Database

computare (Latin word) -> to compute/calculate

Charles Babbage

Job of Computer

```
(input)         (processing)
Data  ->        Computer ->           (output)
(raw facts)                          Information
(meaningless)                        (meaningful data)
22021984                             (data on whose basis the management can take some decision or you can take
some action)
```

processing -> work done by the computer to convert data into information

Database -> Database is a collection of LARGE amounts of data

DBMS -> Database Management System
DBMS ->readymade s/w that helps you to mannage your data
ANSI defination (American National Standards Institute)
collection of programs that allows you to Insert ,Update,Delete and Process.

e.g. MS Excel, dBase,Foxbase,Numbers,Google Spreadsheets,Foxpro,Dataease,Dataflex,DB vista Advanced Revelation etc.

Mysql -> RDBMS (Relational Database Managment System)

DBMS vs RDBMS
DBMS (e.g MS Excel, Foxpro,etc.)
----
a. Field
b. Record
c. File
1. Naming conventions(Nomenclature)
2. Relationship between 2 files is maintained programatically
3. More Programming
4. More time is required for s/w development
5. High Network traffic
6. Processing is on Client machine
7. Processing is on Client machine
8. Client-Server architecture is not supported.
9. File level looking
10. Not suitable for multi-user
11. Distributed Database are not supported
12. No security of Data
DBMS allows acess to the file through the OS
===
======
RDBMS (e.g Mysql, Oracle etc.)

----
a. Column, Attribute
b. Row, Tuple, Entity
c. Table, Relation, Entity class
2.Relationship between 2 tables can be specified at the time of table creation (e.g Foreign key constraint)
3. Less programming
4. Less time is required for s/w development
5.Low network traffic
6.Faster (in terms of network speed) and cheaper(in terms of hardware coste, network cost, infrastructure cost)
7.Processing is on Server machine (known client server architecture)
8. Most of the RDBMS support Client-Server architecture (eg. MySQL and Oracle support ,)
9. Row level locking (Table is not a file ; internally every row is a file)
10. suitable for multiuser
11. Most of the RDBMS support Distributed Databases
12.Multiple levels of security
a. Logging in security
(MySQL username and password)
b. Command level security
(to issue MySQL commands)
e.g. create table, create function, create procedure, create view, create user etc.
c. Object level security
(to access the tables and other objects of other users)

RDBMS DOES NOT allow acess to the table through the OS

various RDBMS available:-
#Informix (fastest interms of processing speed)

#Oracle (slowest interms of processing speed)
* most popular because theprogramming is very easy
* product of Oracle Corporation
* largest s/w company in the world
* largest overall s/w company in the world
* 63% of world commertial database market in the client server environment
* 86% of world commertial database market in the internet environment
*more than 90% of Fortune 500 companies in the world use Oracle
* 10/10 of the largest companies in the world use Oracle
* available on 113 OS
# Sybase
* going down
* Sybase has recently acquired by SAP

# MS SQL Server
* good RDBMS from Microsoft
* 16% commertial database market share
* It only works with windows OS

Ingres
Postgres
Unify

# Server has to be a mainframe (super computer):
DB2
* good RDBMS from IBM
CICS
TELON
IDMS

#Single user RDBMS
MS Access
Paradox
Vatcom SQL
* Personal Oracle (single use version of Oracle) //Free RDBMS

#MySQL
* MySQL was launched by a Swedish company in 1995
* its name is a combination of "My", the name of co-founder Michel Widenius 'daughter',
* Widenius'daughter, and "SQL"
* MySQL is an open source RDBMS
* most widely used open-source RDBMS
* part of the widely used LAMP open source web application software stack(and other "AMP" stacks)
L -> Linux
A ->Apache
M -> Mysql
p -> perl or python or PHP
* Facebook,Joomla,WordPress, Twitter ,Flicker , Instagram ,Google(though not for searches), Youtube etc.
* free software open source projects that require a full-featured RDBMS often use MySQL
* MySQL occupies 42 of world open-source database market
* sun Microsystems acquired MySQL in 2008
* Oracle acquired sun Microsystems in 2010

DEPT

DEPTNO	DNAME	LOC
10	TRN	Bombay
20	EXP	Delhi

EMP

EMPNO	ENAME	SAL	DEPTNO
1	POOJA	4000	10
2	OJAS	3000	20

* Facebook is currently in process of migratin from MySQL to mangoDB

# Various s/w development tools from MySQL:
MySQL database
* database server s/w
* store table data, retrieve table data , secure table data etc.

SQL
*Structured Query Language

* Create, Drop ,Alter
Insert,Update,Delete
Grant, Revoke , Select
* conforms to ANSI standards (e.g 1 character = 1 Byte, ANSI datatypes, char ,int ,etc.)
* conforms to ISO standards (for quality Assurance)
* common for all RDBMS
* initially founded by IBM (1975-77)
* earlier known as RQBE (Relational Query by Example) old name of SQL
* IBM gave RQBE to ANSI
* now controlled by ANSI (hence common for all RDBMS)
* ANSI renamed RQBE to SQL

DR. Codd founder of database.

SQL source code:
90% in C C++
10% in assembly
*wrote the source code of SQL
1.Larry
2.Thomos
3.Scott
4.Chris
5.Huma
6.Gavin
this 6 person started Oracle after SQL source code (Greek word) some one will knows future

MySQL PL
* MySQL Programming Language
* programming languge from MySQL
* used for database programming
eg. HRA_CALC, TAX_CALC ,ATTENDANCE_CALC, etc.

MySQL Command Line Client
* MySQL Client s/w
* character based(text based)
* used for running SQL commands , MySQL PL programs and MySQL commands
* interface with database

MySQL Workbench
* MySQL Client s/w
*GUI based(Graphical User Interface)
*used for running SQL commands, MySQL PL programs , and MySQL commands interface with database

MySQL Connectors
* for MySQL database connectivity
*JDBC drivers for JAVA,ODBC drivers for Python C,C++ etc.

MySQL for Excel
* import , export and edit MySQL data using Microsoft Excel.

