SQL SERVER

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

1 SQL

2 T-SQL (Transact-SQL)

SQL databases :-

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

sql server from microsoft

oracle from oracle corp

db2 from ibm

mysql from oracle corp

postgresql from postgresql forum

rds from amazon

SQL SERVER ORACLE MYSQL

SQL SQL SQL

NoSQL databases :-

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

1 mongoDB

2 cassandra

06-may-23

Database :-

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

=> a Database is a organized collection of interrelated data. For example

a univ db stores data related to students,courses,faculty etc and

a bank db stores data related to customers,trans,accounts etc.

Types of Databases :-

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

1 OLTP DB (online transaction processing) => DB

2 OLAP DB (online analytical processing) => DWH

=> organizations uses OLTP DB for storing day-to-day transactions and

OLAP for data analysis.

=> OLTP is for running business and OLAP is for to analyze business.

=> day-to-day operations on db includes

C create

R read

U update

D delete

DBMS :-

-----------

=> DBMS stands for Database Management System and it is a software

used to create and to manage database.

=> DBMS is an interface between user and database.

Evolution of DBMS :-

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

1960 fms (file mgmt system)

1970 hdbms (hierarchical dbms)

ndbms (network dbms)

1980 rdbms (relational dbms)

1990

RDBMS :-

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

=> RDBMS concepts are introduced by E.F.CODD

=> E.F.CODD introduced 12 rules called CODD rules

=> a db software that supports all 12 rules is called perfect rdbms

information rule :-

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

=> according information rule in rdbms data must be organized in tables

i.e. rows and columns.

CUST

CID NAME ADDR => columns/fields/attributes

10 SACHIN MUM

11 VIJAY HYD

12 RAHUL DEL => row/record/tuple

DATABASE = COLLECTION OF TABLES

TABLE = COLLECTION OF ROWS & COLS

ROW = COLLECTION OF FIELD VALUES

COLUMN = COLLECTION OF VALUES BELONGS TO ONE FIELD

=> every table must contain primary key to uniquely identify the records

ex :- accno,empid,aadharno,panno,voterid

RDBMS features :-

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

1 easy to access and manipulate data

2 less redundency (duplication of data)

3 more security

4 gurantees data quality

5 supports data sharing

6 supports transactions

RDBMS softwares :- (SQL databases)

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

SQL SERVER from microsoft

ORACLE from oracle corp

DB2 from ibm

MYSQL from oracle corp

POSTGRESQL from postgresql global development forum

RDS from amazon

NoSQL Databases :-

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

1 mongoDB

2 cassandra

08-may-23

ORDBMS :-

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

=> object relational database management system

=> it is the combination of rdbms & oops

ordbms = rdbms + oops (reusability)

=> rdbms doesn't support reusability but ordbms supports reusability

ordbms softwares :-

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

1 sql server

2 oracle

3 postgresql

DB Development Life Cycle :-

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

Analysis

Design

Development

Testing

Implementation

Maintenance

Design :-

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

=> Designing db means designing tables.

=> tables are desinged by designer or architect.

=> DB designer designs db by using following tools

1 ER MODEL

2 NORMALIZATION

Development :-

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

=> DB is developed by DB Developers & DBAs

=> DB is developed by using any rdbms tool like sql server

DEVELOPER DBA (DB Admin)

creating tables installation of sql server

creating views creating database

creating synonyms creating logins

creating sequences db backup & restore

creating indexes db export & import

creating procedures db upgradation & migration

creating functions performance tuning

creating triggers

writing queries

sql server 2005 sql server 2022 db upgradation

mysql sql server 2022 db migration

09-may-23

Testing :-

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

=> db is tested by QA team (Quality Assurance) by using some tools called testing tools

Implementation :-

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

=> implementation means moving db from dev server to prod server.

=> once db is moved to prod server then end user can use db for day-to-day operations.

summary :-

what is db ?

what is dbms ?

what is rdbms ?

what is ordbms ?

what is db development life cycle ?

=====================================================================

SQL SERVER

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

=> SQL SERVER is basically a rdbms product and also supports ordbms features

and used to create and to manage database.

=> SQL SERVER can be used for DB Development & DB Administration

versions of sql server :-

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

versions of sql server :-

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

version year

SQL SERVER 1.1 1991

SQL SERVER 4.2 1993

SQL SERVER 6.0 1995

SQL SERVER 6.5 1996

SQL SERVER 7.0 1998

SQL SERVER 2000 2000

SQL SERVER 2005 2005

SQL SERVER 2008 2008

SQL SERVER 2012 2012

SQL SERVER 2014 2014

SQL SERVER 2016 2016

SQL SERVER 2017 2017

SQL SERVER 2019 2019

SQL SERVER 2022 2022

sql server 2016 :-

1 polybase

2 json (javascript object notation)

3 temporal table to save data changes.

4 dynamic data masking and row level security

sql server 2017 :-

1 identity cache

2 New String functions

3 Automatic Tuning

sql server 2019 :-

1 Read, write, and process big data from Transact-SQL

2 Easily combine and analyze high-value relational data with high-volume big data.

3 Query external data sources.

4 Store big data in HDFS managed by SQL Server.

5 Query data from multiple external data

CLIENT/SERVER Architecture :-

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

1 SERVER

2 CLIENT

SERVER :-

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

=> server is a system where sql server software is installed and running.

=> inside the server sql server manages two memories

1 DB

2 INSTANCE

=> DB is created in hard disk and acts as permanent storage

=> INSTANCE is created in ram and acts as temporary storage

CLIENT :-

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

=> client is also a system where users can

1 connect to server

2 submit the requests

3 receive response

client tool :-

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

SSMS => SQL SERVER MANAGEMENT STUDIO

USER------SSMS-------------------------------------SQL SERVER-----DB

USER------SQLPLUS-----------------------------------ORACLE----------DB

USER----MYSQLWORKBENCH----------------------MYSQL------------DB

USER----PGADMIN-----------------------------------------POSTGRESQL--------DB

SQL :-

-----------

=> SQL stands for structured query language.

=> It is a language used to communicate with sql server.

=> user communicates with sql server by sending commands called queries.

=> a query is a command/instruction/question submitted to sql server to perform some

operation over db.

=> SQL is orignially introduced by IBM and initial name of this language was "SEQUEL"

and later it is renamed to SQL.

