT

* **ENTITY- Each and every table** is called entity.

Example--->

**Entity**

Table name- **Student**

***Student***

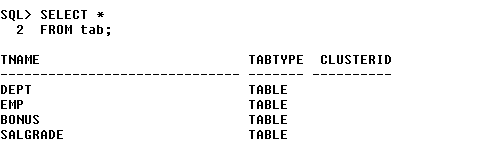
**Column Name/ Table Structure**

**Record /Data**

|  |  |  |
| --- | --- | --- |
| **S\_ID** | **S\_Name** | **Course** |
| 01 | Ram | SQL |
| 02 | Sayam | JAVA |

**Show All table Name in SQL Code -**

* To display tables names in the any SQL database then we have to use below query

**

Syntax- SELECT \*

FROM tab;

Example- ***SELECT \****

***FROM tab***;

**Display Column Name In SQL -Code: -**

* To display column names from any table in a SQL database then we have to use below query

Syntax- DESC Table-name;

Example- DSEC emp;

**O/p**-

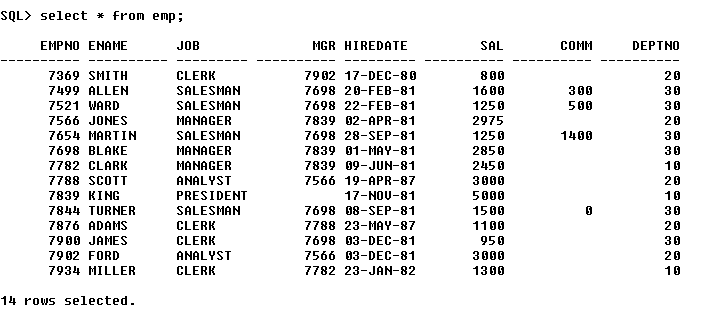
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table** | **Column** | **Data Type** | **Length** | **Precision** | **Scale** | **Primary Key** | **Nullable** | **Default** | **Comment** |
| [EMP](javascript:ret_Column('WKSP_KURESH02.EMP');) | [EMPNO](javascript:ret_Column('EMPNO');) | NUMBER | - | 4 | 0 | 1 | - | - | - |
|  | [ENAME](javascript:ret_Column('ENAME');) | VARCHAR2 | 50 | - | - | - | nullable | - | - |
|  | [JOB](javascript:ret_Column('JOB');) | VARCHAR2 | 50 | - | - | - | nullable | - | - |
|  | [MGR](javascript:ret_Column('MGR');) | NUMBER | - | 4 | 0 | - | nullable | - | - |
|  | [HIREDATE](javascript:ret_Column('HIREDATE');) | DATE | 7 | - | - | - | nullable | - | - |
|  | [SAL](javascript:ret_Column('SAL');) | NUMBER | - | 7 | 2 | - | nullable | - | - |
|  | [COMM](javascript:ret_Column('COMM');) | NUMBER | - | 7 | 2 | - | nullable | - | - |
|  | [DEPTNO](javascript:ret_Column('DEPTNO');) | NUMBER | - | 2 | 0 | - | nullable | - | - |
|  |  |  |  |  |  |  |  |  |  |

**Display any particular table information in SQL Code: -**

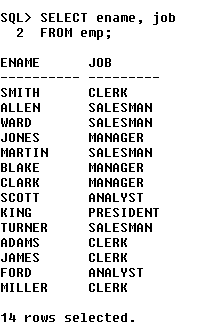
* To display any particular table information in a SQL database then we have to use below query

Syntax- SELECT \* FROM table-name;

Example- SELECT \* FROM emp;

O/p-

**Display multiple column names from a table information in SQL Code: -**

* To display multiple column names from any table in a SQL database then we have to use below query

Syntax- SELECT column1name, column2name FROM tablename;

Example- SELECT ename, job FROM emp;

O/p🡪

## \*Note/Basic Rules for SQL:-

1. We can’t use other column name which in not present in the table from which we are trying to retrieve data.

(Ex- If we searching for dept. name in emp table then we can’t find it 8 because dept. name information is present in the dept. table only)

I/p- *Select dept from emp;*

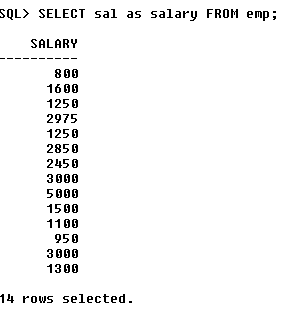
O/p- Error ….Dept column not present in the emp table

1. SQL in not a case-sensitive language.
2. Knowledge Table name and Column name is mandatory.
3. Table names and Column name need to be use as it is present in the table
4. SQL end with ;

# SQL Operator and Function

## Aliasing -

**Display Aliasing column in SQL Code: -**

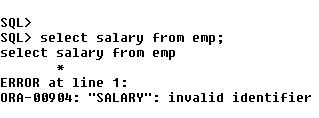
* **Giving another name** for an existing column name for a table is called aliasing.

