Day 1
27/03/19

DBMS ex:- Clipper, Advanced Revelation, Quattro Pro etc.

Computaire → french word (to Compute / Calculate)

Computer convert data into Information
processing — Input — Output

Input :- Raw facts
Output :- Meaningfull data, processed data)

Database :- It is a Collection of large amount of data.

DBMS :- Database Management System. (DB Vista)
eg:- MS EXCEL, dBase, foxbase, foxpro
It is a readymade software which helps to manage your data.

ANSI defination :- Collection of program that allows you to insert, update, delete & process.

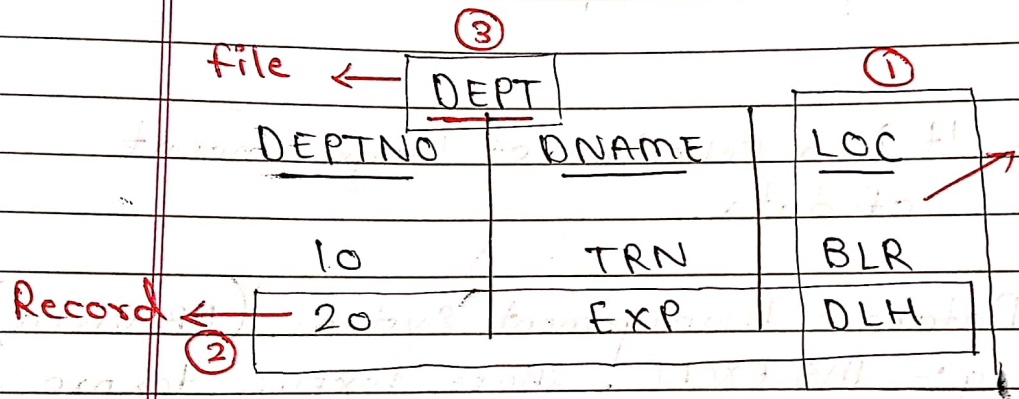
MS Excel program :- Known as Macro (VBA programming)

53% in industry

Oracle 11g

RDBMS → Relational DBMS

DBMS vs RDBMS
DBMS (eg :- Ms Excel, foxpro etc)
RDBMS (eg :- Oracle, MySQL etc)



① DBMS - field
RDBMS - Column, Attribute, Key, Method

② DBMS - Record
RDBMS - Row, tuple, Entity, Opportunity

③ DBMS - file
RDBMS - Table, relation, Entity class, matrix, applet

Dept

Deptno	Dname	Loc
10	TRN	BIR
20	EXP	DLH

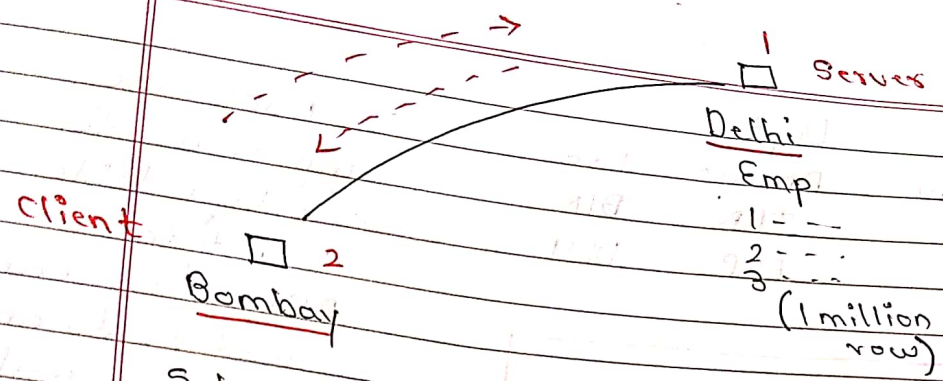
Emp

Empno	Ename	sal	Deptno
1	Bhavana	5000	10
2	Nikita	6000	20
3	Bhakti	7000	99 X

In DBMS the emp is acc to the Dept table but in RDBMS we can add or do any operation without refering to the dept

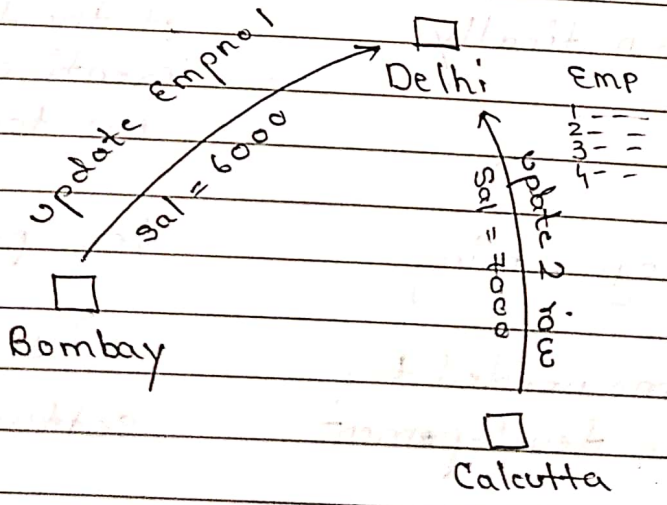
DBMS	RDBMS
• Relationship between 2 files is maintained programmatically	Relationship between 2 tables can be specified at the time of table creation. eg:- foreign key constraint
• More programming	less programming
• More time needed for software development	less time needed for software development
• High network traffic.	low network traffic.
Slow & expensive	faster & cheaper
Processing on client Machine (client-server X)	Processing on server machine. (client-Server Arch)