=> SQL is common to all RDBMS

sql server oracle mysql postgresql

sql sql sql sql

=> based on operations over db SQL is categorized into following sublanguages

DDL (DATA DEFINITION LANG)

DML (DATA MANIPULATION LANG)

DQL (DATA QUERY LANG)

TCL (TRANSACTION CONTROL LANG)

DCL (DATA CONTROL LANG)

SQL

DDL DML DQL TCL DCL

create insert select commit grant

alter update rollback revoke

drop delete save transaction

truncate merge

DATA & DATA DEFINITION :-

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

EMPID ENAME SAL => DATA DEFINITION / METADATA

1 A 5000 => DATA

USER---SSMS-------------SQL COMMANDS-------------SQL SERVER------DB

USER---SQLPLUS--------SQL COMMANDS-------------ORACLE-----------DB

How to connect to sql server :-

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

=> to connect to sql server open ssms and enter following details

SERVER TYPE :- DB ENGINE

SERVER NAME :- DESKTOP-G2DM7GI

AUTHENTICATION :- WINDOWS/SQL SERVER

LOGIN :- SA (SYSTEM ADMIN)

PASSWORD :- 123

=> click connect

10-may-23

CREATING DATABASE :-

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

=> in object explorer select Databases => New Database

Enter Database Name :- DB11AM

=> click OK

=> a database is created with following two files

1 DATA FILE

2 LOG FILE

=> Data File stores data and Log file stores operations

Name Type Initial Size Autogrowth path

DB11AM DATA 8 MB 64 MB

DB111AM\_LOG LOG 8 MB 64 MB

path :-

----------

C:\program files\microsoft sql serve\mssql15.mssqlserver\mssql\data\

DB11AM.MDF

DB11AM\_LOG.LDF

MDF => master data file

LDF => log data file

Downloading & installation :-

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

downloading sql server :-

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

https://www.microsoft.com/en-in/sql-server/sql-server-downloads

step by step installation of sql server 2022 :-

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

https://www.sqlshack.com/how-to-install-sql-server-developer-edition/

download ssms :-

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

https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16

Datatypes in sql server :-

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

=> a Datatype specifies

1 what type of data allowed in column

2 how much memory allocated for column

DATATYPES

CHAR INTEGER FLOAT CURRENCY DATE BINARY

ASCII UNICODE tinyint numeric(p,s) money date binary

char nchar smallint decimal(p,s) smallmoney time varbinary

varchar nvarchar int datetime varbinary(max)

varchar(max) nvarchar(max) bigint

numeric(p)

char(size) :-

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

=> allows character data upto 8000 chars

=> recommended for fixed length char columns

ex :- NAME CHAR(10)

sachin----

wasted

ravi------

wasted

=> in char datatype extra bytes are wasted , so char is not recommended

for variable length fields and char is recommended for fixed length fields.

ex :- GENDER CHAR(1)

M

F

STATE\_CODE CHAR(2)

AP

TS

COUNTRY\_CODE CHAR(3)

IND

USA

VARCHAR(size) :-

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

=> allows character data upto 8000

=> recommended for variable length fields

ex :- NAME VARCHAR(10)

SACHIN----

released

=> char/varchar allows ascii chars (256 chars) that includes a-z,A-Z,0-9,

special chars.

ex :- PANNO CHAR(10)

VEHNO CHAR(10)

EMAILID VARCHAR(20)

VARCHAR(MAX) :-

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

=> allows character data upto 2GB.

NCHAR/NVARCHAR/NVARCHAR(MAX) :-

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

=> allows unicode chars (65536 chars) that includes characters of

different languages.

11-may-23

Integer Types :-

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

=> allows numbers without decimal part

TINYINT 1 BYTE 0 TO 255

SMALLINT 2 BYTES -32768 TO 32767

INT 4 BYTES -2147483648 to 2147483647

BIGINT 8 BYTES -9223372036854775808

to

9223372036854775807

AGE TINYINT

EMPID SMALLINT

NUMERIC(P) :-

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

=> allows numbers upto 38 digits

ex :- EMPID NUMERIC(4)

10

100

1000

10000 => INVALID

AADHARNO NUMERIC(12)

ACCNO NUMERIC(13)

MOBILE NUMERIC(10)

NUMERIC(P,S) / DECIMAL (P,S) :-

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

=> allows numbers with decimal (float)

P => precision => total no of digits

S => scale => no of digits after decimal

ex :- SALARY NUMERIC(7,2)

5000

5000.55

50000.55

500000.55 => INVALID

BALANCE NUMERIC(11,4)

CURRENCY TYPES :-

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

=> used for fields related to money

=> sql server supports 2 currency types

SMALLMONEY 4 BYTES -2,14,748.3648 to 2,14,748.3647

MONEY 8 BYTES -922337203685477.5808

to

922337203685477.5807

EX :- SAL SMALLMONEY

BAL MONEY

DATE & TIME :-

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

1 DATE => allows only date

2 TIME => allows only time

3 DATETIME => allows date & time

=> default date format in sql server is YYYY-MM-DD

=> default time format is HH:MI:SS

ex :- DOB DATE

2003-03-15

LOGIN TIME

10:00:00

TXNDT DATETIME

2023-05-11 9:30:00

Binary Types :-

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

=> binary types allows multimedia objects like audio,video,images

BINARY => allows binary data upto 8000 bytes

VARBINARY => allows binary data upto 8000 bytes

VARBINARY(MAX) => allows binary data upto 2GB

ex :- photo BINARY(1000) => extra bytes are wasted

photo VARBINARY(1000) => extra bytes are released

============================================================

CREATING TABLES IN SQL SERVER DB :-

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

CREATE TABLE <TABNAME>

(

COLNAME DATATYPE(SIZE),

COLNAME DATATYPE(SIZE),

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

)

Rules :-

----------

1 tabname should start with alphabet

2 tabname should not contain spaces , special chars but allows \_,$,#

3 tabname can be upto 128 chars

4 table can have 1024 cols

5 no of rows unlimited

ex :- 123EMP INVALID

EMP 123 INVALID

EMP\*123 INVALID

EMP\_123 VALID

=> create table with following structure ?

EMP

EMPID ENAME JOB SAL HIREDATE DNO

CREATE TABLE EMP

(

EMPID SMALLINT ,

ENAME VARCHAR(10),

JOB VARCHAR(10),

SAL SMALLMONEY,

HIREDATE DATE,

DNO TINYINT

)

INSERTING DATA INTO TABLE :-

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

=> "INSERT" command is used to insert data into table.

=> we can insert

1 single row

2 multiple rows

inserting single row :-

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

syn :- INSERT INTO <TABNAME> VALUES(V1,V2,V3,------)

ex :-

1 INSERT INTO EMP VALUES(100,'sachin','clerk',5000,'2023-05-11',10)

2 INSERT INTO EMP VALUES(101,'vijay','analyst',8000,getdate(),20)

inserting multiple rows :-

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

INSERT INTO EMP

VALUES(102,'phani','manager',9000,'2019-10-05',30) ,

(103,'sindhu','analyst',7000,'2021-04-20',20)

inserting nulls :-

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

=> a nulls means blank or empty

=> it is not equal to 0 or space

=> nulls can be inserted in two ways

method 1 :-

INSERT INTO EMP VALUES(104,'kumar',null,3000,'2020-02-15',null)

method 2 :-

INSERT INTO EMP(empid,ename,sal,hiredate)

VALUES(105,'james',5000,getdate())

remaining fields job,dno filled with nulls

INSERT INTO EMP VALUES(104,'kumar',null,3000,'2020-02-15',null)

12-may-23

Operators in sql server :-

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

Arithmetic Operators => + - \* / %

10+5 15

10-5 5

10\*5 50

10/5 2

10%5 0

Relational Operators => > >= < <= = <> or !=

10 > 5 true

10 < 5 false

10=5 false

10<>5 true

Logical Operators => AND OR NOT

Special Operators => BETWEEN

IN

LIKE

IS

ANY

ALL

EXISTS

Set Operators => UNION

UNION ALL

INTERSECT

EXCEPT

Displaying Data :-

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

=> "SELECT" command is used to display data from table.

syn :- SELECT columns / \* FROM tabname ;

SQL = ENGLISH

QUERIES = SENTENCES

CLAUSES = WORDS

FROM clause => specify tablename

SELECT clause => specify column names

\* => all columns

=> display employee names and salaries ?