Syntax :- SELECT columnname as aliasing FROM tablename;

Example:- SELECT sal as salary

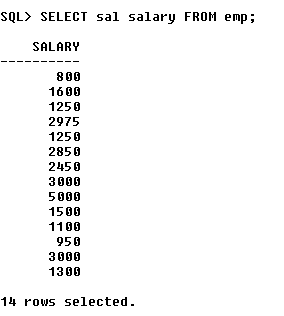
FROM emp;

**Aliasing Name**

Example: - SELECT salary FROM emp;

**O/p-** Error at Line 1

“Salary”: invalid Identification.

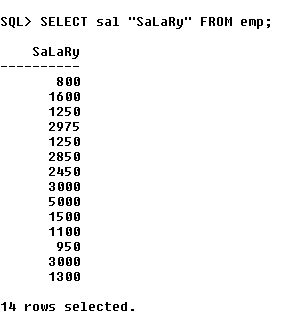
**Note-**

1.  Aliasing is not a permanent change. It is used for just to display purpose for Column name/Table name output. -
2. By default, aliasing will take “as” keyword.

**Example: - *SELECT* *sal salary FROM emp;***

**Note-** To Display the aliasing name as it is given in the query then we have to use “aliasingname”.

**Example: - *SELECT* *sal “SaLaRy” FROM emp;***

******

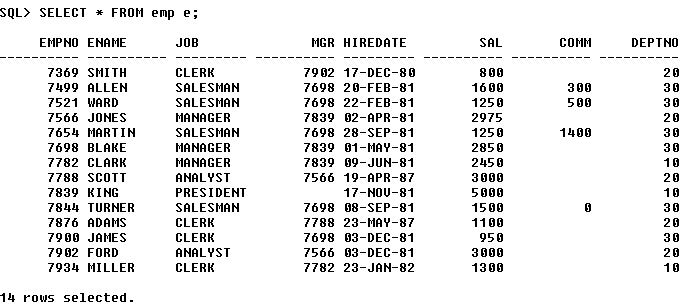
* Giving another name for an existing table name to a old table is called aliasing.

**Display Aliasing a table in SQL Code: -**

Syntax: - SELECT \* FROM table\_name aliasing\_name;

**Example: -** SELECT \* FROM emp e;

O/p-



## Red Foam HandArithmatic Operator –

1. Red Foam Hand“+” Addition
2. Red Foam Hand“-“ Substruction
3. Red Foam Hand“\*” Multiplication
4. Red Foam Hand“/” Division
5. “Modl” Modulus

## Use of “ + ”Arithmetic Operator

On select statement we can able to perform Arithmetic Operation.

Example- ***SELECT Ename, sal+100 FROM emp;***

|  |  |
| --- | --- |
| **Ename** | **Sal** |
| Smith | 800 |
| Allen | 1600 |

|  |  |
| --- | --- |
| **Ename** | **sal+100** |
| Smith | 900 |
| Allen | 1700 |

**Write a query to display employee names and increase the salary of 100 rupees?**

Using Arithmetic Operator “+” Table

Original Table

## Use of “ - ”Arithmetic Operator

**Write a query to display employee names and decrease the salary of 100 rupees?**

O/p- ***Select Ename, sal-100 from emp;***

|  |  |
| --- | --- |
| **Ename** | **Sal** |
| Smith | 800 |
| Allen | 1600 |

|  |  |
| --- | --- |
| **Ename** | **Sal** |
| Smith | 700 |
| Allen | 1500 |

Original Table

Using Arithmetic Operator “**-**” Table

## Use of “\*”Arithmetic Operator:-

**Write a query to display employee names and their 12 month salary in rupees? Write a query to display employee names and their 1/10th salary in rupees?**

O/p- ***Select Ename, sal\*12 from emp;***

Original Table

|  |  |
| --- | --- |
| **Ename** | **sal** |
| Smith | 800 |
| Allen | 1600 |

|  |  |
| --- | --- |
| **Ename** | **Sal\*12** |
| Smith | 9600 |
| Allen | 19200 |