Network cost ↑ faster & cheaper ↓ Hardware & infra cost



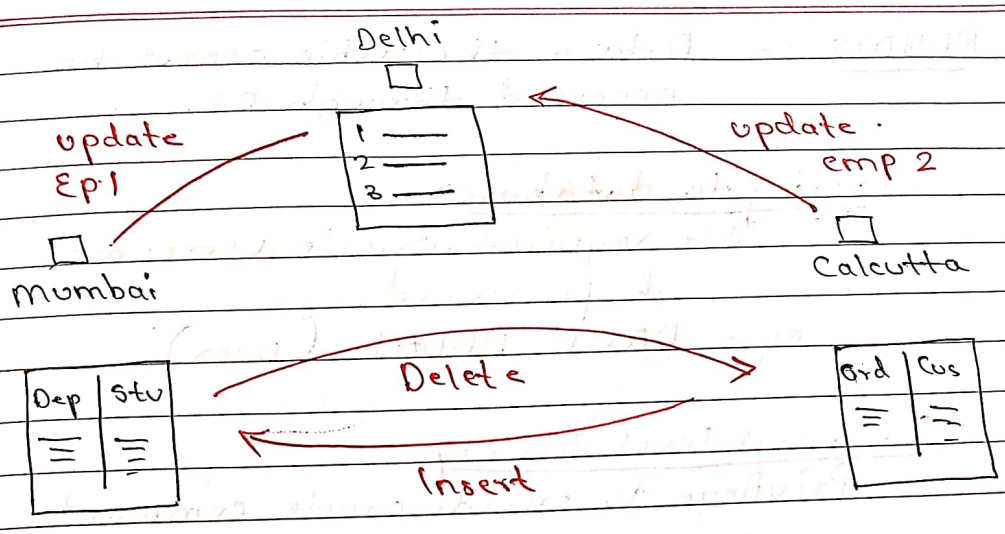
DBMS — Copy will send to 2 & process in 2 & takes the required data & again send to 1

RDBMS — Data is processed there itself in 1 & send to 2



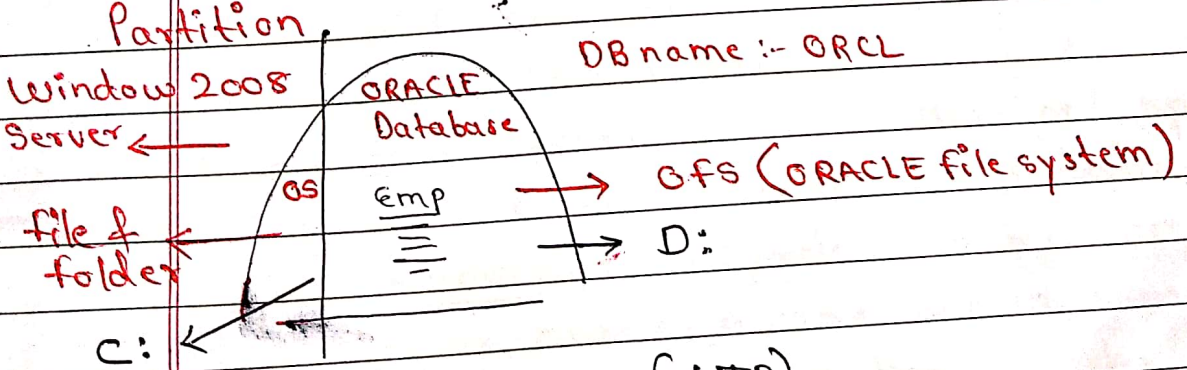
DBMS — It's file level locking
Not suitable for multiuser

RDBMS - It is row level locking
this suitable for multiuser
In RDBMS, Internally every row is a file.



DBMS :- Distributed databases are not supported.
No security of data

RDBMS :- Most of the RDBMS support Distributed Database
Multiple levels of Security
a) logging in security
b) Command level security
c) Object level security
Security is inbuilt feature of RDBMS



IP Add :- 192.168.1.201

RDBMS :- Data in the table cannot be accessed through os.

Logging to database

We required oracle username & password.

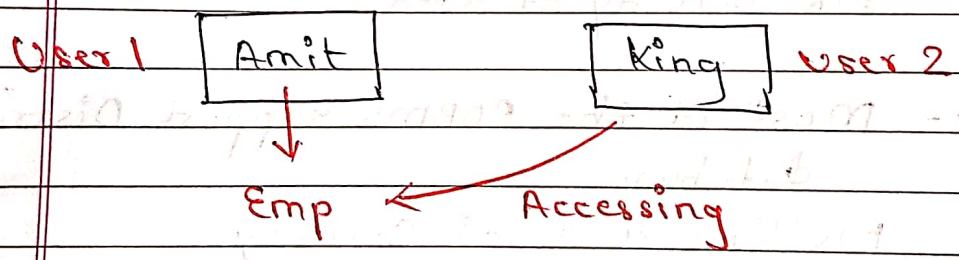
eg:- DAC1, DBDA1 (user)

Command level Security

Privilege to execute oracle commands

eg:- Create table, create trigger,

Object level Security



Accessing the tables of other users.

Personal Oracle :- Single user Software (free)

Various RDBMS available :-

Informix — fastest in term of processing speed
Oracle — Most popular — slow — programing easy
Sybase — Good RDBMS (works only with Window OS)
MS SQL Server
Ingres
Postgress } Character based RDBMA (text based) Works on MSDOS & Unix
Unify
DB2 — Good RDBMS from IBM
CICS
TELON
IDMS
MS Access
Paradox } PC based RDBMS (Single user RDBM
Vatcom SQL
MySQL

going down ↓

63% of world commercial database market in the client-server environ & 86% in the Internet environment

Database software company in the world.
→ Overall software company in the world.

- Oracle Corporation has a tie-up with DEC (Digital Electronics Coorporation)
  DEC has introduced a server model by the name Turbo Leisure.
  This server has been designed specifically for hosting the oracle database
  If you install Oracle database on this server then oracle works faster than Informix + therefore Oracle become the fastest RDBMA the world.
  Bombay Stock Exchange uses this server (3200 gb RAM)
  (Expensive Server)

Oracle RDBMS works on 113 OS.

- DB2 database server has to be a super computer (mainframe).
- 10/10 of the world largest database in the world uses ORACLE
- largest database in world (www.Amazon.com) (100's of Terabytes daily data) uses oracle
- largest Distributed Database in the world www.yahoo.com uses oracle.
- More than 90% of fortune 500 companies uses oracle.

