***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)**

# Introduction 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 |

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

Example--->

**Entity**

Table name- **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;**

O/p- TNAME TABTYPE CLUSTERID

------------------------------ ------- ----------

DEPT TABLE

EMP TABLE

BONUS TABLE

SALGRADE TABLE

**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 tablename;**

Example- ***desc 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 tablename;**

Example- ***Select \* from emp;***

O/p-

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
| 7839 | KING | PRESIDENT |  | 11/17/1981 | 5000 |  | 10 |
| 7698 | BLAKE | MANAGER | 7839 | 05-01-1981 | 2850 |  | 30 |
| 7782 | CLARK | MANAGER | 7839 | 06-09-1981 | 2450 |  | 10 |
| 7566 | JONES | MANAGER | 7839 | 04-02-1981 | 2975 |  | 20 |
| 7788 | SCOTT | ANALYST | 7566 | 12-09-1982 | 3000 |  | 20 |
| 7902 | FORD | ANALYST | 7566 | 12-03-1981 | 3000 |  | 20 |
| 7369 | SMITH | CLERK | 7902 | 12/17/1980 | 800 |  | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 02/20/1981 | 1600 | 300 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 02/22/1981 | 1250 | 500 | 30 |
| 7654 | MARTIN | SALESMAN | 7698 | 09/28/1981 | 1250 | 1400 | 30 |
| 7844 | TURNER | SALESMAN | 7698 | 09-08-1981 | 1500 | 0 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 01-12-1983 | 1100 |  | 20 |
| 7900 | JAMES | CLERK | 7698 | 12-03-1981 | 950 |  | 30 |
| 7934 | MILLER | CLERK | 7782 | 01/23/1982 | 1300 |  | 10 |

**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-

|  |  |
| --- | --- |
| ENAME | JOB |
| KING | PRESIDENT |
| BLAKE | MANAGER |
| CLARK | MANAGER |
| JONES | MANAGER |
| SCOTT | ANALYST |
| FORD | ANALYST |
| SMITH | CLERK |
| ALLEN | SALESMAN |
| WARD | SALESMAN |
| MARTIN | SALESMAN |
| TURNER | SALESMAN |
| ADAMS | CLERK |
| JAMES | CLERK |
| MILLER | CLERK |

**\*Note:-**

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 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 ;

* **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;***

O/p-

|  |
| --- |
| **SALARY** |
| 5000 |
| 2850 |
| 2450 |
| 2975 |
| 3000 |
| 3000 |
| 800 |
| 1600 |
| 1250 |
| 1250 |
| 1500 |
| 1100 |
| 950 |
| 1300 |

**Example: - *select* *salary from emp;***

O/p- Error at Line 1

“Salary”: invalid Identification.

**Note-**

|  |
| --- |
| **SALARY** |
| 800 |
| 900 |
| 1200 |

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

**Example: - *select* *sal salary from emp;***

|  |
| --- |
| ***SaLaRy*** |
| 800 |
| 900 |
| 1200 |

 **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;***

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

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

Syntax :- **select \* from table\_name aliasing\_name;**

**Example: - *select* *from emp e;***

O/p-

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Emp No** | **Ename** | **Job** | **Mcr** | **HireDate** | **Sal** | **Comm** |
| 7867 | Tuna | Cleark | 7402 | 20-8-2020 | 800 | 300 |
| 8765 | Muna | Manager | 7562 | 23-8-2020 | 1600 | 500 |

# 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

On select statement we can able to perform Arithmetic Operation.