SELECT ENAME,SAL FROM EMP

=> display employee names,jobs and hiredate ?

SELECT ENAMEJOB,HIREDATE FROM EMP

=> display all the data from emp ?

SELECT \* FROM EMP

WHERE clause :-

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

=> used to access specific row/rows from table based on a condition.

SELECT columns

FROM tabname

WHERE condition

condition :-

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

COLNAME OP VALUE

=> OP must be any relational opertor like > >= < <= = <>

=> if cond = true row is selected

=> if cond = false row is not selected

=> display employee details whose id = 102 ?

SELECT \* FROM EMP WHERE EMPID=102

=> display employee details whose name = kumar ?

SELECT \* FROM EMP WHERE ENAME = 'kumar'

=> employees earning more than 5000 ?

SELECT \* FROM EMP WHERE SAL > 5000

=> employees joined after 2020 ?

SELECT \* FROM EMP WHERE HIREDATE > 2020 => ERROR

SELECT \* FROM EMP WHERE HIREDATE > '2020-12-31'

=> employees joined before 2020 ?

SELECT \* FROM EMP WHERE HIREDATE < '2020-01-01'

=> employees not working for dept 10 ?

SELECT \* FROM EMP WHERE DNO <> 10

compound condition :-

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

=> multiple conditions combined with AND / OR operators is called

compound condition.

WHERE COND1 AND COND2 RESULT

T T T

T F F

F T F

F F F

WHERE COND1 OR COND2 RESULT

T T T

T F T

F T T

F F F

=> display employee details whose id = 100,103,105 ?

SELECT \*

FROM EMP

WHERE EMPID = 100 OR EMPID = 103 OR EMPID = 105

=> employees working as clerk,analyst ?

SELECT \*

FROM EMP

WHERE JOB='CLERK' OR JOB='ANALYST'

=> display employees working as analyst belongs to 20th dept ?

SELECT \*

FROM EMP

WHERE JOB='ANALYST' AND DNO=20

=> employees earning more than 5000 and less than 10000 ?

SELECT \*

FROM EMP

WHERE SAL > 5000 AND SAL < 10000

=> employees joined in 2020 year ?

SELECT \*

FROM EMP

WHERE HIREDATE >= '2020-01-01'

AND

HIREDATE <= '2020-12-31'

=> employees working as clerk,analyst and earning more than 5000 ?

SELECT \*

FROM EMP

WHERE JOB='CLERK'

OR

JOB='ANALYST'

AND

SAL>5000

SELECT \*

FROM EMP

WHERE (

JOB='CLERK'

OR

JOB='ANALYST'

)

AND

SAL>5000

13-may-23

IN operator :-

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

=> use IN operator for comparision with list.

=> use IN operator for "=" comparision with multiple values.

WHERE COLNAME = V1,V2,V,3,--- => INVALID

WHERE COLNAME IN (V1,V2,V3,----) => VALID

=> employees whose id = 100,103,105 ?

SELECT \* FROM EMP WHERE EMPID IN (100,103,105);

=> employees working as clerk,manager ?

SELECT \* FROM EMP WHERE JOB IN ('CLERK','MANAGER')

=> employees not working for dept 10,20 ?

SELECT \* FROM EMP WHERE DNO NOT IN (10,20)

BETWEEN operator :-

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

=> use BETWEEN operator for comparision with range.

WHERE COLNAME BETWEEN V1 AND V2 (COL>=V1 AND COL<=V2)

=> employees earning between 5000 and 10000 ?

SELECT \* FROM EMP WHERE SAL BETWEEN 5000 AND 10000

=> employees joined in 2020 year ?

SELECT \*

FROM EMP

WHERE HIREDATE BETWEEN '2020-01-01' AND '2020-12-31'

=> not joined in 2020 ?

SELECT \*

FROM EMP

WHERE HIREDATE NOT BETWEEN '2020-01-01' AND '2020-12-31'

=> employees working as clerk,manager and earning between 5000 and

10000 and joined in 2019 and not working for dept 10,20 ?

SELECT \*

FROM EMP

WHERE JOB IN ('CLERK','MANAGER')

AND

SAL BETWEEN 5000 AND 10000

AND

HIREDATE BETWEEN '2019-01-01' AND '2019-12-31'

AND

DNO NOT IN (10,20)

=>

products

prodid pname price category brand

list of samsung,redmi,realme mobiles phones price between 10000 and 20000 ?

SELECT \*

FROM PRODUCTS

WHERE CATEGORY='MOBILES'

AND

BRAND IN ('SAMSUNG','REDMI','REALME')

AND

PRICE BETWEEN 10000 AND 20000

=>

CUSTOMERS

CID NAME CITY AGE GENDER

list of male customers staying in hyd,mum,del age between 20 and 30 ?

SELECT \*

FROM CUSTOMERS

WHERE GENDER='MALE'

AND

CITY IN ('HYD','DEL','MUM')

AND

AGE BETWEEN 20 AND 30

LIKE operator :-

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

=> use LIKE operator for comparision with patterns

WHERE COLNAME LIKE 'PATTERN'

=> pattern contains alphabets,digits,wildcard chars

wildcard chars :-

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

% => 0 or many chars

\_ => exactly 1 char

=> employees name starts with 's' ?

SELECT \* FROM EMP WHERE ENAME LIKE 'S%'

=> employees name ends with 's' ?

SELECT \* FROM EMP WHERE ENAME LIKE '%S'

=> employees name contains 'a' ?

SELECT \* FROM EMP WHERE ENAME LIKE '%A%'

=> 'a' is the 3rd char in their name ?

SELECT \* FROM EMP WHERE ENAME LIKE '\_\_A%'

=> 'a' is the 3rd char from last ?

SELECT \* FROM EMP WHERE ENAME LIKE '%A\_\_'

=> name contains 4 chars ?

SELECT \* FROM EMP WHERE ENAME LIKE '\_\_\_\_'

=> employees joined in feb month ?

SELECT \*

FROM EMP

WHERE HIREDATE LIKE '\_\_\_\_\_02\_\_\_'

YYYY-MM-DD

=> employees joined 2020 year ?

SELECT \*

FROM EMP

WHERE HIREDATE LIKE '2020%'

15-may-23

IS operator :-

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

=> use IS operator for comparision with NULL / NOT NULL

WHERE COLNAME IS NULL

WHERE COLNAME IS NOT NULL

=> employees not assigned to any dept ?

SELECT \* FROM EMP WHERE DNO IS NULL

=> employees who are assigned to dept ?

SELECT \* FROM EMP WHERE DNO IS NOT NULL

summary :-

WHERE COLNAME IN (V1,V2,V3,---)

WHERE COLNAME BETWEEN V1 AND V2

WHERE COLNAME LIKE 'PATTERN'

WHERE COLNAME IS NULL

Question :-

STUDENT

SID SNAME S1 S2 S3

1 A 80 90 70

2 B 30 60 50

=> list of students who are passed ?

SELECT \* FROM STUDENT WHERE S1>=35 AND S2>=35 AND S3>=35

=> list of students are who are failed ?