MySQL Notifier
* Stratup and Shutdown Database

MySQL Enterprise Backup
* used to backup table data

MySQL Enterprise High Availability
* for Replication (for standby database)

MySQL Enterprise Encryption
*to encrypt the table data

MySQL Enterprise Monitor
* for Performance Tuning (fo Performance Managment)

MySQL Query Analyzer
* to analyze and speed up the queries

etc.

# 4 sub divisions of SQL :-
DDL (Data Defination Language) (Create , Drop , Alter)
DML (Data Manupulation Language) (Insert , Update , Delete)
DCL (Data Control Language) (Grant , Revoke)
DQL (Data Query Language) (Select)

5th Component of SQL :-
Not an ANSI standard ;-
Extra in MySQL RDBMS and Oracle RDBMS :-

DTL/TCL (Data Transaction Language) /(Transaction Control Language)
(Commit , Rollback , Savepoint)
DDL(Rename ,Truncate)

Extra in Oracle RDBMS ONLY :-
DML (Merge ,Upsert)

# RULES FOR tablenames , columnname ,variablenames : -

* max 30 characters
* A - Z , a - z , 0 - 9 allowed
* has to begain with an alphabet
EMP2021
2021EMP   <- Error

* Special characters $, # , _allowed
* in MySQL if you want to use reserved character # in tablename and coloumnname , then enclose it in back quotes
* ` ` back quotes

eg. `EMP#`
*134 reserved words not allowed in tablename

# MySQL Documentation
http://docs.oracle.com

# Datatypes

Char
* allows any character
* could be alpha-numeric also
* max upto 225 characters
* e.g PANNO , ROLL_NO
* fixed length character string
* wastage of HD space
* searching and retrival is Fast

eg ADHAR_NO

Varchar
* variable character
* allows any character
* could be alpha - numeric also
* max upto 65,535 characters (64 Kb -1)
* no default width (width has to be specified)
* variable length character string
* conserve on HD space
* searching and retrival is slow

eg. ADDRESS

Text
* stored outside the table
* stored away from the table row
* MySQL maintains a LOCATOR (HD address from the table row to the text column data)
* used only for those coloumns that will are to be used in searching
* Benefit is that the processing speed and performance in the table is not affected
* e.g REMARKS , COMMENTS , EXPERIENCE , RESUME , FEEDBACK , PRODUCT_DTLS etc.

width does not have to be specified for this datatype

Tinytext
* max 225 characters

Text
* max 65, 535 characters

Mediumtext
* max 16 , 777 , 215 characters (16 Mb - 1)

Longtext
* max 4 ,294 .967 ,295 characters (4 Gb - 1)

Binary
* fixed length binary string (e.g '10101010000011')

* max 255 Bytes of binary data
* width need not be specified
* e.g. small images ,barcodes ,picture codes ,QR codes , thumbnails, signatures , fingerprints etc.

Varbinary
* variable length binary string
* max 65,535 Bytes of binary data
* no default width (width has to be specified)
*e.g small images , thumbnails ,icons etc.
* both of the above (Binary and Varbinary) are stored as character strings of 1's and 0's

Blob ->
* Binary Large Object
* stored outside the table
* stored away from the table row
* MySQL maintains a LOCATOR from the table row to the Blob data
* width does not have to be specified.

Tinyblob
* max upto 255 Bytes of Binary Data

Blob
*max upto 65,535 Bytes of binary data

Mediumblob
* upto 16,777,215 Bytes of binary data (16 Mb - 1)

Longblob
* upto 4,294,967,295 Bytes of binary data

e.g photoshop

# Integer Types (Exact value) :
* Signed or Unsigned
* by default it is Signed

Tinyint (-128 to 127) or (0 to 255)
* 1 Byte
* e.g age int unsigned

Smallint
* 2 Bytes

Mediumint
* 3 Bytes

Int
* 4 Bytes

Bigint
* 8 Bytes

# Floating -Point Types (Approximate value) :

Float
* single precision
* upto 7 decimals

Double
* upto 15 decimals

# Fixed-Points Types (Exact value) :

Decimal
* stores double as a string
e.g "653.7"
* max number of digits is 65
* used when it is important to preserve exact precisions.

Boolean
* logical datatype
* True and False evaluate to 1 and 0 respectively
* e.g maraital _status boolean

# Date and Time Datatypes

Date
*1st Jn 1000 AD to 31st Dec 9999AD
* 'YYYY-MM-DD' is the default date format in MySQL
*e.g
'2021-09-24'
'21-09-24'
* year values in the range 70-99 are converted to 1970-1999
* year values in the range 00-69 are converted to 2000-2069
(1970 is the cut-off year)
* date1 - date2 -> returns the number of days between the 2 dates.
* how MySQL reads dates
1st Jan 1000AD -> 1
2nd Jan 1000AD -> 2
.
24th Sept 2021 AD -> 2147593 (number of days sence 1st Jan 1000AD)

* internally date is stored as a fixed length number.
7 Bytes of storage

# Time
* 'hh:mm:ss' or 'HHH:MM:SS'
*   time values may range from '-838;59;59'to'838:59:59'

Datetime

* date and time is stored together

* 'YYYY-MM-DD hh:mm:ss'
* '1000-01-01 00:00:00' to '9999:12:31 23:59:59'
* datetime1 - datetime -> returns number of days between the two and the remainder hours , minutes and seconds

Year
* YYYY
*1901 to 2155

# In MySQL
* max 4,096 columns per table provided the row size <= 65,535 Bytes
* no limit on number for rows per table provided the table size<+64 Terabytes

#CREATE TABLE

```
create table Emp
(
Empno char(4),
Ename varchar(25),
Sal float,
City varchar(15),
Dob date
);
```

* ; is known as delimiter. it indicates end of command
* SQL commands are case-insensitive.