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

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

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

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

Original Table

Using Arithmatic Operator “+” Table

## Use of “-”Arithmetic Operator

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

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

|  |  |
| --- | --- |
| **Ename** | **Sal-100** |
| Smith | 700 |
| Allen | 1500 |

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

Using Arithmatic Operator “**-**” Table

Original Table

# Use of “\*”Arithmetic Operator:-

**Example- Write a query to display employee names and their 12 month salary in rupees?**

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

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

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

Using Arithmatic Operator “**\***” Table

Original Table

Using Arithmatic Operator”**-**” Table

# Use of “/”Arithmetic Operator:-

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

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

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

O/p- ***Select Ename, sal/10 from emp;***

Using Arithmatic Operator”**\***” Table

Original 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 |

Using Arithmatic Operator “ **mod**” Table

Original 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 uma from dual;

Dummy Column Name

Attribute

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

**O/p**-

**Example-** select‘ename’ from emp;

|  |
| --- |
| **Ename** |
| 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;

O/p-

|  |
| --- |
| **ENAME||JOB** |
| KINGPRESIDENT |
| BLAKEMANAGER |
| CLARKMANAGER |
| JONESMANAGER |
| SCOTTANALYST |
| FORDANALYST |
| SMITHCLERK |
| ALLENSALESMAN |
| WARDSALESMAN |
| MARTINSALESMAN |
| TURNERSALESMAN |
| ADAMSCLERK |
| JAMESCLERK |
| MILLERCLERK |

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

**Answer** – *select ‘My name is’ || ename from emp;*

O/p-

|  |
| --- |
| **'MYNAMEIS'||ENAME** |
| My name is KING |
| My name is BLAKE |
| My name is CLARK |
| My name is JONES |
| My name is SCOTT |
| My name is FORD |
| My name is SMITH |
| My name is ALLEN |
| My name is WARD |
| My name is MARTIN |
| My name is TURNER |
| My name is ADAMS |
| My name is JAMES |
| My name is MILLER |

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

**Answer** – *select ename || ' is earning ' || sal from emp;*

O/p-

|  |
| --- |
| **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 |

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

**Answer** – ***select ename || ‘is a [‘ || job||’]’ from emp****;*

O/p-

|  |
| --- |
| **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 ] |

## Orderby

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

Ex:

Select sal from emp

Order by sal desc;

o/p:

Note:

By default order by is in ascending order.

Ex:

Select ename from emp order by ename;

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’;

O/p:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

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

Select ename, job, hiredate from emp where job = ‘CLERK’;

O/P:

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

Select \* from emp where hiredate = ‘23-MAY-87’ ;

O/P:

# RELATIONAL OPERATOR

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

* > Greater than
* < smaller than
* >= grater 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?

Select \* from emp where sal>1500;

O/P:

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

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?

Select \* from emp where job=’CLERK’ and deptno = 20;

O/P:

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

Select \* from emp where job=’SALESMAN’ and sal>1500;

O/P:

Write a query to display all employee details who belongs to 10& 30 dept?

Select \* from emp where deptno = 10 or deptno = 30;

O/P:

Write a query to display all employee details of salesman & manager?

Select \* from emp where JOB=’SALESMAN’ or JOB=’MANAGER’;

O/P:

# SPECIAL OPERATOR

IN:

* The record should present in same list or same column
* When we use multiple ‘or’ operator then we are going to replace a special operator called ‘in’.

EX:

Select \* from emp where mgr in(7698,7839,7566);

O/P:

Write a query to display all employee details of CLERK & MANAGER who belongs to 30th dept?

Select \* from emp where job in(‘CLERK’,’MANAGER’) and deptno= 30;

O/p:

Write a query to display all employee details who joined in the year 81?

Select \* from emp where hiredate>=’01-JAN-81’ and hiredate<=’31-DEC-81’;

O/P:

# BETWEEN

When ever we want to display the values in range then we are going to use a special operator called between.

Ex:

Select \* from emp where hiredate between ’01-JAN-81’ and ’31-DEC-81’;

O/P:

Write a query to display all employee details who is earning salary 2000 to 3000?