SELECT \* FROM STUDENT WHERE S1<35 OR S2<35 OR S3<35

ALIAS :-

----------

=> alias means another name

=> used to change column headings

syn :- COLNAME / EXPRESSION <AS> <ALIAS>

ex :-

=> display ENAME ANNUAL SALARY ?

SELECT ENAME,SAL\*12 AS [ANNUAL SALARY]

FROM EMP

=> display ENAME SAL HRA DA TAX TOTSAL ?

HRA = house rent allowance = 20% on sal

DA = dearness allowance = 30% on sal

TAX = 10% on sal

TOTSAL = SAL + HRA + DA - TAX

SELECT ENAME,SAL,

SAL\*0.2 AS HRA,

SAL\*0.3 AS DA,

SAL\*0.1 AS TAX,

SAL + (SAL\*0.2) + (SAL\*0.3)-(SAL\*0.1) AS TOTSAL

FROM EMP

ORDER BY clause :-

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

=> ORDER BY clause used to sort rows based on one or more columns either in

acending or in descending order.

SELECT columns

FROM tabname

[WHERE cond]

ORDER BY colname ASC/DESC , -------

=> default order is ASC for descending order use DESC option

examples :-

=> arrange employee list name wise asc order ?

SELECT \*

FROM EMP

ORDER BY ENAME ASC

=> arrange employee list sal wise desc order ?

SELECT \*

FROM EMP

ORDER BY SAL DESC

=> arrange employee list dept wise asc and with in dept sal wise desc ?

SELECT EMPNO,ENAME,SAL,DEPTNO

FROM EMP

ORDER BY DEPTNO ASC , SAL DESC

1 A 5000 20 5 E 4000 10

2 B 3000 10 2 B 3000 10

3 C 1000 30 ======> 4 D 6000 20

4 D 6000 20 1 A 5000 20

5 E 4000 10 6 F 3000 30

6 F 3000 30 3 C 1000 30

16-may-23

=>

STUDENTS

SNO SNAME M P C

1 A 80 90 70

2 B 60 70 50

3 C 90 80 70

4 D 90 70 80

=> arrange student list avg wise desc , m desc, p desc ?

SELECT \*

FROM STUDENTS

ORDER BY (M+P+C)/3 DESC,M DESC,P DESC

3 C 90 80 70

4 D 90 70 80

1 A 80 90 70

2 B 60 70 50

=> but to display avg in output

SELECT \* , (M+P+C)/3 AS AVG

FROM STUDENTS

ORDER BY (M+P+C)/3 DESC,M DESC,P DESC

where & order by :-

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

=> dispay employees working as clerk,manager and arrange list sal wise desc order ?

SELECT EMPNO,ENAME,JOB,SAL

FROM EMP

WHERE JOB IN ('CLERK','MANAGER')

ORDER BY SAL DESC

NOTE :-

=> in sorting nulls treated low

=> in ascending order nulls arragned first

=> in descending order nulls arranged last

DISTINCT clause :-

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

=> distinct clause eliminates duplicates from select stmt output

DISTINCT col

DISTINCT col1,col2,--

DISTINCT \*

SELECT DISTINCT JOB FROM EMP

ANALYST

CLERK

MANAGER

PRESIDENT

SALESMAN

SELECT DISTINCT DEPTNO FROM EMP

10

20

30

TOP clause :-

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

=> used to select top N rows from table

SELECT TOP <n> colnames/\*

FROM tabname

[WHERE cond]

[ORDER BY colname ]

=> display first 5 rows from emp ?

SELECT TOP 5 EMPNO,ENAME,SAL

FROM EMP

=> diplay top 5 highest paid employees ?

SELECT TOP 5 EMPNO,ENAME,SAL

FROM EMP

ORDER BY SAL DESC

=> display top 5 employees based on experience ?

SELECT TOP 5 EMPNO,ENAME,SAL,HIREDATE

FROM EMP

ORDER BY HIREDATE ASC

=> display top 3 max salaries ?

SELECT DISTINCT TOP 3 SAL

FROM EMP

ORDER BY SAL DESC

DML commands :- (Data Manipulation Lang)

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

INSERT

UPDATE

DELETE

MERGE

=> DML commands acts on table data.

=> all DML commands are auto committed (saved).

=> to stop auto commit execute the following command

SET IMPLICIT\_TRANSACTIONS ON

=> to save operations execute COMMIT command

=> to cancel operations execute ROLLBACK command

UPDATE command :-

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

=> command used to modify the table data.

=> we can update all rows or specific rows

=> we can update single column or multiple columns

syn :-

UPDATE tabname

SET colname = value , colname = value , ---------

[WHERE condition]

ex :-

=> update all employees comm with 500 ?

UPDATE emp SET comm = 500

=> update employees comm with 500 whose comm = null ?

UPDATE emp SET comm = 500 WHERE comm IS NULL

=> update comm with null whose comm <> null ?

UPDATE emp SET comm = NULL WHERE comm IS NOT NULL

NULL assignment =

NULL comparision IS

=> update sal with 2000 and comm with 500 whose id = 7369 ?

UPDATE emp SET sal = 2000 , comm = 500 WHERE empno = 7369

=> increment sal by 20% and comm by 10% those working as salesman

and joined in 1981 year ?

UPDATE emp

SET sal = sal + (sal\*0.2) , comm = comm + (comm\*0.1)

WHERE job='SALESMAN'

AND

hiredate LIKE '1981%'

=>

PRODUCTS

prodid pname price category brand

increase the price of all samsung,realme,oneplus brand mobiles phones by 10% ?

UPDATE PRODUCTS

SET PRICE = PRICE + (PRICE\*0.1)

WHERE BRAND IN ('SAMSUNG','REDMI','REALME')

AND

CATEGORY='MOBILES'

17-MAY-23

DELETE command :-

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

=> command used to delete row/rows.

=> we can delete all rows or specific rows.

syn :- DELETE FROM <TABNAME> [WHERE COND]

ex :-

=> delete all rows from emp table ?

DELETE FROM EMP

=> delete employees joined in 1980 ?

DELETE FROM EMP WHERE HIREDATE LIKE '1980%'

DDL commands :- (DATA DEFINITION LANG)

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

CREATE

ALTER

DROP

TRUNCATE

=> all DDL commands acts on table structure (columns,datatype and size),

ALTER command :-

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

=> command used to modify table structure

=> using ALTER command we can

1 add columns

2 drop columns

3 modify a column

changing datatype

changing size

Adding columns :-

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

ALTER TABLE <TABNAME>

ADD COLNAME DATATYPE(SIZE) , COLNAME DATATYPE(SIZE),------

EX :-

=> add column gender to emp table ?

ALTER TABLE EMP

ADD GENDER CHAR(1)

after adding by default the new column is filled with nulls , to insert data into

the new column use update command.

UPDATE EMP SET GENDER='M' WHERE EMPNO = 7499

Droping column :-

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

ALTER TABLE <TABNAME>

DROP COLUMN COL1,COL2,------

EX :-

=> drop column gender from emp table ?

ALTER TABLE EMP

DROP COLUMN GENDER

Modifying a column :-

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

ALTER TABLE <TABNAME>

ALTER COLUMN COLNAME DATATYPE(SIZE)

EX :-

=> increase size of ename to 20 ?