#INSERT

```
insert into emp
values('1','Amit','5000','Mumbai','1995-10-15');
```

* data is case-sensitive
* for char , varchar , date , time , datetime , use ' '
* '1995-10-15' -> 'YY-MM-DD' is default date format in MySQL.
* '15-OCT-95' -> 'DD-MON-YY' is default date format in Oracle.

```
insert into emp (empno , sal , ename , city , dob )
values('2' , 'Kaushal ', 'Delhi ', '1990-11-17');    <- Recommended
```
1. Readable
2. Flexible (can specify column values in any order)
3. In future if you alter the table ,if you add a colomn , the INSERT statement will continue to work

```
insert into emp (empno , sal )
values('3' , 7000 );
```

# Null value

* null means nothing
* null value is having ASCII value 0

* special treatmenbt given to null value in all RDBMS
* null value is independent of datatype
* null value occupies only 1 Byte of storage
* if the row is ending with null values, then all those columns will no occupy any space.
* those columns that are likely to have a large number of null values ,its recommanded that they should be specified at the end of the table structure , to conserve on HD space.

```
insert into emp
values('3' ,'shantiram' );
insert into emp
values('3' ,'shantiram',null,null,null );
insert into emp
values('3' ,null, 'shantiram' , null);
```

* To insert multiple rows in a table simultaneously :

```
insert into emp values
('1' , 'A' ,5000,'Mumbai','1990-10-01'),
('2' , 'B' ,5000,'DELHI','1985-08-03'),
('3' , 'C' ,5000,'Mumbai','1994-04-07'),
('4' , 'D' ,5000,'Mumbai','1989-05-09') ;
```

```
insert into emp(empno,sal) values
('1',5000) ,
('2',6000) ,
('3',4000) ,
('4',7000) ;
```

* above 2 commands will works only in MySQL.
* above 2 commands are not supported by Oracle.

# SELECT

```
select * from emp;
```

when you install MySQL, 2 users are automatically created
root
* has DBA privileges
* create users ,assign privileges , take backups , performance monitering , performance tuning ,etc

mysql.sys
* most important user in MySQL
* owner of database
* startup database , shutdown database , perform recovery

To connect to MySQL database using MySQL command Line Client :
* open MySQL Command Line Client
* Specify the password for 'root' user

To connect to MySQL database using MySQL Workbench :
* open MySQL workbench

* MySQL Connections (click on + sign to create a new connection)
* Connection Name : Connection for root user
* Connection method : standard TCP/IP
* Host Name: localhost
* Port no :3306
(for Oracle - > 1521)
* Username: root
* Password: (store in Vault.... Push button) -> click on the button the password
* Click on Ok
* SCHEMA is a synonym for database
* Default Schema : (leave it blank)
* Test Connection (push button)
* Click on Ok
* Click in Ok

To log in to the MySQL database :
* Click on the connection you created.
* you will see Object Nevigator on LHS
* you will see query window at the top
* you will also see output window below
* you will see a Pull down menu at the top and a Horizontall toolbar at the top

Some basic commands post logon :

```
show databases;
```
Ctrl+enter -> to execute

to connect to database ;
```
use <databasename>;
use mysql;
```

To view which all users are created :

```
select*from user;
```
* USER -> is a MySQL created system table
* It is automatically created when you install MySQL.

To create a new user and a Default database forr that user :

```
create database<databasename>;
or
create schema<schemaname>;
create database metiitnashikpgdacsept2021;  <- this command creates a database/schema

show databases;
create user <username> identified by the <password>;

create user pgdac1@'%' identified by 'welcome';

use mysql;
select * from user;
```

To grant the permissions :
Click on server (menu at the top) -> Users and Privileges -> Click on it.
Select the username you created from the user account list on LHS
Go to Administrative Roles (tab)

DBA Role (checkbox) -> Click on it
Click on Apply (push button)

Go to the Schema privilages (tab)

Add Entry ... (push button) -> Click on it

Select Schema (Radio button) -> Click on it
click on the Poplist and select cdacmetiitnshiksept2021

ok(push button) -> Click on it

Select "ALL" (push button) -> CLick on it

Grant option(checkbox) -> click on it
APPLY (push button) -> click on it

File -> Exit
Open MySQL workbench
Create a new Connection for user pgdac1
Default schema : metiitnashikpgdacsept2021

to see which all tables you have created
show tables;

to see the structure of tables:
desc <tablename>;
desc emp;

for putty -> grant dba to pgdac1;

27/09/2021

192.168.4.31

SELEC
----

select * from emp; -> show all rows and columns of emp table

'*' means metacharacter (all columns and rows)

To restrict columns:-

select empno, ename from emp; <- shows all row of empno and ename

select statment shows the output according to user requirement first empno, second is ename.

The position of column in SELECT statment, will determine the position of columns in output
(you will write as per user requirements)

To restrict rows:-
WHERE clause:-

select * from emp where deptno = 10; <- shows the all columns where depno is 10

* WHERE clause is used for searching
* searching takes place in DB server HD
* WHERE clause is used to restrict the rows
* WHERE clause is used to retrive the rows from DB server HD to server RAM

select * from emp where sal>2000;
select ename, sal from emp where sal>2000;

Relational Operator:-
1. >
2. >=
3. <
4. <=
5. != or <>
6. =

precidence is as above

Logical Operator:-

1. NOT
2. AND select * from emp where sal > 2000 and sal < 3000; shows the sal between 2000 and 3000.
3. OR   select * from emp where sal > 2000 or sal < 3000;   Wrong logic it will show all rows.