MySQL

- launched in 1995 by Swedish company (C & C++ source code)
- It's name combination of 'my' the name of co-founder Michael Widenius daughter & 'SQL'
- MySQL is an open source RDBMS.
- Most widely used opens source client server model RDBMS.
- Part of the widely used LAMP open source application software stack (and other "AMP" stacks)

L	- linux	W	- Windows
A	- Apache	A	- Apache
M	- MySQL	M	- MySQL
P	- PHP, Perl/Python	P	- PHP, Perl/Python

- free software open source project that require a full featured database management system often use MySQL
- My SQL occupies 42% of world open source database market.
- SQL use by facebook, Twitter, flickr, Youtube, Instagram, Google (not for searches) Joomla, Wordpress, etc.
- Sun Mircosystems acquired MySQL in 2008
- Oracle acquired Sun Mircosystem in 2010.

Oracle 11g

Best Software development tools :-

SQL — Structured Query language, pronounce as "Sequel"
SQL * Plus
SQL Developer
PL/SQL
Oracle forms
Oracle Report
Oracle Menus
Oracle Graphics
IOS
EXP
IMP

Oraterm
SQL * loader
OEM
Oracle Case
Oracle financials
Oracle Manufacturing
Oracle HRMS
Oracle CRM
Oracle Applications
Personal Oracle

SQL :- • Structured Query language
• Commonly pronounce as "Sequel"

- Create table ; Alter table, Drop table
  Insert row, update row, Delete row
  Grant, Revoke, select
- Conforms to ANSI standard.
  (eg:- 1 character = 1 byte)
  conforms to ISO standard for QA.
- Initially founded by IBM. (1975 - 77)
  - Source code of Sql
    C, C++ → 90%
    Assembly → 10%
  - larry Ellison, Thomas Siebel, Scott Silbernick,
    Student of Dr Codd → Chris Nikitianos, Huma Saleri,
    Gavin Berry hired by IBM to write the source code of SQL.
  - RSBE (Relational Query by Example)
    old name of SQL

- IBM then gave RDBE free of Cost of ANSI → Govt
- ANSI renamed RDBE to SQL & now controlled by ANSI (hence SQL is common for all RDBMS)
- In 2005 the entire source code of SQL has been written in Java. (100% Java)

SQL * Plus :- Product of oracle Coorporation.
- Extension to SQL
- Used to remove the limitation of SQL (extra 54 commands)

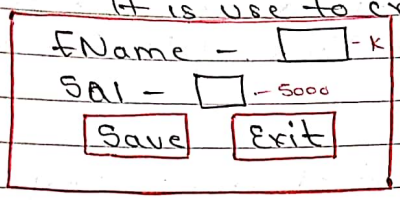
- Used for setting oracle environment.
  eg :- linesize, pagesize, sqlprompt, etc
- Character based reporting tools.
  (only text report)
- Use for business Intelligence (BI)
- SQL*plus is oracle client software.
- Interface with database for running SQL, and SQL * plus commands and PL/SQL programs.

SQL Developer
- Oracle client software.
- GUI based

PL/SQL
- Procedural language SQL
- Programming language from Oracle
- Used for database programming
  eg:- HTA_Cale, TAX_Calculation, ATTENDANCE_Calculation
- #1 database programming language in the world.

- overall programming language in the world.

Oracle forms
It is use to create data entry screen.



Oracle reports :- Reportwriter.
GUI based.
for business Intelligence (BI)

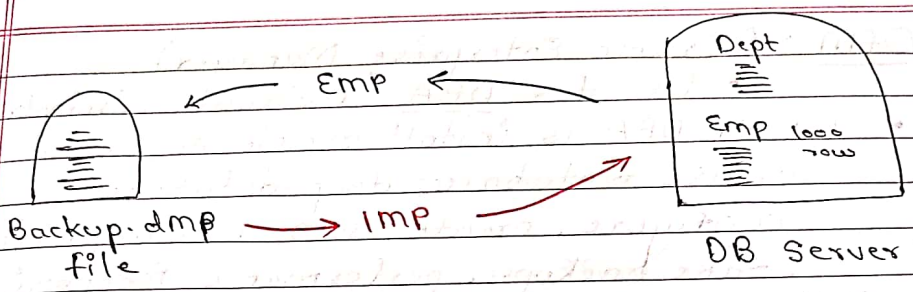
Oracle Menus :- Menu (Navbar)
full screen, bar, pulldown,
popup menus, toolbars, etc.

Oracle Graphics :- x-y graphs, bar graphs,
pie charts etc.

IDS :- (Integrated development suite)
forms + report + menu
+ graphics = IDE
(Called as frontend software
from oracle).
job of frontend :- Input & output

EXP :- (Export)
It is use for taking backup of table
data.

5.50

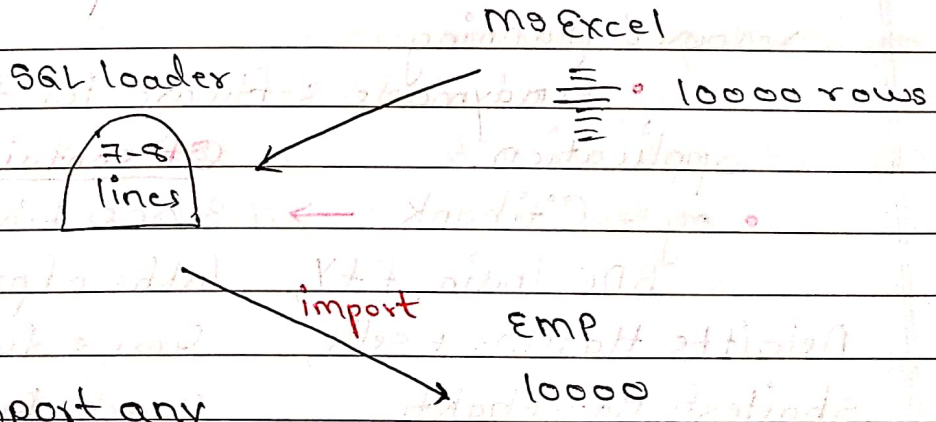


Imp :- (Import)
Used for restoring from the backups.

Oraterm :- (Oracle terminal)
Use for setting the keyboard mapping at application level.

SQL*Loader :-

- Used to load the data from other DBMS, RDBMS, languages into oracle database.



- We can import any DBMS data to oracle through SQL loader.
- Normally used at the time of migration.