ALTER TABLE EMP

ALTER COLUMN ENAME VARCHAR(20)

ALTER TABLE EMP

ALTER COLUMN ENAME VARCHAR(5) => ERROR => some names contains

more than 5 chars

=> change the datatype of empno to int ?

ALTER TABLE EMP

ALTER COLUMN EMPNO INT

ALTER TABLE EMP

ALTER COLUMN EMPNO TINYINT => ERROR

DROP command :-

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

=> command drops table from db.

=> drops table structure along with data.

SYN :- DROP TABLE <TABNAME>

EX :- DROP TABLE EMP

TRUNCATE :-

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

=> deletes all the data from table but keeps structure.

=> will empty the table

=> releases memory allocated for table

=> when truncate is executed sql server goes to memory and releases all the pages

allocated for table and when pages are released then data stored in the pages also

deleted.

SYN :- TRUNCATE TABLE <TABNAME>

EX :- TRUNCATE TABLE EMP

DROP VS DELETE VS TRUNCATE :-

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

DROP DELETE/TRUNCATE

drops table structure with data deletes only data but not structure

DELETE VS TRUNCATE :-

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

DELETE TRUNCATE

1 DML DDL

2 we can delete specific rows can delete all rows but cannot delete

specific rows

3 where cond can be used with where cond cannot be used with truncate

delete

4 deletes row-by-row deletes all rows at a time

5 slower faster

6 will not release memory releases memory

7 will not reset identity will reset identity

SP\_RENAME :- (SP => STORED PROCEDURE)

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

=> used to change tablename and column name

SP\_RENAME 'OLDNAME', 'NEWNAME'

EX :- rename table emp to employees ?

SP\_RENAME 'EMP','EMPOYEES'

rename column comm to bonus ?

SP\_RENAME 'EMPOYEES.COMM','BONUS'

Built-in Functions in SQL SERVER :-

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

=> a functions accepts some input performs some calculation and returns one value.

Types of functions :-

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

1 DATE

2 STRING

3 NUMERIC

4 CONVERSION

5 SPECIAL

6 ANALYTICAL

7 AGGREGATE

DATE functions :-

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

1 GETDATE() :-

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

=> returns current date , time and milliseconds

SELECT GETDATE() => 2023-05-18 11:26:41.820

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

DATE TIME MS

2 DATEPART() :-

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

=> used to extract part of the date

DATEPART(INTERVAL , DATE)

SELECT DATEPART(YY,GETDATE()) => 2023

MM => 05

DD => 18

DW => 5 ( 1 SUN ---- 7 SAT)

HH => hour part

MI => minutes

SS => seconds

Q => quater (1-4)

1 jan-mar

2 apr-jun

3 jul-sep

4 oct-dec

DAYOFYEAR(DY) => 138 (day number in year)

=> employees joined in 1980,1983,1985 ?

SELECT \*

FROM EMP

WHERE DATEPART(YY,HIREDATE) IN (1980,1983,1985)

=> employees joined in leap year ?

SELECT \*

FROM EMP

WHERE DATEPART(YY,HIREDATE) % 4 = 0

=> who are joined in jan,apr,dec months ?

SELECT \*

FROM EMP

WHERE DATEPART(MM,HIREDATE) IN (1,4,12)

=> who are joined on sunday ?

SELECT \*

FROM EMP

WHERE DATEPART(DW,HIREDATE) = 1

=> who are joined in 2nd quarter of 1981 year ?

SELECT \*

FROM EMP

WHERE DATEPART(YY,HIREDATE) = 1981

AND

DATEPART(Q,HIREDATE) = 2

DATENAME() :-

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

=> used to extract part of the date

MM DW

DATEPART 5 5

DATENAME MAY THURSDAY

=> display ENAME JOIN DAY ?

SELECT ENAME,DATENAME(DW,HIREDATE) AS DAY

FROM EMP

=> write a query to display on which day india got independence ?

SELECT DATENAME(DW,'1947-08-15') => Friday

FORMAT() :-

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

=> used to display dates in different formats

FORMAT( date , 'format' )

SELECT FORMAT(GETDATE(),'dd-MM-yyyy') => 18-05-2023

=> DISPLAY ENAME HIREDATE ? DISPLAY HIREDATES IN MM/DD/YYYY FORMAT ?

SELECT ENAME,FORMAT(HIREDATE,'MM/dd/yyyy') AS HIREDATE FROM EMP

SELECT FORMAT(GETDATE(),'dd-MM-yyyy hh:mm:ss') => 18-05-2023 12:17:25

scenario :-

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

INSERT INTO emp(empno,ename,sal,hiredate)

VALUES(9999,'ABC',5000,getdate())

=> list of employees joined today ?

SELECT \*

FROM EMP

WHERE HIREDATE = GETDATE() => NO ROWS

2023-05-18 = 2023-05-18 12:25:20.123

"=" comparision with GETDATE() always fails to overcome this problem use FORMAT function

SELECT \*

FROM EMP

WHERE HIREDATE = FORMAT(GETDATE(),'yyyy-MM-dd')

2023-05-18 = 2023-05-18

DATEDIFF() :-

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

=> used to find difference between two dates

DATEDIFF(INTERVAL,START DATE,END DATE)

SELECT DATEDIFF(YY,'2022-05-18',GETDATE()) => 1

SELECT DATEDIFF(MM,'2022-05-18',GETDATE()) => 12

SELECT DATEDIFF(DD,'2022-05-18',GETDATE()) => 365

=> display ENAME EXPERIENCE in years ?

SELECT ENAME,DATEDIFF(YY,HIREDATE,GETDATE()) AS EXPERIENCE

FROM EMP

=> display ENAME EXPERIENCE ?

M YEARS N MONTHS

experience = 40 months = 3 years 4 months

years = months/12 = 40/12 = 3 years

months = months%12 = 40%12 = 4 months

SELECT ENAME,

DATEDIFF(MM,HIREDATE,GETDATE())/12 AS YEARS,

DATEDIFF(MM,HIREDATE,GETDATE())%12 AS MONTHS

FROM EMP

1980-12-17 => 2022-12-17 => 42

2023-01-17 1

02

03

04

05

19-may-23

DATEADD() :-

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

=> used to add / subtract days,months,years to/from a date.

DATEADD(INTERVAL , INT , DATE)

SELECT DATEADD(DD,10,GETDATE()) => 2023-05-29

SELECT DATEADD(MM,2,GETDATE()) => 2023-07-19

SELECT DATEADD(YY,-1,GETDATE()) => 2022-05-19

scenario :-

GOLD\_RATES

DATEID RATE

2020-01-01 ?

2020-01-02 ?

2023-05-19 ?

=> display today's gold rate ?

=> display yesterday's gold rate ?

=> display last month same day gold rate ?

=> display last year same day gold rate ?

1 SELECT \*

FROM GOLD\_RATES

WHERE DATEID = FORMAT(GETDATE(),'yyyy-MM-dd')

2 SELECT \*

FROM GOLD\_RATES

WHERE DATEID = FORMAT(DATEADD(DD,-1,GETDATE()),'yyyy-MM-dd')

3 SELECT \*

FROM GOLD\_RATES

WHERE DATEID = FORMAT(DATEADD(MM,-1,GETDATE()),'yyyy-MM-dd')

4 SELECT \*

FROM GOLD\_RATES

WHERE DATEID = FORMAT(DATEADD(YY,-1,GETDATE()),'yyyy-MM-dd')

=> display last 1 month gold rates ?