To change the precidence use brakets "()"

select * from emp where (deptno =10 or sal >2000) and sal <3000;

select * from emp where job = 'MANAGER';

select * from emp where job = 'manager';

* in MySQL, queries are case-insensitive
(More user-friendly, less secure)

* in Oracle, queries are case-sensitve

(Less user-friendly, more secure)

select * from emp where job = 'MANAGER' or job = 'CLERK';   Mnanger or Clerk

select * from emp where job = 'MANAGER' and job = 'CLERK';   Manager and Clek

----
28/09/2121

select deptno, job ename, sal, hiredate from emp;

* in a DBMS, data is stored in a file
* in a file, rows are stored sequentially
* concept of Row numbering is abailale in DBMS
* hence DBMS, it is possible to see the first 'N' rows in a file or the
last 'N' rows in a file

* table is not a file, every row is a file
* rows of a table are scattered (fragmented) all over the DB server HD
* when you INSERT into a table, wherever it finds the free space in the DB server HD, it will store the row there
* the reason why RDBMS does this, is to speed up the INSERT statement
especially considering a multi-user environment
* when you SELECT from a table, the searching will always be sequential
* when you SELECT from table, the order of rows in the output will
always be in ascending order of row address
* hence in a RDBMS, it is not possible to see the first 'N' rows in a
table of the last 'N' rows in a table

* if you UPDATE a row, if the row length is increasing, then the row address
May change (it's only in the case of VARCHAR that the row lenght may increase
or decrease)

ORDER BY clause:-

* Used for sorting (to make it more presentable)
(to view the output in a specific order)

select deptno, job, ename, sal, hiredat from emp order by ename;

select deptno, job, ename, sal, hiredat from emp order by ename desc;
asc -> ascending (by default)
desc -> descending

select deptno, job, ename, sal, hiredat from emp order by deptno, job;

select deptno, job, ename, sal, hiredat from emp order by deptno desc, job;

select deptno, job, ename, sal, hiredat from emp order by deptno desc, job desc;

* no upper limit on the number of columns in ORDER BY clause

* sorting is one operation that always slows down your SELECT statement
* if you have a large number of columns in ORDER BY clause, it will be slow, because that much sorting has to take place.

select deptno, job, ename, sal, hiredat from emp where deptno=10 order by ename;

* WHERE clause has to be specified before the ORDER BY clause
* SELECT statement executes from top to bottom, and left to right

select ename, sal*12 from emp;
select ename, sal*12 from emp order by sal*12;

* ORDER by clause is the last clasue in select statment
select ename, sal*12 annual from emp order by annual;
select ename, sal*12 "annual" from emp order by "annual";
select ename, sal*12 "Annual" from emp order by "Annual";

select ename, sal*12 "Annual" from emp order by 2;

select * from emp order by 2;

SELECT * from emp WHERE ename > 'A' and ename < 'B';

SELECT * from emp WHERE ename >= 'A' and ename <= 'B';

* Blank-padded comparision semantics:-
Whenever you compare 2 strings of different lengths, the shorter of the 2 strins is temporatily padded with blank spaces on RHS, such that their lengths become equal; then it will start the comparison, charcter by character, based on ASCII value

select * from emp where ename like 'A%';

like - special operator
Wildcards (used for pattern matching)
% any characer and any number of character

In Oracle, to make the query case-insensitive:-

select * from emp where ename like 'A%' or ename like 'a%';

select * from emp where ename like 'A%'; 'Starting with 'A'

select * from emp where ename like '%A'; Ending with 'A' select * from emp where ename like '%A%'; start, end, or any containing A

select * from emp where ename like 'A%A'; starting with 'A' and Ending with 'A'

select * from emp where ename like '_ _A%'; thired letter is A

'_' Wild card

select * from emp where ename not like 'A%'

29/09/2021

LIKE

select * from emp
where sal >=2000 and sal<=3000;

* Between is a special operator

select * from emp
where sal between 2000 and 3000;

* Between -> inclusive
-faster if we have large number of rows
-whenever you are searching for data that lies within a range, use the BETWEEN operator:-
a. readymade method by the name of BETWEEN is already present in the database in the COMPILED FORMAT, the EXECUTION PLAN etc. is ready, and it directly executes
b. easier to write (closer to everyday spoken English)

* NOT BETWEEN

select * from emp
where sal not between 2000 and 3000;

* NOT Between -> exclusive

select * from emp
wheree hiredate>= '2020-01-01' and hiredate <= '2021-12-31;

select * from emp
wheree hiredate between '2020-01-01' and '2021-12-31;

select * from emp

where ename >= 'A' and ename <= 'F';   include F but not F____

select * from emp
where ename between 'A' and 'F';

* ANY
perform as logical OR

select * from emp
where deptno = 10 or deptno = 20 or deptno =30;

select * from emp
where deptno = ANY(10,20,30); (FAST)

* IN
perform as logical OR

select * from emp
where deptno = IN(10,20,30);   (FASTEST)

in
not in
=,>,<,!= any

* IN operator is faster than ANY operator
* ANY operator is more powerful than IN operator
* With IN operator, you can check for IN and NOT IN
* with ANY operator, you can check for =ANY, !=ANY, >ANY, >=ANY, <ANY, <=ANY.
* if you want to check for equality or inequality, then use the IN
operator; if you want to check for >,>=,<, or <= then use the ANY operator

select * from emp
where city in ('Mumbai', 'Delhi');

* ANY operator works in MySQL only provided if it use with sub-query

DDL -> create
DML -> insert, update
DQL -> select

UPDAT
-----

update emp
set sal = 10000
where empno = 1;

```
update emp
set sal = sal + sal*0.4
where empno = 1;

update emp
set sal = 10000, city = 'Nashik'
where empno = 1;

update emp
set sal = 10000
where city = 'Mumbai';

update emp
set sal = 10000, city = 'Nashik'
where city = 'Mumbai';
```

30/09/1997

DDL-> create, drop
DML-> insert,update, delete
DQL-> select

```
delete from emp
where city = 'Mumbai'

delete from emp;
```

Drop

```
Drop table emp;
```

* You cannot have a where clause with drop table
* if you want to drop multiple tables, then you will have to drop
each table respectively.