OEM :- (Oracle Enterprise Manager)

- Use for DBA (Database administration)
- Job of DBA is install oracle software, create database, drop database, configure, create users, assign privileges, take backups, performance monitoring, performance tuning etc.

Oracle Case :- Case tool

- CASE :- (Computer aided Software engineering)
- If you draw a flowcharts, ERD, DfDs etc then oracle CASE will write a program and create the entire application.
- Use for RAD (Rapid Application Development)

Oracle financial

- It is an ERP (Enterprisewide resource planning.
- Readymade software for financial application.
- eg :- Citibank, BDO India, EtY, Deloitte Haskins & sells, Shailesh Haribhakti, TMC (Indian merchant Chamber) etc.

Of contain → 32000 tables, lakhs of program, Oracle forms, Oracle Report

- #1 ERP for financial application

Oracle Manufacuring

- ERP (Enterprise wide Resource planning) software.
- Readymade software for manufacuring industry

eg:- L&T, Siemens, Maruti Udyog, Whirlpool, LG, Samsung, Godrej etc.

18000 tables
lakhs of Prog Source code

#1 ERP for manufacuring industry → SAP (40%)
#2 ERP for manufacuring industry → Orade Mfg (30%)
#3 ERP for manufacuring industry → J.D Edwards (20%)

Oracle HRMS (Human resource management system)

- ERP software
- Readymade software for HRD
- eg:- Accenture, Cognizant, Capgemini, Siebel system, etc

2000 tables

#1 ERP for HRD → Peoplesoft.

Oracle CRM (Customer relationship management)

- ERP software
- Readymade software for CRM

1800 table

#1 ERP for CRM → Siebel System

Oracle Versions

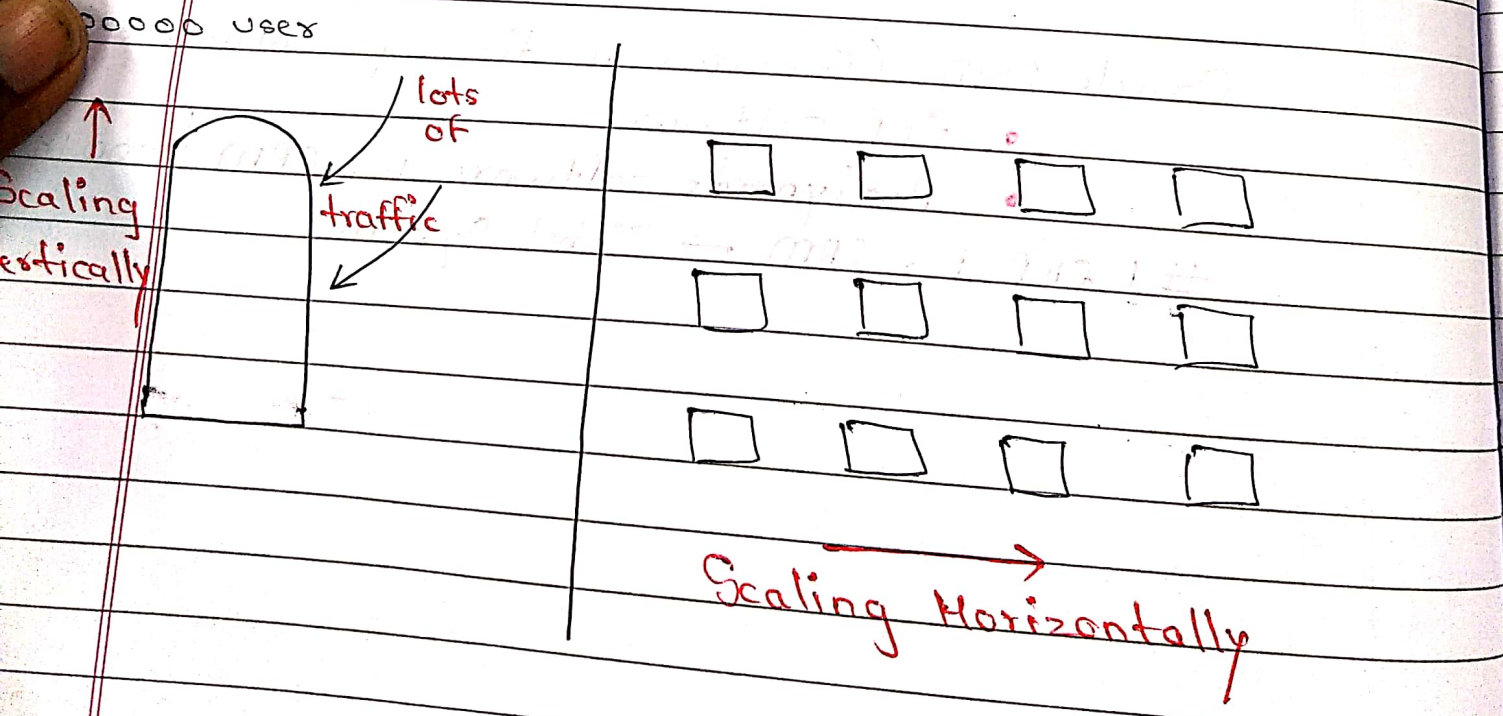
- Version 2, Version 3, Version 4 → DBMS
- Version 5, Version 6 → Partial RDBMS (tables, indexes, SQL, not null, constraint, etc introduced)
- Version 7 → 100% RDBMS (all constraints, Client-Server, Distributed database, etc introduced)

Version 8 → RDBMS + OODBMS → ORDBMS (Object Relational DBMS)

V8i, V9i → Internet computing (possible to host the database on the Internet)

V10g, V11g → Grid Computing. (Create a grid of server) lots of little boxes) Scaling horizontally

Grid of server will share the load and the resource



Install 10g, 11g / Enterprise Edition

v12c, 18c, 19c → Cloud Computing.
SAAS (Software as a service)
PAAS (Platform as a service)
IAAS (Infrastructure as a service)

Oracle Edition.

All edition are free for learning, development and testing.

1) Mobile Edition (lite Edition)
2) Personal Edition
3) ~~Standard~~ Express Edition
4) Standard Edition
5) Enterprise Edition

Mobile Edition :- Oracle lite edition
- Available on Android, Window ME, BB10, Symbian.
- free for end user.