SELECT \*

FROM GOLD\_RATES

WHERE DATEID BETWEEN DATEADD(MM,-1,GETDATE()) AND GETDATE()

=> first day in sql server calendar ?

SELECT DATEADD(DD,0,0) => 1900-01-01

EOMONTH() :-

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

=> returns last day of the month

EOMONTH(DATE,INT)

SELECT EOMONTH(GETDATE(),0) => 2023-05-31

SELECT EOMONTH(GETDATE(),1) => 2023-06-30

SELECT EOMONTH(GETDATE(),-1) => 2023-04-30

1 display next month first day ?

2 display current month first day ?

3 display next year first day ?

4 display current year first day ?

STRING FUNCTIONS :-

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

UPPER() :-

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

=> used to convert string to uppercase

UPPER(string)

SELECT UPPER('hello') => HELLO

LOWER() :-

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

=> used to convert string to lowercase

LOWER(string)

SELECT LOWER('HELLO') => hello

=> display EMPNO ENAME SAL ? display names in lowercase ?

SELECT EMPNO,LOWER(ENAME) AS ENAME,SAL FROM EMP

=> convert names to lowercase in table ?

UPDATE EMP SET ENAME = LOWER(ENAME)

LEN() :-

-----------

=> returns string length i.e. no of chars

LEN(string)

SELECT LEN('hello') => 5

=> display ENAME LENGTH ?

SELECT ENAME,LEN(ENAME) AS LENGTH FROM EMP

=> display employees name contains 4 chars ?

SELECT \* FROM EMP WHERE ENAME LIKE '\_\_\_\_'

SELECT \* FROM EMP WHERE LEN(ENAME)=4

LEFT() :-

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

=> returns characters starting from left

LEFT(STRING,LEN)

SELECT LEFT('hello welcome',5) => hello

=> employees name starts with 's' ?

SELECT \* FROM EMP WHERE LEFT(ENAME,1)='s'

RIGHT() :-

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

=> returns characters starting from right

RIGHT(STRING,LEN)

SELECT RIGHT('hello welcome',7) => welcome

=> employees name ends with 's' ?

SELECT \* FROM EMP WHERE RIGHT(ename,1)='s'

=> employees name starts and ends with same char ?

SELECT \* FROM EMP WHERE ENAME LIKE 'a%a'

OR

ENAME LIKE 'b%b'

SELECT \* FROM EMP WHERE LEFT(ENAME,1) = RIGHT(ENAME,1)

SUBSTRING() :-

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

=> returns characters starting from specific position

SUBSTRING(string,start,len)

SELECT SUBSTRING('hello welcome',7,4) => welc

SELECT SUBSTRING('hello welcome',10,3) => com

scenario :-

=> generate emailids for employees ?

EMPNO ENAME EMIALID

7369 smith smi736@tcs.com

7499 allen all749@tcs.com

'a' + 'b' => ab

'a'+ ' ' + 'b' => a b

SELECT EMPNO , ENAME ,

LEFT(ENAME,3) + LEFT (EMPNO,3) + '@tcs.com' AS EMAILID

FROM EMP

=> store emailids in db ?

STEP 1 :- add emailid column to emp table

ALTER TABLE EMP

ADD EMAILID VARCHAR(30)

STEP 2 :- update the column with emaiids

UPDATE EMP

SET EMAILID = LEFT(ENAME,3) + LEFT (EMPNO,3) + '@tcs.com'

20-may-23

CHARINDEX() :-

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

=> returns position of a character in a string

CHARINDEX(CHAR,STRING,[START])

SELECT CHARINDEX('O','HELLO WELCOME') => 5

SELECT CHARINDEX('X','HELLO WELCOME') => 0

SELECT CHARINDEX('O','HELLO WELCOME',6) => 11

Assignment :-

CUST

CID NAME

10 SACHIN TENDULKAR

11 ROHIT SHARMA

display CID FNAME LNAME ?

10 SACHIN TENDULKAR

using :- LEFT,RIGHT,SUBSTRING,CHARINDEX

REPLICATE() :-

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

=> used to repeat char for given no of times

REPLICATE(CHAR,LEN)

SELECT REPLICATE('\*',5) => \*\*\*\*\*

Display ENAME SAL ?

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

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

SELECT ENAME,REPLICATE('\*',LEN(SAL)) AS SAL FROM EMP

scenario :-

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

ACCOUNTS

ACCNO ACTYPE BAL

12345678962 S 10000

your a/c no XXXX8962 debited ---------

REPLICATE('X',4) + RIGHT(ACCNO,4)

REPLACE() :-

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

=> used to replace one string with another string.

REPLACE(str1,str2,str3)

=> in str1 , str2 replaced with str3

SELECT REPLACE('hello','ell','abc') => habco

SELECT REPLACE('hello','l','abc') => heabcabco

SELECT REPLACE('hello','elo','abc') => hello

SELECT REPLACE('@@he@@ll@@o@@','@','') => hello

STUFF() :-

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

=> similar to replace i.e. used to replace one string with another string

=> it is based on start and length

STUFF(string1,start,length,string2)

SELECT STUFF('HELLO WELCOME',10,3,'ABC') => HELLO WELABCE

SELECT STUFF('A,B,C,D,',8,1,'') => A,B,C,D

TRANSLATE() :-

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

=> used to translate one char to another char

TRANSLATE(str1,str2,str3)

SELECT TRANSLATE('HELLO','ELO','ABC') => HABBC

E => A

L => B

O => C

SELECT TRANSLATE('HELLO','ELO','') => ERROR

NOTE :- TRANSLATE function can be used to encrypt data i.e. changing plain text to cipher text

DISPLAY ENAME SAL ?

SELECT ENAME,

TRANSLATE(SAL , '0123456789.','$bT\*k@G^#%&') AS SAL

FROM EMP

JONES 2975.00 T%^@&$$

=> remove all special chars from '@#HE%$LL&^O\*@' ?

output :- hello

TRANSLATE( '@#HE%$LL&^O\*@','@#%$&^\*','')

STEP 1 :- translating all special chars to one special char

TRANSLATE( '@#HE%$LL&^O\*@','@#%$&^\*','\*\*\*\*\*\*\*') => \*\*HE\*\*LL\*\*O\*

STEP 2 :- replace '\*' with null

SELECT

REPLACE(TRANSLATE( '@#HE%$LL&^O\*@','@#%$&^\*','\*\*\*\*\*\*\*'),'\*','') => HELLO

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

\*\*HE\*\*LL\*\*O\*

Numeric Functions :-

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

ABS() :- reurns absolute value

ABS(NUMBER)

SELECT ABS(10) => 10

SELECT ABS(-10) => 10

POWER() :- calculates power

POWER(num1,num2)

SELECT POWER(3,2) => 9

SQRT() :- returns square root

SQRT(number)

SELECT SQRT(16) => 4

SQUARE() :- returns square of a number

SQUARE(number)

SELECT SQUARE(5) => 25

SIGN() :- to check whether given number is positive or negative

SIGN(number)

SELECT SIGN(10) => 1

SIGN(-10) => -1

SIGN(0) => 0

22-may-23

Rounding numbers :-

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

38.3456789234 => 38

38.34

38.3456

ROUND

CEILING

FLOOR

ROUND() :-

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

=> rounds number to integer or to decimal places.