* UPDATE and DELETE commands without WHERE clause will not be allowed
in MySQL Workbech

To Try out the above commands:-

In MySQL Workbech:-

Click on Edit(menu at the top) -> Preferences -> SQL Editor -> "Safe Updates"
(checkbox at the bottom) -> Uncheck it -> Click on OK

This requires a reconnection to the databases server
Click on Query (menu at the top) -> reconnect server

MySQL - SQL - TRANSACTION PROCESSING

* Commit will save all the DML changes since the last committed state
* When the user issues the commit, it is known as End of Transacion
Commit will make the Transaction permanent
* Transaction is a sub-unit of Work
* Total Work done = T1 + T2 + T3 + ... + Tn;
* User/Client will decide when to issue the Commit
* When to issue the Commit will depend upon the logical scope of work.
* Rollback will undo all the DML changes since the last commiteds state

```
commit work;
```

* WORK -> ANSI SQL
* WORK -> optional in MySQL RDBMS and Oracle RDBMS

```
commit;
```

```
rollback work;
```
* Work -> AnSI SQL
* WORK -> optional in MySQL RDBMS and Oracle RDBMS

* Only the DML commands are affected by Rollback and Commit
* Any DDL command, it automatically commits (not ony will it commit itself, but it will commit everything above it)

* When you exit from SQL*Plus(Oracle), it qutomatically Commits
* Any kind of power failure, networ failure, system failure, PC reboot, window close, improper exit, etc. ; in all such cases your last uncommited transaction is automatically Rolled bask (in MySQL and Oracle)

* Transaction is a unit of Work
* Savepoint is a point within your Work
* You can Rollback to a Savepoint
* You cannot Commit to a savepoint
* Commit will save all the DML changes since the last Commited state
* When you Rollback or Commit, the intermediate savepoints are automatically cleared (they no longer exist); if you want to use those savepoints again, then you will have to reissue them in a new transaction.

* Savepoint is the sub-unit of Work
* You can only Rollback sequentially
* for savepoint name

```
rollback to pqr;
rollback to pqr;
```

* within a Transaction, you can have 2 Savepoints with the same name; the latest savepoint overwrites the older one; the older savepont no longer exists

* To try out Rollback, Commit and Savepoint in MySQL Workbench:-
Click on Query (menu at the top) -> Auto-Commit Transaction-> Uncheck it

----
01/10/2021

READ and WRITE Consistency

* In a multi-user environment , when you select from a table, you can view ONLY the commited data of all users plus changes made by you

cl scr > clear screen

set sqlprompt 'user1>'

In Oracle:-
set autocommit on
set autocommit off

Global login
This is a startup file of SQL*Plus
This file is automatically exectued when you start SQL*plus on your computer

glogin.sql (ThisPC/WindowsC/app/db_home/sqlplus/admin/glogin)
open in notepad

set pagesize 20
set linesize 100
set autocommit on
etc.

ROW LOCKING
* When you UPDATE or DELETE a row, that row is automatically locked for other usrs
* ROW LOCKING IS AUTOMATIC IN MYSQL AND ORACLE
* when you UPDATE or DELETE a row, that row becomes READ_ONLY for other users
* other users can SELECT from that table; they will view the old data before changes
* other user can INSERT rows into that table
* other user can UPDATE or DELETE "other" rows from that table
* no other user can UPDATE or DELETE your LOCKED ROW till you have issue a ROLLBACK or COMMIT
* LOCKES AUTOMATICALLY REALEASED WHEN YOU ROLLBACK OR COMMIT

OPTIMISTIC ROW LOCKING -> automactic row locking mechnism of MySQL and Oracle

PESSIMISTIC ROW LOCKING-> manually lock the rows in advanced BEFORE issuing UPDATE or DELETE

To lock the rows manually, you will have to use SELECT statement with a FOR UPDATE clause

e.g.
select * from emp for update; -> entire table is locked for update or delete manually

```
select * from emp
where deptno = 10
for update;
```

* LOCKS ARE AUTOMATICALLY REALEASED WHEN YOU ROLLBACK OR COMMIT

```
select * from emp
where deptno = 10
for update wait;            DEFAULT WAIT

select * from emp
where deptno = 10
for update nowait;          NOWAIT

select * from emp
where deptno = 10
for update wait 60;         Wait for 60 seconds
```

TO try out row locking in MySQL:-
Open MyQL Workbench
Click on Query (menu at the top) -> New tab to current server -> Click on in

Now you have 2 query windows to try out row locking

In MySQL to abort the operation (to abort the Request Queue) :-
Click on Query (menu at the top) -> Click on Stop

Manual row locking in MySQL:-
WAIT/NOWAIT options not available in MySQL

02/10/2021

MySQL - SQL -Functions

Function -> Routine that retruns a value

Character Functions

applicable on Char and Varchar datatypes

EMP

FNAME	LNAME
Arun	Puran
Tarun	Arun
Sirun	Kirun
Nutan	Puran

fname varchar(15)
lname varchar(15)

select fname, lname from emp;

concat(str1, str2)

select concat(fname, lname) from emp;

ArunPurun
TarunArun
SirunKirun

select concat(concat(fname,' '), lname) from emp;

* max upto 255 levels for function within function > common for all RDBMS

select upper(fname) from emp;

Arun
Tarun
Sirun

for display purpose

update emp set fname = upper(fname);

we can use the functions with update delete also

select * from emp where fname = 'ARUN';   <- works in MySQL, doesnot work in Oracle

in Oracle data is case sensitive

Solution for Case-insensitive query in Oracle RDBMS:-
select * from emp where upper(fname) = 'ARUN';

select * from emp where lower(fname) = 'arun';

select initcap(fname) from emp;
Arun
Tarun
Sirun

EMP
ENAME
Arun Purun
Tarun Arun
Sirun Kirun   only one column

by default character data is left justified and numeric data is right jusified

select lpad (ename,25,' ') from emp;

select lpad (ename,25,'*') from emp;   -> padded with * on right side

Uses:-
a. Right justification
b. Cheque printing

select rpad(ename,25,' ') from emp;

select rpad(ename,25,'*') from emp;   padded with * on left side

Uses:-
a. Left justification of numeric data
b. Cheque printing
c. convert varchar to char (convert variable lenght to fixed lenght)

for center justification use both rpad and lpad (function within function)

select ltrim(ename) from emp; -> Removes the blanck spaces from left side (trim)
uses:-
a. left justification

select rtrim(ename) from emp; -> Removes the blanck spaces from right side (trim)
Use:-
a. To convert char to varchar (convert fixed lenght to variable length)
b. Right justification of char data (lpad(rtrim(ename ...)....))

select trim(ename) from emp;   -> removes blank spaces from both the sides in MySQL

select substr(ename,3) from emp;

-> starting from the 3rd postion

un Parun
run Arun
run Kirun

select substr(ename,3,2) from emp;