Personal Edition :- Desktop/laptop version
- single user oracle.
- Available on MSDOS & Windows
- free for end user.

Express Edition :- Server Edition
- Server edition
- Multiuser oracle
- available Window, unix, lunix, ios,
- free for end user.
- DB Size <= 10 GB.

Standard Edition :- Server edition
multi user oracle
available on 113 OS
no limit on DB Size
Rs 2.5 lakhs / 10 user
Advanced feature such as table partitioning, index partitioning, real application cluster, etc are not available.

Enterprise Edition :- Server edition
Multiuser user oracle
available on 113 OS
no limit on DB Size
Rs 4.01/ 10 user
Complete Oracle.
Advanced feature are available.

MySQL

Software development tools of MySQL

mySQL database
MySQL Command line Client
MySQL Workbench
MySQL PL

MySQL database :- Database software.

- Enterprise and Community Edition.
- available on window, Unix, Lunix,
- ios
- v 5.7 Enterprise Edition.

MySQL Command line Client

MySQL client software
thin client
Character based (text based)
Interface with database for running SQL commands and MySQL PL program.
~~Gui~~

MySQL Workbench.

MySQL client Software
thick client
GUI based (menu driven)
Interface with database for running SQL command & MySQL PL program

MySQL PL

MySQL Programming language.
Use for database programming.

Day 2

Drop - delete the entire table.
Delete - delete a single row.

Oracle 11g - SQL

Structured Query language.
Commonly pronounced as "Sequel"
Conforms to ANSI and ISO standard.
Common for all RDBMS.
Initially founded by IBM
Earlier known as RGBE
Now Controlled by ANSI
Earlier source code → C, C++ (90%) Assembly (10%)
In 2005, 100% Java source code.

- 4 sub division of SQL (common for all RDBMS)
  - DDL (Data Defination language) (Create, drop, Alter) — add column
  - DML (Data Manupilation language) (Insert, Update, delete)
  - DCL (Data Controll language) (Grant, Revoke)
  - DQL (Data Query language) (Select)

- Extra in Oracle & MySQL
  6th Component of SQL

DTL/TCL (Data Transaction language) / (Transaction Control language)
(Commit, Rollback, Savepoint)
↓
Save & cancel

DDL (Truncate, Rename)
↳ delete & commit

Oracle Documentation & SQL

http://docs.oracle.com

- Extra in Oracle only :-

DML (Merge, Upsert)
↳ Combination of update & Insert.

- Table.

Rules for Tablename, Columnname, variablenames :-
- Maximum 30 characters
- A-Z, a-z, 0-9 allowed.
- In Oracle, tablename is case-insensitive
- In MySQL, tablename is
- Tablename has to begin with an alphabet.
- Special character such as $, #, _ allowed.
- # does not work in MySQL.
- In MySQL to use reserved character such as # in tablename and columnname enclose it in back quotes.
  Use :- ` ` → backquotes
  eg `#Emp`
- Reserved words not allowed in tablename (total-134) eg:- Create, Insert etc.

Oracle datatypes :- Char, Number } ANSI Datatype

Date
Varchar2
long
Raw
Long Raw } Extra in Oracle

- ANSI doesn't recognize DATE datatype
- DATE datatype is extra in Oracle & MySQL

Char

- Char allows any character (max 2000 character)
- It could be alphanumeric also
- eg:- PANNO, ROLL NO, AADHAR CARD_NO etc

Number

- Range → $1.0 * 10^{-130}$ to $9.9 * 10^{125}$
- eg:- SAL, comm etc.

Date

- Range → 1st Jan 4712 BC to 31st Dec 9999 AD
  - 8 Date format eg:- DDMMYYYY
- Feature :-
  - We can subtract two date
  - It returns the no of days between two dates.

  i.e date1 - date2
- Internally date is stored as number

  1st Jan 4712 BC → 1

  2nd Jan 4712 BC → 2

  3rd Jan 4712 BC → 3

  :

  :

  28th March 2019 AD → 2178936

No of days since 1st Jan 4712 BC

2nd Jan 4712 BC → 1st Jan 4712 BC = 1 day

- Internally date is a fixed length number & that number is the number of days since 1st Jan 4712 BC.
- Date Datatype occupies :- 7 bytes of storage
- 5th Oct 1582 AD to 14th Oct 1582 AD
  (Calender changed from Julian to Gregorian)
  15th Oct 1582 AD - 4th Oct 1582 AD - 1 day
- Date and Time is stored together.
  DOB :- 10-JUL-1995 11:30:45 PM
- datetime1 - datetime2 → returns number of days between the two and also the remainder hours, minutes & seconds.
- Internally time is stored as a fraction of a day
  10-JUL-1995 11:30:45 PM
  01 MAR (current mon) 00 00 00 → 12 am Midnight.
  12 am midnight is the default value of time.
- eg:- HIREDATE, ORDERDATE etc.

Varchar2

- (allows any character) max upto 4000 character
  Could be alpha numeric also.

Char	Varchar
ENAME · char (20)	ENAME varchar (20)
↓	↓
AMIT ∅ ∅ ∅ ∅ ∅	AMIT
20B	4B

- Char is a wastage of HD space
- Varchar will conserve on HD space
- If you use Char datatype, the searching & retrieval will be very fast.
- If you use Varchar2 datatype, the searching & retrieval will be slow.

eg :- ENAME, ADDRESS, CITY etc

LONG

allows upto 2 GB of character (data variable)
eg :- REMARKS, COMMENT, RESUME etc

Raw

It allows upto 2000 bytes of binary data.
eg :- BARCODES, QR_CODES, SIGNATURES, FINGERPRINT

Long Raw

It allows upto 2GB of binary data.
eg :- ICONS, PICTURES, MAPS, GRAPH, SOUND, MUSIC, VIDEOS.

long raw is the multimedia datatypes.

Oracle	MySQL
Maximum 1000 columns per table.	Maximum 4096 columns per table provided the row size <= 65,535 bytes
In Oracle, no limit on row size.	limit on row size.
You can have only one long column per table either Long or long Raw.	
Even though 1 long column is allowed per table, Oracle Coorporation recommends that you should always store the long column in a seperate table because a long column always slows down the processing speed of other columns.	
No limit on the number of rows per table and no limit on the size of table.	No limits on the number of rows per table provided the table size <= 64TB.