=> round function acts according to average.

ROUND(number,decimal places)

SELECT ROUND(38.4567,0) => 38

38-----------------------------38.5------------------------------39

number < avg => rounded to lowest

number >= avg => rounded to highest

SELECT ROUND(38.5567,0) => 39

SELECT ROUND(38.4567,2) => 38.46

SELECT ROUND(38.4547,2) => 38.45

SELECT ROUND(356,-2) => 400

300--------------------350-------------------------400

SELECT ROUND(356,-1) => 360

350----------------------355---------------------------360

SELECT ROUND(356,-3) => 0

0--------------------------500---------------------------1000

Question :-

SELECT ROUND(4567,-1),ROUND(4567,-2),ROUND(4567,-3)

o/p :- 4570 4600 5000

CEILING() :-

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

=> rounds number always to highest

CEILING(number)

SELECT CEILING(38.1) => 39

38---------------------------------------------39

FLOOR() :-

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

=> rounds number always to lowest

SELECT FLOOR(38.9) => 38

Conversion functions :-

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

=> used to convert one datatype to another datatype

1 CAST

2 CONVERT

CAST :-

-----------

CAST(SOURCE-EXPR AS TARGET-TYPE)

SELECT CAST(10.5 AS INT) => 10

=> display SMITH earns 800 ?

ALLEN earns 1600

SELECT ENAME + ' earns ' + SAL FROM EMP => ERROR

SELECT ENAME + ' earns ' + CAST(SAL AS VARCHAR) FROM EMP

=> display smith joined on 1980-12-17 as clerk ?

SELECT ename + ' joined on ' + CAST(hiredate as varchar) + ' as ' + job

FROM emp

CONVERT() :-

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

CONVERT(TARGET-TYPE,SOURCE-EXPR)

SELECT CONVERT(INT,10.5) => 10

=> difference between CAST & CONVERT ?

1 using convert function we can display dates in different formats but not possible using

cast function

2 using convert we can display money in different formats but not possible using cast

Date Styles :-

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

CONVERT(VARCHAR,DATE,STYLE-NUMBER)

EX :-

SELECT CONVERT(VARCHAR,GETDATE(),101) => 05/22/2023

102 => 2023.05.22

103 => 22/05/2023

114 => 12:11:41:240

display ENAME HIREDATE ?

display hiredates in DD-MM-YYYY format ?

SELECT ENAME,CONVERT(VARCHAR,HIREDATE,105) AS HIREDATE

FROM EMP

SELECT DATEADD(DD,5,'10/5/2023') => 2023-10-10 00:00:00.000

SELECT DATEADD(DD,5,CONVERT(DATE,'10/5/2023',103)) => 2023-05-15

without convert 10/5/2023 => mm/dd/yyyy

with convert 10/5/2023 => dd/mm/yyyy

Money Styles :-

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

CONVERT(VARCHAR,MONEY,STYLE-NUMBER)

SELECT ENAME,CONVERT(VARCHAR,SAL,1) AS SAL FROM EMP

CONVERT(VARCHAR,SAL,0) => 1600.00

CONVERT(VARCHAR,SAL,1) => 1,600.00

CONVERT(VARCHAR,SAL,2) => 1600.0000

SELECT CONVERT(VARCHAR,CAST(5000 AS MONEY),1) => 5,000.00

Special Functions :-

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

ISNULL() :-

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

=> used to convert null values

ISNULL(arg1,arg2)

if arg1 = null returns arg2

if arg1 <> null returns arg1 only

SELECT ISNULL(100,200) => 100

SELECT ISNULL(NULL,200) => 200

display ENAME SAL COMM TOTSAL ?

SELECT ENAME,SAL,COMM,SAL+COMM AS TOTSAL FROM EMP

smith 800.00 NULL NULL

allen 1600.00 300.00 1900.00

SELECT ENAME,SAL,COMM,SAL+ISNULL(COMM,0) AS TOTSAL FROM EMP

smith 800.00 NULL 800.00

allen 1600.00 300.00 1900.00

23-may-23

display ENAME SAL COMM ?

if comm = NULL display NO COMM ?

SELECT ENAME,SAL,

ISNULL(CAST(COMM AS VARCHAR),'NO COMM') AS COMM

FROM EMP

Analytical Functions / Window Functions :-

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

RANK,DENSE\_RANK :-

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

=> both functions are used to find ranks

=> ranking is always based on some column like sal,hiredate etc

=> for rank functions data must be sorted

RANK() OVER (ORDER BY COLNAME ASC/DESC)

DENSE\_RANK() OVER (ORDER BY COLNAME ASC/DESC)

Ex :-

=> find the ranks of the employees based on sal and highest paid employee

should get 1st rank ?

SELECT ENAME,SAL,

RANK() OVER (ORDER BY SAL DESC) AS RNK

FROM EMP

SELECT ENAME,SAL,

DENSE\_RANK() OVER (ORDER BY SAL DESC) AS RNK

FROM EMP

=> difference between rank & dense\_rank ?

1 rank function generates gaps but dense\_rank will not generate gaps.

2 in rank function ranks many not be in sequence but in dense\_rank

ranks will be always in sequence.

SAL RANK DENSE\_RANK

5000 1 1

4000 2 2

3000 3 3

3000 3 3

3000 3 3

2000 6 4

2000 6 4

1000 8 5

=> display ranks of the employees based on hiredate ?

SELECT ENAME,HIREDATE,

DENSE\_RANK() OVER (ORDER BY HIREDATE ASC) AS RNK

FROM EMP

=> display ranks of the employees based on sal , if salaries are same then

ranking should be based on hiredate ?

SELECT ENAME,HIREDATE,SAL,

DENSE\_RANK() OVER (ORDER BY SAL DESC,HIREDATE ASC) AS RNK

FROM EMP

PARTITION BY clause :-

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

=> used to find ranks with in group for ex with in dept

=> to find ranks with in dept first we need to divide the table dept wise

and apply dense\_rank function on each dept instead of applying it on

whole table.

SELECT ENAME,HIREDATE,SAL,DEPTNO,

DENSE\_RANK() OVER (PARTITION BY DEPTNO

ORDER BY SAL DESC) AS RNK

FROM EMP

ROW\_NUMBER() :-

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

=> returns record number

=> row number is also based on some column

=> for row number also data must be sorted

ROW\_NUMBER() OVER (ORDER BY COLNAME ASC/DESC,---)

Ex :-

SELECT EMPNO,ENAME,SAL,

ROW\_NUMBER() OVER (ORDER BY EMPNO ASC) AS RNO

FROM EMP

LAG() & LEAD() :-

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

LAG(COLNAME,INT) OVER (ORDER BY COLNAME) => returns previous value

LEAD(COLNAME,INT) OVER (ORDER BY COLNAME) => returns next value

example 1 :-

SELECT EMPNO,ENAME,HIREDATE,SAL,

LAG(SAL,1) OVER (ORDER BY HIREDATE ASC) AS PREV\_SAL

FROM EMP

7369 smith 1980-12-17 800.00 NULL

7499 allen 1981-02-20 1600.00 800.00

7521 ward 1981-02-22 1250.00 1600.00

Example 2 :-

=> display ENAME HIREDATE PREV\_HIRE DAYS ?