-> starting from 3rd letter; 2 characters
means 3rd and 4th character

un
ru
ru
ta

Uses:-
To extract a part of the stirn

select substr(ename,-3) from emp; -> start at -3

last 3 letters

run
run
run

select substr(ename,-3,2) from emp;

ru
ru
ru

substr('Nashik Road',1,6)
Nashik

substr('Nashik Road',8)
Road

select replace(ename,'un','xy') from emp;
un->xy

Arxy Purxy
Tarxy Arxy

use for nickname
Indian IND

1->One
2->Two
3->Three

vise versa

a->n
b->k
c->o

USED FOR ENCRYPTION OR DECRYPTION

TRANSLATE Function Works in Oracle, not supported in MySQL:-

select translate(ename, 'un','xy') from emp;

u->x
n->y

Arxy Pxrxy
etc.

select instr(ename,'un') from emp;
3
4
4
10
-> returns starting positon of string
if string is not found then it returns 0

Uses:-
a. check if one string exists in another one

EMP
ENAME
Arun
Bannerjee
Charlie

select lenght(ename) from emp;
-> returns lenght of characters in emp
4
9
7

select ascii(ename) from emmp;
-> reurns the ascii value of first char
65
66
67

select ascii(substr(ename,2)) from emmp;
-> reurns the ascii value of 2nd char

select ascii('z') from emp;   ruturns ascii value of 'z' 3(rows) times

122
122
122

select distinct ascii('z') from emp;
122

```
use mysql;
select ascii('z') from dual;
122
```

* DUAL is a system table
* DUAL is a dummy table
* it contains only 1 row and 1 column

```
select sutstr('Nashik Road',1,6) from dual;
select 'Welcome to MET' "MESSAGE" from dual;
select 3*12 from dual;


select char(65 using utf8) from dual;
```

->returns char value from ascii value 65>>A

A

->where utf8 is the given character set for U.S. English else default is binary character set

```
select * from emp where ename = 'Aroon';

slect * from emp where soudex(ename) = soudex('Aroon');
```

04/09/221

* NUMBER FUNCTIONS

Q3 Answer

```
EMP
SAL
1234.567
1849.019
1375.618
1751.51


select round(sal) from emp;
1235
1849
1376
1751

select round(sal,1) from emp;
```

```
1234.6
1849
1375.7
1751.2
```

```
select round(sal,2) from emp;
1234.57
1849.02
1375.62
1751.15
```

```
select round(sal,-2) from emp;
1200
1800
1400
1800
```

```
select round(sal,-3) from emp;
1000
2000
1000
2000
```

```
select truncate(sal,0) from emp;-> Removes decimal
1234
1849
1375
1751
```

Uses:-
Date and Time calculation

```
select truncate(sal,1) from emp;
1234.5
1849
1375.6
1751.1
```

```
select truncate(sal,2) from emp;
1234.56
1849.01
1375.61
1751.15
```

```
select truncate(sal,-2) from emp;
1200
1800
1300
1700
```

CEIL -> ceiling (if there is any value at all in decimal then it will add one to the whole number)

```
select ceil(sal) from emp;
1235
1850
1376
1752
```

Uses:-
a. Bill payments, Interest payments, EMI payments, etc.

* FLOOR :-

```
select floor(sal) from emp;
1234
1849
1375
1751
```

```
select truncate(3.6,0), floor(3.6), truncate(-3.6,0), floor(-3.6) from dual;
3                  3                 -3                 -4
```

```
select truncate(3.2,0), floor(3.2), truncate(-3.2,0), floor(-3.2) from dual;
3                  3                 -3                 -4
```

* SIGN:-

if number is -ve returns -1
if number is +ve returns 1
if number is 0 returns 0

```
select sign(-15) from dual;
```

Uses:-
a. To check if num is +ve or -ve
b. sign (bloodgrop)
c. sign (coronatest)
d. sign (reports)
e. sign (traffic_signal)
f. sign (feedback)
g. sign (bank_balacne)

To find out the greater of 2 numbers
x->10
y->20
sign(x-y)

To find Profit and Loss
sign(SP-CP)

*MOD:-
returns reminder

```
select mod(9,5) from dual;
4
```

```
select mod(8.22,2.2) from dual;
1.62
```

*SQRT
only works for +ve number
-ve number gets an error

```
select sqrt(81) from dual;
9
```

-ve
concat(sqrt(sign(-10)*(-10)),i)

*POWER
for cube
select power(10,3) from dual;

10power3 -> 1000

for cuberoot
select power(10,1/3) from dual;

*ABS
always return +ve number

```
select abs(-10) from dual;          -> absolute value
10
```

* Trignometric Functions

sin(x)
cos(x)
tan(x)

x should be in radians

convert data into radians before applying trignometric functions

* LOG

ln(y)
log(n,m)   -> n=base m=value   log n(m)

* DATE FUNCTIONS

date function is different in mysql and oracle

In MySQL

EMP
HIREDATE
2019-10-15
2019-12-31
2020-01-15

1. Date, Time, Datetime, Year
2. 1st Jan 1000 AD to 31st Dec 9999 AD
3. 'YYYY-MM-DD' is the default date format in MySQL
4. 'YY-MM-DD'
5. date1-date2
6. internally date is stored as a fixed-length number
7. internally date is stored as number of days since 1st Jan 1000 AD
8. 7 Bytes of Storage
9. Datetime (time is stored as a fraction of days e.g. 1.5 -> 1 day 12 hours)

3.5123
1 day = 24 hours = 24*60 mins = 24*60*60 secs = 24*3600

3-> days
.5123 -> .5123*24*3600 seconds

* sysdate()
- Is a funciton, it returns the current date and time
- returns the server date and time

select sysdate() from dual;

* adddate()

select adddate(sysdate(),1) from dual;

add 1 day and time (adds 1 day ==24 hours)

select adddate(sysdate(),-1) from dual;
yersterday date

* datediff()

```
select datediff(sysdate(), hiredate) from emp;
```

- date difference
- returns number of days

```
715
692
668
```

Oracle Date -> 1st Jan 4712 BC to 31st Dec 9999 AD

* date_add
Adding months

```
selcet date_add(hiredate,interval 2 month) from emp;

selcet date_add(hiredate,interval -2 month) from emp;


selcet date_add(hiredate,interval 1 year) from emp;
```

for adding days use adddate
for adding months use date_add
for adding year use date_add