Select \* from emp where sal between ‘2000’ and ‘3000’;

O/P:

# LIKE(Pattern matching)

“%” It derives ‘0’ to ‘n’ characters

“\_” it derives a single character

|  |  |  |
| --- | --- | --- |
| SID | SNAME | COURSE |
| 1 | RAJA | BIOLOGY |
| 2 | RANI | ZOOLOGY |
| 3 | CHAMPA | BOTANY |
| 4 | CHAMELI | SOCIAL |

STD

Select \* from std where sname like ‘R%’;

O/P: RAJA

RANI

Select \* from std where course like ‘%y’;

O/P: BIOLOGY

ZOOLOGY

BOTANY

Select \* from std where course like ‘B%Y’;

O/P: BIOLOGY

BOTANY

Select \* from std where sname like ‘%A%’;

O/P: RAJA

RANI

CHAMPA

CHAMELI

Select \* from std where sname like ‘\_\_\_\_’;

O/P: RAJA

RANI

Select \* from std where sname like ‘\_A%’;

O/P: RAJA

RANI

Select \* from std where sname like ‘R\_\_I’;

O/P: RANI

Write a query to display all names who’s names starts with S?

Select \* from emp where ename like ‘S%’;

O/P:

Write a query to display all employee details whose designation ends with ‘MAN’?

Select \* from emp where job like ‘%MAN’;

O/P:

Write a query to display all employee details who joined in year of 81?(by using like operator)

Select \* from emp where hiredate like ‘%81’;

O/P:

Write a query to display all employee details who joined in the month of December?

Select \* from emp where hiredate like ‘%DEC’;

O/P:

Write a query to display all employee details whose salary contains 2nd letter 5 in it?

Select \* from emp where SAL like ‘%5\_’;

O/P:

# NULL

* It is an empty space or blank space. Null means nothing.
* It didn’t occupy space in the memory.
* Null!=0
* To achieve null we have to use special operator called “is”.

Write a query to display all employee details who does not have commission?

Select \* from emp where comm is null;

Write a query to display all employee details who don’t have manager number?

Select \* from emp where mgr is null;

Special Operator

Logical Operator

IN

BETWEEN

LIKE

IS

AND

OR

NOT

Write a query to display all employee details apart from the names start with S?

Select \* from emp where ename not like ‘S%’;

O/P:

Write a query to display all the employee details apart from 10 & 30 dept?

Select \* from emp where deptno not in(10,30);

O/P:

Write a query to display all employee details who is earning salary less than 1000 and more than 2000?

Select \* from emp where sal not between 1000 and 2000;\

O/P:

Write a query to display all employee details who have some commission?

Select \* from emp where comm is not null;

O/P:

# FUNCTION

* SINGLE ROW FUNCTION
* MULTI ROW FUNCTION
* DATE FUNCTION

Single row function: It will take multiple inputs and gives you correspondent output for each input.

Upper()

Lower()

Initcap()

Concat()

Replace()

Length()

Substr()

Write a query to display all the employee name in lower case and designation initial capital letter?

Select lower(ename), initcap(job) from emp;

O/P:

Concat()

Syntax:- select concat(arg1, arg2) from tablename;

Arg1,arg2: column names/ literals.

Concat function is similar to concatenation. Where we are going to join or merge two or more column values or literals. Concat function will take maximum of two arguments.

Select concat (ename, job) from emp;

O/P:

Write a query to display in below format?

My name is SMITH

My name is ALLEN

Select concat (‘my name is’ , ename) from emp;

O/P:

Write a query to display in below format?

Smith is earning 800

Select concat(caoncat(ename, ‘ is earning ‘ ), sal) from emp;

O/P:

Write a query to display below format?

Smith is earning 800 and joined on 7th-DEC-80

Select concat (concat(concat(ename, ‘ is earning ’), sal, ‘ and joined on ’),hiredate) from emp;

O/P:

## Replace

Replace function will replace or will change old text data to new text data.

Syntax:

Select replace(arg1, arg2, arg3) from tablename;