Emp

Emp no char (4)
Ename varchar2 (25)
Sal number (7, 2)

7 → Precision
2 → Scale

Can create
Sal number (5, 0)
OR
Sal number (5)

12345.67
↳ Scale

Code

```
Create table Emp
(
  Emp no char (4),
  Ename varchar2 (25),
  Sal number (7, 2),
  City varchar2 (15),
  Dob date
);
```

INSERT

```
insert into emp
values ('1', 'AMIT', 'Mumbai', '10-JAN-91', '5000');
```

- SQL commands are case - insensitive
- At the time of INSERT, Data is case sensitive in Oracle & MySQL
- for char, varchar2, Date use ' '

- In Oracle default date format is 'DD-MON-YY'
- In Oracle can also use 'DD MON YYYY'

10-JAN-95 → 10-JAN-1995
10-JAN-19 → 10-JAN-2019

- 1970 is the cut-off year.
- year value in the range 00-69 are taken as 2000-2069
- year value in the range 70-99 are taken as 1970-1999

'10-JAN-1969' In case of 69, give 1969.

Another way of INSERT

← Readable flexible

```
insert into emp(empno, sal, ename, city, dob)
values ('2', 6000, 'KING', 'DELHI', '15-MAR-1981');

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

↓ In future if you alter the table & add a column the insert statement will continue to work.

Emp

EMPNO	ENAME	SAL	CITY	DOB
1	Amit	5000	Mumbai	10-JAN-95
2	king	6000	Delhi	15-MAR-81
3	(IB)	7000	(OB)	(OB)
4	Atul	(OB)	(OB)	(OB)
5	(IB)	5000	(OB)	(OB)

null

-95');

- null means nothing
- null value is having ASCII value 0
- Special treatment given to NULL in all RDBMS
- Null value is independent of datatype
- You can insert a null value in a column of any datatype.
- null value occupies only 1 byte of storage
- If row is ending with null value, those columns will not occupy any space.
- Oracle Corporation recommends that those columns that are likely to large number of nulls, should preferably be specified at the end of the table structure, to conserve on HD space.

INSERT

```
insert into emp
values ('4', 'Atul');        → Error

insert into emp
value ('4', 'Atul', null, null, null);

insert into emp
values ('5', null, 5000, null, null);
```

To INSERT multiple rows at once (only support in mySQL)

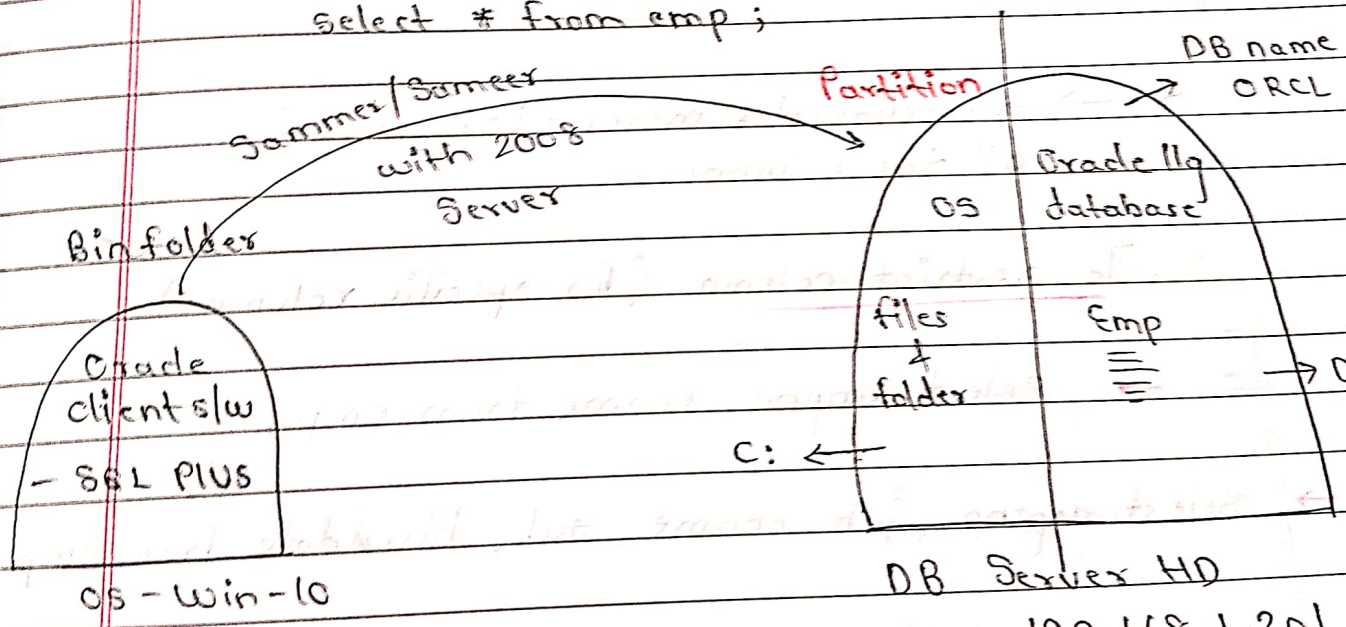
```
insert into emp value
('1', 'Amit', 5000, 'Mumbai', '10-JAN-1995');
('2', 'Atul', 6000, 'Delhi', '14-MAR-1991');
('3', null, 7000, 'Mumbai', null);
```

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

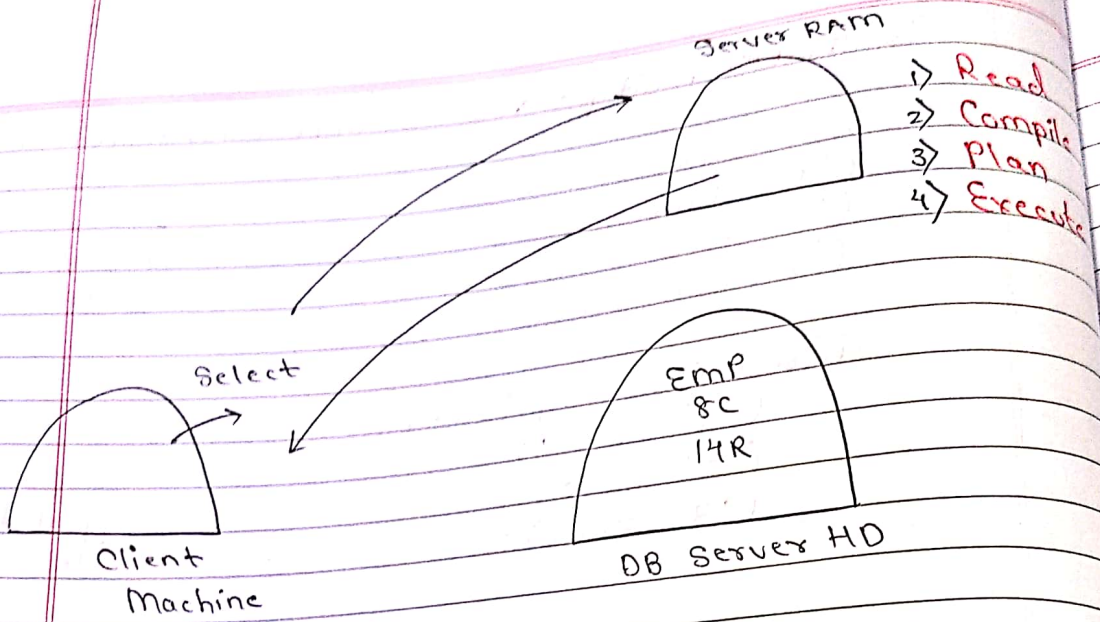
Not supported in Oracle.

SELECT → to view data