*last_day
- returns last date of months
- present only in Mysql and oracle

```
select last_day(hiredate) from emp;

2019-10-31
2019-12-31
```

*dayname()
returns day of week

```
select dayname(sysdate()) from dual;
Monday

select upper(dayname(sysdate())) from dual;
MONDAY

select substr(dayname(sysdate()),1,3) from dual;
Mon
```

* addtime

```
select addtime('2021-01-10 11:30:00', '1') from dual;
'2021-01-10 11:30:01'
```

add the second

```
select addtime('2021-01-10 11:30:00', '1:30:45') from dual;
H:M:S

'2021-01-10 13:00:45'
```

*LILST FUNCTIONS (ifnull, greatest,least)

- independent of datatype
- any comparision done with null, returns null
- Any operation done with null, returns null

```
select * from emp where comm = null;
```

Pessimistic querying -> Searching for null value

```
select * from emp where comm is null;
```

is null-> Sepcial Operator

```
select * from emp where comm != null;
select * from emp where comm is not null;
```

```
select sal+comm form emp;
```

*ifnull()

```
select sal + ifnull(comm,0) from emp;
```

```
select ifnull(sal,0) + ifnull(comm,0) from emp;
```

```
ifnull(comm,0)
ifnull(comm,100)
ifnull(city,'Mumbai')
ifnull(orderdate,'2021-04-01')
```

*greatest()

```
select greatest(sal,3000) from emp;
```

Uses:-
a. used to set lower limit on some value

```
greatest(val1,val2,.....val255)
```

```
greatest(num1,num2,num3)
```

greatest(str1,str2,str3)

greatest(date1,date2,date3)

Bonus= 10% sal, Min Bonus = 300

select sal, greatest(sal*0.1,300) bonus from emp;

* Least()
- used to set a upper limit on some value

select least(sal,3000) from emp;

Cashback = 10% amt, Max Cashback = 3000

select amt, least(amt*0.1,3000) cashback from orders;
/\
||

EMP

SAL	DEPTNO
1000	10
2000	10
3000	20
4000	30
5000	40

||
\/

* CASE expression clause

```
select
case
when deptno = 10 then 'Training'
when deptno = 20 then 'Exports"
when deptno = 30 then 'Sales'
else 'Others'
end "DEPTNAME"
from emp;
```

```
Training
Training
Exports
Sales
Others
```

- If you don't specify ESLE it returns Null value
- This is most powerful function in my whole MySQL

if deptno = 10 then HRA = 40% annual
if deptno = 20 then HRA = 25% annual
else HRA = 15% annual

```
select deptno, ename, sal, sal*12 annual,
case
when deptno = 10 then sal*12*0.4
when deptno = 20 then sal*12*0.25
esle sal*12*.15
end as HRA
from emp;
```

if sal>3000 then REMARK ='High Income'
if sal=3000 then REMARK ='Middle Income'
if sal<3000 then REMARK ='Low Income'

```
select ename,sal,
case
when sign(sign-3000) = 1 then 'High Income'
when sign(sign-3000) = -1 then 'Low Income'
else 'Middle Income'
end "REMARKS"
from emp
order by 2;
```

* Environment Functions

```
select user() from dual;
```
-> returns mysql username

$whoami -> linux command

ORDERS

```
ONUM  ODATE          CNUM   AMT    COL1    COL2
5001  '2021-08-10'   1001   300    PGDAC1          '2021-10-04'
```

```
insert into orders
values(5001,'2021-08-10',1001,300,user(),sysdate());
```

used to maintain a log (audit trails) of DML operations

05/10/2021

06/10/2021

* SQL JOINS
- To view/ combine the columns of 2 or more tables

```
select ename, deptno from emp;
```

```
ENAME DEPTNO
-----  ------
Arin   1
Ali    2
Kirun  1
Jack   2
```

dept-> driving table
emp -> driven table
<<-----

```
select dname, ename from emp, dept
where dept.deptno = emp.deptno;
```

tablename.columnname

```
DNAME ENAME
-----  -----
TRN    Arun
TRN    Ali
TRN    Kirun
EXP    Jack
EXP    Thomas
```

emp-> driving table
dept-> driven table

```
select dname, ename from dept, emp
where dept.deptno = emp.deptno;
```

output is same

Slow (Processing time is more)

DRIVING TABLE SHOULD BE MINIMUM ROWS

IN ORDER FOR THE JOIN TO WORK FASTER,
PREFERABLLY THE DRIVING TABLE SHOULD

* the common column in both the tables(i.e. deptno), the common column naem need not be the same in both the tables, because the same column may have a different meaning elsewhere in some other table

```
select * from emp, dept
where dept.deptno = emp.deptno
oerder by 1;

select deptno, dname, loc, ename,job, sal from emp,dept
where dept.deptno = emp.deptno
order by 1;

select dept.deptno, dname, loc, ename,job, sal from emp,dept
where dept.deptno = emp.deptno
order by 1;

select emp.deptno, dname, loc, ename,job, sal from emp,dept
where dept.deptno = emp.deptno
order by 1;


select deptno, sum(sal) from emp
group by deptno;

DEPTNO          SUM(SAL)
------      --------
1           1800
2           1700

select dname,sum(sal) from emp, dept
where dept.deptno = emp.deptno
gorup by dname;

DEPTNO          SUM(SAL)
------      --------
TRN         1800
EXP         1700

select upper(dname),sum(sal) from emp, dept
where dept.deptno = emp.deptno
gorup by upper(dname);

DEPTNO          SUM(SAL)
------      --------
TRN         1800
EXP         1700


select upper(dname),sum(sal) from emp, dept
```

```
where dept.deptno = emp.deptno
gorup by upper(dname)
having .........
order by........;
```