SELECT EMPNO,ENAME,HIREDATE,

LAG(HIREDATE,1) OVER (ORDER BY HIREDATE ASC) AS PREV\_HIRE,

DATEDIFF(DD,

LAG(HIREDATE,1) OVER (ORDER BY HIREDATE ASC),

HIREDATE) AS DAYS

FROM EMP

Example 3 :-

SELECT EMPNO,ENAME,SAL,

LEAD(SAL,1) OVER (ORDER BY SAL ASC) AS PREV\_SAL

FROM EMP

24-may-23

Aggregate Functions :-

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

=> these functions takes group of rows and returns one value

MAX() :-

-----------

=> returns maximum value

MAX(arg)

SELECT MAX(SAL) FROM EMP => 5000

SELECT MAX(HIREDATE) FROM EMP => 1983-01-12

SELECT MAX(ENAME) FROM EMP => ward

MIN() :-

---------

=> returns minimum value

MIN(arg)

SELECT MIN(SAL) FROM EMP => 800

SUM() :-

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

=> returns total

SUM(arg)

SELECT SUM(sal) FROM EMP => 29025.00

=> round total sal to hundreds ?

SELECT ROUND(SUM(sal),-2) FROM EMP => 29000

29000--------------29050 -----------------29100

=> after rounding display total sal with thousand seperator ?

SELECT CONVERT(VARCHAR,ROUND(SUM(sal),-2),1) FROM EMP

o/p :- 29,000.00

=> display total sal paid to managers ?

SELECT SUM(sal) FROM EMP WHERE JOB='MANAGER'

=> display total sal including comm ?

SELECT SUM(SAL+COMM) AS TOTSAL FROM EMP => 7800.00

SAL COMM SAL+COMM

5000 NULL NULL

4000 500 4500

3000 NULL NULL

SUM(SAL) = 12000

SUM(SAL+COMM) = 4500

SAL COMM SAL+ISNULL(COMM,0)

5000 NULL 5000

4000 500 4500

3000 NULL 3000

SUM(SAL) = 12000

SUM(SAL+ISNULL(COMM,0)) = 12500

SELECT SUM(SAL+ISNULL(COMM,0)) AS TOTSAL FROM EMP => 31225

AVG() :-

-----------

=> returns avg value

AVG(arg)

SELECT AVG(SAL) FROM EMP => 2073.2142

=> round avg(sal) to lowest ?

SELECT FLOOR(AVG(SAL)) FROM EMP => 2073

NOTE :- SUM,AVG cannot be applied on char,date columns can be applied only on numeric

columns

COUNT() :-

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

=> returns no of values present in a column

syn :- COUNT(arg)

SELECT COUNT(EMPNO) FROM EMP => 14

SELECT COUNT(COMM) FROM EMP => 4

COUNT(\*) :-

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

=> returns no of rows in a table.

SELECT COUNT(\*) FROM EMP => 14

T1

F1

10

NULL

20

NULL

30

COUNT(F1) => 3 (nulls are not counted)

COUNT(^) => 5 (nulls are counted)

=> how many employees joined in 1981 year ?

SELECT COUNT(\*) FROM EMP WHERE DATEPART(YY,HIREDATE) = 1981

=> how many employees joined on sunday ?

SELECT COUNT(\*)

FROM EMP

WHERE DATENAME(DW,HIREDATE)='SUNDAY'

=> how many employees joined in 2nd quarter of 1981 year ?

SELECT COUNT(\*)

FROM EMP

WHERE DATEPART(YY,HIREDATE)=1981

AND

DATEPART(Q,HIREDATE) = 2

NOTE :-

=> aggregate functions are not allowed in where clause and they are allowed only in

SELECT,HAVING clauses

SELECT ENAME

FROM EMP

WHERE SAL = MAX(SAL) => ERROR

WHERE COUNT(\*) = 3 => ERROR

summary :-

DATE :- datepart,datename,dateadd,datediff,eomonth

STRING :- upper,lower,len,left,right,substring,charindex,replicate,replace,translate,stuff

NUMERIC :- abs,power,sqrt,square,sign,round,ceiling,floor

CONV :- cast, convert

SPECIAL :- isnull

ANALYTICAL :- rank,dense\_rank,row\_number,lag,lead

AGGREGATE :- max,min,sum,avg,count,count(\*)

CASE statement :-

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

=> case stmt is similar to switch case

=> used to implement if-then-else

=> used to return values based on condition

=> case statements are 2 types

1 simple case

2 searched case

simple case :-

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

=> use simple case when conditions based on "=" operator

CASE COLNAME

WHEN VALUE1 THEN RETURN EXPR1

WHEN VALUE2 THEN RETURN EXPR2

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

[ELSE RETURN EXPR]

END

=> display ENAME DNAME ?

IF DEPTNO=10 DISPLAY ACCOUNTS

20 RESEARCH

30 SALES

SELECT ENAME,

CASE DEPTNO

WHEN 10 THEN 'ACCOUNTS'

WHEN 20 THEN 'RESEARCH'

WHEN 30 THEN 'SALES'

ELSE 'UNKNOWN'

END AS DNAME

FROM EMP

=> increment employee salaries as follows ?

if job=CLERK incr sal by 10%

SALESMAN 15%

MANAGER 20%

OTHERS 5%

UPDATE EMP

SET SAL = CASE JOB

WHEN 'CLERK' THEN SAL + (SAL\*0.1)

WHEN 'SALESMAN' THEN SAL + (SAL\*0.15)

WHEN 'MANAGER' THEN SAL + (SAL\*0.2)

ELSE SAL + (SAL\*0.05)

END

25-may-23

searched case :-

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

=> use searched case when conditions not based on "=" operator

CASE

WHEN COND1 THEN RETURN EXPR1

WHEN COND2 THEN RETURN EXPR2

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

[ELSE RETURN EXPR]

END

DISPLAY ENAME SAL SALRANGE ?

IF SAL>3000 HISAL

SAL<3000 LOSAL

SAL=3000 AVGSAL

SELECT ENAME,SAL,

CASE

WHEN SAL>3000 THEN 'HISAL'

WHEN SAL<3000 THEN 'LOSAL'

ELSE 'AVGSAL'

END AS SALRANGE

FROM EMP

=> display SNO TOTAL AVG RESULT ?

STUDENT

SNO SNAME S1 S2 S3

1 A 80 90 70

2 B 30 60 50

SELECT SNO,

S1+S2+S3 AS TOTAL,

(S1+S2+S3)/3 AS AVG,

CASE

WHEN S1>=35 AND S2>=35 AND S3>35 THEN 'PASS'

ELSE 'FAIL'

END AS RESULT

FROM STUDENT

=>

SELECT SUM(CASE JOB

WHEN 'CLERK' THEN SAL

END) AS CLERK,

SUM(CASE JOB

WHEN 'MANAGER' THEN SAL

END) AS MANAGER

FROM EMP

CLERK MANAGER

4565 9930