```
select * from emp;
```



Day - 3



EXIT :- It will Save & Exit.

* → metacharacter means all the rows & all the columns

To restrict column (for specific columns)

→ Select empno, ename from emp;

→ Select deptno, job, ename, sal, hiredate from emp;

→ The position of column in the SELECT statement will determine the position of columns in the output.

Day - 3

To Restrict rows

WHERE clause :-

→ Select * from emp
where deptno = 10 ;

The searching takes place in database server harddisk.

→ where clause is used to retrieve the rows
→ where clause is used to restrict the rows
→ where clause is used to retrieve the rows from DB server HD to server RAM.

→ select deptno, ename, sal from emp
where deptno = 10 ;

→ select * from emp
where sal > 2000 ;

Relational Operator

>, >=, <, <=, =
!= or <>

precedence :-

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

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

Logical Operator (with precedence)

1) NOT
2) AND
3) OR

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

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

→ select * from emp (It will shows all the rows)
where sal > 2000 or sal < 3000 Inefficient select

→ select * from emp
where job = 'MANAGER';

- At the time of insert data is case sensitive in Oracle and MySQL.
- When you're searching, queries are case sensitive in oracle. (more secure, not user friendly)
- When you're searching, queries are case insensitive in MySQL. (less secure, user friendly)

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

→ select * from emp
where job = 'MANAGER' and job = 'CLERK'; false

~~select * from emp~~

→ Select ename, sal, sal*12;
from emp;

sal * 12 is computed column (derived column)
(fake column) (Pseudo column)

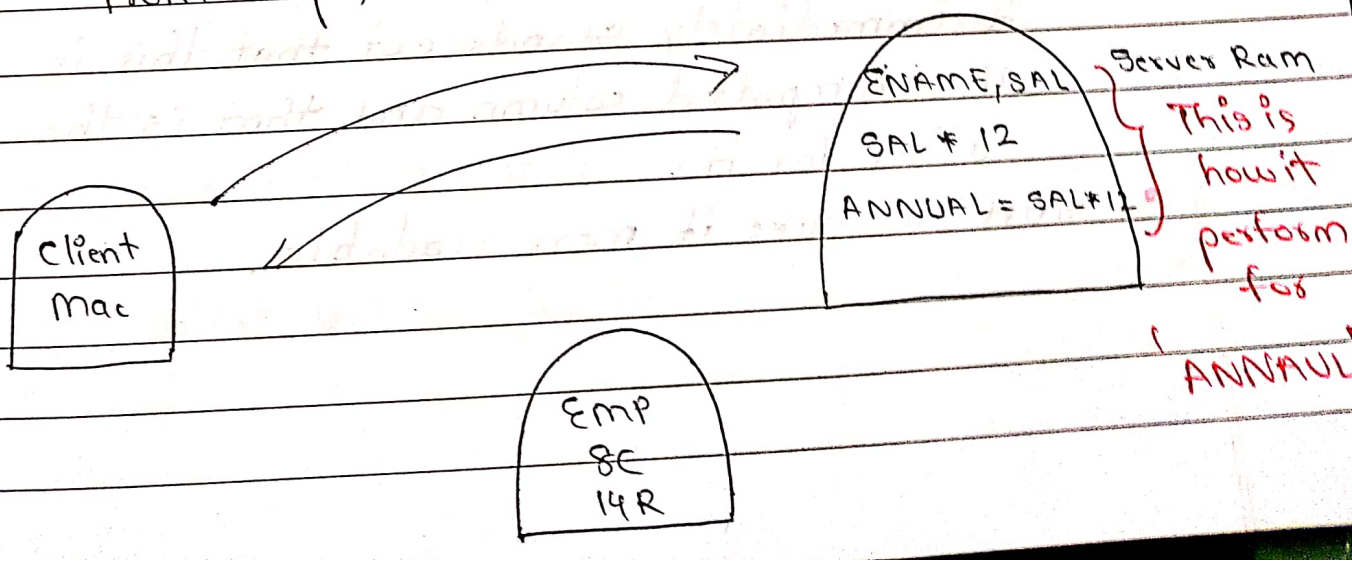
Arithmetic Operator (with precedence)

1) ()
2) ** exponentiation. sal**3 → sal^3
3) /
4) * sal**(1/3) → Cube root of sal
5) +
6) -

** does not work in SQL
** works in Oracle PL/SQL programming

Alias

select ename, sal, sal*12 as 'ANNAUL'
from emp;



Alias in MySQL :- 'AS' Keyword

Select ..... AS 'full name' from ...