TYPES OF JOINS:-

1. Equijoin (also known as Natural join)
- Join based on equality condiations
- Shows the matching rows of both the tables
- Uses:-
a. data is stored in multiple tables, you want to view the columns of 2 sor more tables, then you will require an Equijoin
- Most frequently used join(more than 90%)
hence it also known as Natural join

```
select dname, ename from emp, dept
where dept.deptno = emp.deptno;
```

```
DNAME	ENAME
-----	------
TRN	Arun
TRN	Ali
TRN	Kirun
EXP	Jack
EXP	Thomas
```

2. Inequijion (Non-Equijoin) :-

- join based on inequality condition
- Shows Non-Matching row of both the tables
- Uses:-
a. Used in Exeption Reports
- Very rareley used

```
select dname, ename from emp, dept
where dept.deptno != emp.deptno;
```

```
DNAME	ENAME
-----	------
TRN	Jack
TRN	Thomas
EXP	Arun
EXP	Ali
EXP	Kirun
MKTG	Arun
MKTG	Ali
MKTG	Kirun
```

```
MKTG   Jack
MKTG   Thomas
```

3. Outerjoin:-

a. Half Outerjoin(+) sign on only 1 side (LHS or RHS))
i Right Outerjoin   ii. Left Outerjoin

b. Full Outerjoin

- join with (+) sign
- shows matching rows of both the tables plus non-matching rows of "Outer" table
- Outer table -> table which is on 'Outer' side of (+) sign
- Outer table -> table which is on 'Opposite' side of (+) sign

- Uses:
Parent-Child Report (Master-Detail Report)

```
select dname, ename from emp, dept
where dept.deptno = emp.deptno (+);
LHS        =        RHS                 <- Right Outerjoin
```

```
DNAME ENAME
-----  ------
TRN    Arun
TRN    Ali
TRN    Kirun
EXP    Jack
MKTG   Thomas
```

EMP table is Child table known as Detail Table
DEPT table is Parent table   known as Master Table

diagram 6.30 time

```
select dname, ename from emp, dept
where dept.deptno (+) = emp.deptno;    <- Left Outerjoin
```

```
DNAME ENAME
-----  ------
TRN    Arun
TRN    Ali
TRN    Kiran
EXP    Jack
EXP    Thomas
.      Scott
```

dia 6:40

b. Full Outerjoin:-

- union of Right Outerjoin and Left Outerjoin
- shows matching rows of both the tables plus
- non-matching rows of both the tables
- based on Nested Do-While loop

```
select dname, ename from emp, dept
where dept.deptno = emp.deptno (+)     <- Right Outerjoin
UNION                                  Union
select dname, ename from emp, dept
where dept.deptno (+) = emp.deptno;    <- Left Outerjoin
```

```
DNAME ENAME
-----  ------
TRN    Arun
TRN    Ali
TRN    Kiran
EXP    Jack
EXP    Thomas
MKTG   .
.      Scott
```

* (+) sign for outer join works only in Oracle RDBMS, not supported by any othe RDBMS

* ANSI sytax for Full outerjoin :-
- supported by all RDBMS except for MySQL

```
selecct dname, ename from emp full outer join dept
on (dept.deptno = emp.deptno);
```

*To achive full outerjoin in Mysql:-
- Take UNION of ANSI syntax of Right Outerjoin and ANSI syntax of Left outerjoin

----
* ANSI sytax for Right outerjoin :-
- supported by all RDBMS including MySQL and Oracle

```
selecct dname, ename from emp right outer join dept
on (dept.deptno = emp.deptno);
```

----
* ANSI sytax for Left outerjoin :-
- supported by all RDBMS including MySQL and Oracle

```
selecct dname, ename from emp left outer join dept
on (dept.deptno = emp.deptno);
```

4. Inner Join -> by default every join is Inner join; using the (+) sing in Oracle or Using the Keyworld Outerjoin is What makes it an Outerjoin

* DO NOT MENTION INNNER JOIN IN INTERVIEWS UNLESS EXPLICITELY ASKED BY INTERVIEWER

----

4. Catesian join:-

- also known as cross product of 2 rows
- join without a WHERE clause
- Every row of driving table is combined with each an every row of driven table
- returns all the possible combinations
- Uses:-
a.it is used for printing purpose
e.g in the University, in students table you have all the students name, in subject table you have all the subjects name

dept-> driving table
emp-> driven table

select dname, ename from emp, dept; <-Fast

```
DNAME ENAME
-----  -----
TRN    Arun
TRN    Ali
TRN    Kirun
TRN    Jack
TRN    Thomas
EXP    Arun
EXP    Ali
EXP    Kirun
EXP    Jack
EXP    Thomas
MKTG   Arun
```

select dname, ename from emp, dept; <-Fast (less I/O between server HD and server RAM)

select dname, ename from dept, emp; <-Slow(more I/O between server HD and server RAM)

* the lesser the I/O between server HD and server RAM, the faster it will execute
* the more the I/O between server HD and server RAM, the slower it will execute
----
08/10/2021

screenshot 8/10 4:20

```
(5)     (3)     (2)
select dname ,ename , dhead from emp, dept,depthead
```

```
where depthead.deptno = dept.deptno
and dept.deptno= emp.deptno;
```

1000 100 10

```
select cname, snamem, onum, odate, amt from orders,customers, salespeople;
```

Types of Relationship amongst tables:-

1:1	(Dept: Depthead) or (Depthead:Dept)
1:Many	(Dept: EMP) or (Depthead:EMP)
Many:1	(:) or (:)
Many:Many	

PROJECTS

ProjectNo	ClientName	Location	Description
P1	DeloitteMumbai		CapitalGainsSystem
P2	BNP Paribas	Goregaon	Macros Programming
P3	MorganStanley	Lower Parel	Asset Managment S/W
P4	ICIC Bank	Bandra	Pension Processing S/W
P5	AMFI	Lower Parel	Website Devlopment

Problem- Display the Ename who is receiving the min(sal):-

Ans-

```
select ename from emp            <- main query(parent)(outer)
where sal =
(Select min(sal) from emp);      <- sub-query (child)(inner)
```

Max upto 255 levels for sub-queries