Using Arithmetic Operator “**\***” Table

**Write a query to display employee names and their 1/10th salary in rupees?**

## Use of “/”Arithmetic Operator:-

**Example-**

|  |  |
| --- | --- |
| **Ename** | **sal** |
| Smith | 800 |
| Allen | 1600 |

|  |  |
| --- | --- |
| **Ename** | **Sal/10** |
| Smith | 80 |
| Allen | 160 |

O/p- ***SELECT Ename, sal/10 FROM emp;***

Original Table

Using Arithmatic Operator “ **/**” Table

## Use of “Modulus“Arithmetic Operator:-

**Example- Write a query to display salary and show the reminder after divided by 3?**

O/p- ***Select salary, mod(sal,3) from emp;***

If we divide salary with mod operator it will divide salary with the mod number and show the reminder ex- 800/3=2

|  |  |
| --- | --- |
| **Sal** | **Mod(sal,3)** |
| 800 | 2 |
| 1600 | 1 |

|  |  |
| --- | --- |
| **Ename** | **sal** |
| Smith | 800 |
| Allen | 1600 |

Original Table

Using Arithmatic Operator”**mod**” Table

## LITERALS

* **Usages of data directly** is called literals.
* Data will be classified into 3 things.

***Numbers***- sal, age, marks ….Example **07, 08, 09**.

***String***- ename, jobs, location….Example **‘Satya’, ‘Ram’, ‘Sayam’**.

***Date***- AD, BC, DOB ….Example **‘07-Dec-2021’**.

* **Note -**
* **Number data** are use **directly**.
* Using **String and Date data** should mention in **single cote** ‘ ’.
* Dual **–**
* Dual is a dummy table which works as a worksheet to do analysis.

Example - SELECT 1 FROM dual;

Dummy Column

Name

Attribute

|  |
| --- |
| 1 |
| 1 |

**O/p**-

Example - SELECT 1 FROM dual

Dummy Column

Name

Attribute

|  |
| --- |
| uma |
| uma |

**O/p**-

Example- SELECT ‘ename’ FROM emp;

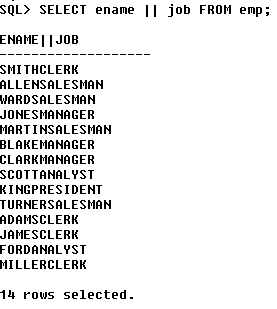
|  |
| --- |
| **Ename** |
| Ename |
| ……….. |
| Ename |
|  |

**O/p-**

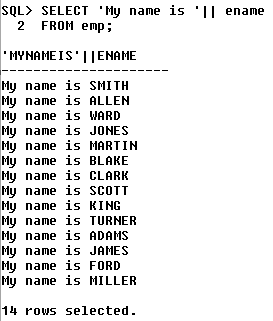
14 times

## **Concatenation** –

* **Joining or merging two or more column values or literals** is called **concatenation**.
* To **achieve concatenation**, we have to **use** (**||**) **double vertical pipes**.

Example- SELECT ename ||job FROM emp;

**Write a query to display my name is ename from emp table?**



***Answer*** – SELECT ‘My name is ’ || ENAME FROM emp;

Answer – ***SELECT ename || ' is earning ' || sal FROM emp;***

Write a query to display ename is earning salary from emp table?

O/p-

Write a query to display **ename is a [job]** from emp table?

|  |
| --- |
| **ENAME||'ISEARNING'||SAL** |
| KING is earning 5000 |
| BLAKE is earning 2850 |
| CLARK is earning 2450 |
| JONES is earning 2975 |
| SCOTT is earning 3000 |
| FORD is earning 3000 |
| SMITH is earning 800 |
| ALLEN is earning 1600 |
| WARD is earning 1250 |
| MARTIN is earning 1250 |
| TURNER is earning 1500 |
| ADAMS is earning 1100 |
| JAMES is earning 950 |
| MILLER is earning 1300 |

Answer – SELECT ename || ‘is a [‘ || job||’]’ FROM emp;

|  |
| --- |
| **ENAME||'ISA['||JOB||']'** |
| KING is a [ PRESIDENT ] |
| BLAKE is a [ MANAGER ] |
| CLARK is a [ MANAGER ] |
| JONES is a [ MANAGER ] |
| SCOTT is a [ ANALYST ] |
| FORD is a [ ANALYST ] |
| SMITH is a [ CLERK ] |
| ALLEN is a [ SALESMAN ] |
| WARD is a [ SALESMAN ] |
| MARTIN is a [ SALESMAN ] |
| TURNER is a [ SALESMAN ] |
| ADAMS is a [ CLERK ] |
| JAMES is a [ CLERK ] |
| MILLER is a [ CLERK ] |

## ORDER BY-

By using ‘orderby’ clause we will be able to rearrange the output in ascending or descending order.

***Example***: SELECT sal FROM emp

ORDER by sal desc;

|  |
| --- |
| **SAL** |
| 5000 |
| 3000 |
| 3000 |
| 2975 |
| 2850 |
| 2450 |
| 1600 |
| 1500 |
| 1300 |
| 1250 |

O/p:

**Note:**

By default order by is in ascending order.

Example: SELECT ename FROM emp ORDER BY ename;

|  |
| --- |
| **ENAME** |
| ADAMS |
| ALLEN |
| BLAKE |
| CLARK |
| FORD |
| JAMES |
| JONES |
| KING |
| MARTIN |
| MILLER |

O/p:

**Select**: it chooses which column needs to displayed in the output

**From**: it chooses the table to fetch the data

**Where**: As per the condition provided it will execute each and every row and decides which row to pick which row to reject.

STD

|  |  |  |
| --- | --- | --- |
| SID | SNAME | COURSE |
| 1 | RAJA | JAVA |
| 2 | RANI | MAVA |
| 3 | CHAMPA | BAVA |
| 4 | CHAMELI | RAVA |

Select \* from std where sname = ‘RAJA’ ;

o/p:

|  |  |  |
| --- | --- | --- |
| SID | SNAME | COURSE |
| 1 | RAJA | JAVA |

**Write a query to display BLAKE record?**

Select \* from emp where ENAME = 'BLAKE';

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **EMPNO** | **ENAME** | **JOB** | **MGR** | **HIREDATE** | **SAL** | **COMM** | **DEPTNO** |
| 7698 | BLAKE | MANAGER | 7839 | 05-01-1981 | 2850 | - | 30 |

O/p:

**Write a query to display employee name, designation and joining date, for all the CLERKS?**

Ans- SELECT ename, job, hiredate

FROM emp

WHERE job = 'CLERK';

|  |  |  |
| --- | --- | --- |
| **ENAME** | **JOB** | **HIREDATE** |
| SMITH | CLERK | 12/17/1980 |
| ADAMS | CLERK | 01-12-1983 |
| JAMES | CLERK | 12-03-1981 |
| MILLER | CLERK | 01/23/1982 |

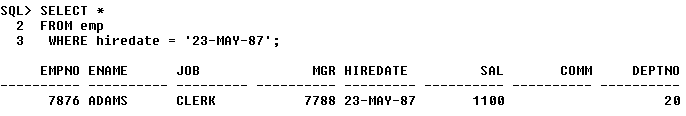
O/P:

**Write a query to display all the employee details who joined on 23-MAY-87?**

Ans- SELECT \*

FROM emp

WHERE hiredate = ‘23-MAY-87’;

O/P:-

# RELATIONAL OPERATOR

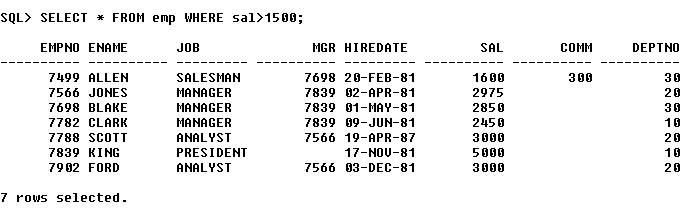
It can be performed only in where statement but not in select statement

* > Greater than
* < smaller than
* >= greater than equal to
* <= smaller than equal to
* = equal to
* != not equal to

Write a query to display all the employee details who is earning salary more than 1500?

Answer- SELECT \* FROM emp WHERE sal >1500;

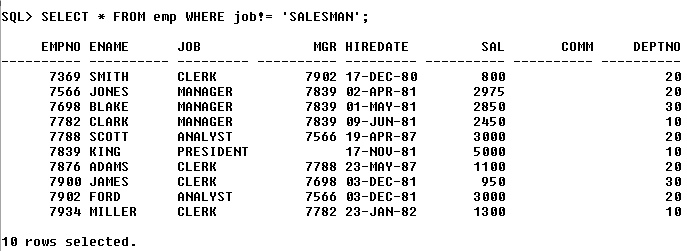
O/P:



**Write a query to display all employee details apart from SALESMAN?**

Ans:- SELECT \* FROM emp WHERE job!= ‘SALESMAN’;

O/P:-



# LOGICAL OPERATOR

“**AND**” when ever we use and operator rows get selected if both the conditions are satisfied.

“**OR**” when ever we use or operator rows get selected if anyone condition is satisfied.

“**NOT**” it is an opposite operator

**Write a query to display all the details of CLERK who belongs to 20th dept?**

Ans- SELECT \* FROM emp WHERE job=’CLERK’ AND deptno = 20;

O/P:-