```
select ename, sal, sal*12 "ANNUAL"
from emp;
```

→ Select ename, sal, sal*12 annual
from

It will show uppercase ANNUAL in the output cause by default it is uppercase. If we want to give case sensitive put it in " "

→ select ename, sal, sal*12, "annual"

→ select ename "EMPNAME",
sal "SALARY",
sal*12 "ANNUAL",
sal*12*0.4 "HRA",
sal*12*0.2 "DA",
sal*12 + sal*12*0.4 + sal*12*0.2
"NET EARNINGS"
from emp;

- It's recommended that you always use " " it immediately stands out that this is the computed column and that is the alias for it.
- Just to make it more readable.

MySQL →

MySQL → The CONCAT_WS function is use to Concat two rows items.
for eg: firstname & lastname

→ select ename, sal, sal*12 "ANNUAL"
from emp
where ANNUAL > 30000; } Error

- you cannot use alias in where clause.
- Where clause is use for searching
- Searching takes place in DB Server HD
- at the time of searching in DB Server HD the alias does not exist.
- the alias is generated in server RAM after the calculation has been performed.

→ select ename, sal, sal*12, "ANNUAL"
from emp
where sal*12 > 30000 } This will work

→ select job from emp;

→ select distinct job from emp;
This will suppress the duplicate job.

- Whenever you use Distinct Sorting takes place internally. that is the most efficient way to suppress the duplicates.

- Instead of distinct we can use Unique also

→ select unique job from emp;

- Unique is synonym for distinct.
  - distinct will work in all RDBMS & Unique will work in oracle.
  - Avoid using unique, use distinct then the same select statement can be use in other RDBMS.

→ select distinct job, ename from emp

→ select(distinct job), ename from emp Error

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

- When you insert a table, wherever Oracle finds the free space it will put the row there.
- In all RDBMS the row in a table are not stored sequentially
- In all RDBMS the rows in a table are not stored in any specific order
- The row in a table are scattered (fragmented) all over the DB server HD
- When you select from a table the order of rows in the output will always in ascending order of row address.
- hence it is not possible to see the first 5 rows or last 'N' rows of table

[If two names are same, while sorting it will be given by row/store address.]

- When you update a row the row address may change, it's only in the case of varchar2 that the row length may increase or decrease.

ORDER by clause (use for sorting)

→ select deptno, job, ename, sal, hiredate from emp order by ename; by default ascending.

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

- Order by sorting is done in Server RAM.

→ select deptno, job, ename, sal, hiredate from emp order by hiredate;

→ select deptno, job, ename, sal, hiredate from emp order by deptno, job;

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

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

- No upper limit on number of column in order by clause

select ....

order by country, state, district, city;

If you have large number of rows in the table and or large number of column in order by clause then the select statement will be slow.

```
select * from emp
where deptno = 10
order by ename ;
```

- Where clause is specified before the order by clause
- order by clause is the last clause in select statement.

→ select ename, sal from emp
order by deptno ;

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

→ select ename, sal * 12 annual from emp
order by annual ; will work
(Alias works with by Order clause)

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

→ select ename, sal * 12, "Annual Salary" from emp
order by 2 ;
(Sort by column no 2)
only in Order clause

→ Select * from emp
order by 2 ;

EMP

EMPNO	ENAME	SAL	CITY	DEPTNO
1	ADAMS	1000	Mum	10
2	BLARE	2000	Del	10
3	ALLEN	2500	Mum	20
4	KING	3000	Del	30
5	FORD	4000	Mum	40

try it out :-
ename varchar2 (10)

To Sort the names starting with 'A'
→ Select * from emp
where ename >='A' and ename <'B';

Blank-padded comparison semantics :-

In oracle when you compare 2 strings of different lengths, the shorter of the 2 strings is temporarily padded with blank spaces on RHS such that their lengths become equal; then oracle will start the comparison character by character based on ASCII value.

ADAMS > A␣␣␣␣

ADAMS < B␣␣␣␣

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

Varchar → Oracle v6 datatype
Varchar2 → Oracle v7 datatype

- Varchar2 is improvement over Varchar
- Varchar does not support Blank padded comparison semantics
- Varchar2 supports Blank-paded comparisor semantics
- In Oracle 11g if you use Varchar datatype then oracle automatically converts to Varchar2 (to maintain compatibility with older version of oracle)
- future version of Oracle may not support varchar
- Avoid using Varchar, use Varchar2

- Oracle 11g :- SQL Special Operator (like)

Wildcards -(use for pattern matching)

% any character and any number of characters

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

→ select * from emp (It will search for A%)
where ename = 'A%';

• Solution for case-insensitive query in Oracle :-

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

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

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

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

→ select * from emp
where ename like '__A%'; It will show the name with 'A' at third place.

→ select * from emp
where ename like '____'; Any four letter

→ select * from emp
where ename like '____D';

→ Select * from emp
where ename like '\%'; To search % give escape char \

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

↓ Alternative — The between method is readymade in the DB

→ select * from emp
where sal between 2000 and 3000;

It is faster than previous

between :- inclusive.

Not between :- exclusive.

faster

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

same as

→ select * from emp
where sal < 2000 or sal > 3000;

→ select * from emp
where sal between 2000 and 3000
and sal != 2500;

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

same as

compile fast

→ select * from emp
where deptno = any(10,20,30);

[Any → logical OR]

faster way

→ select * from emp
where deptno in (10,20,30)

[~~Any~~ & In → logical OR]

- IN is faster than ANY operator but then ANY operator is more powerful than IN operator.
- With IN operator, we can only check for IN and NOT IN
- With ANY operator we can check for =ANY, !=ANY, >ANY, >=ANY, <ANY, <=ANY.

MySQL :- exercise 1, 2 ,

Oracle :- assignment 1-5 ,

- If you need to check for >, >=, <, or <= then use the ANY operator.
- ANY operator is supported by oracle
- ANY operator is not supported by MySQL unless it is used with sub query

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

Day 4

→ select * from tab ; shows all the tables.

- tab → is an oracle created system table.
- 2000 system tables in Oracle 11g.
- DATA DICTIONARY - set of 2000 system tables in oracle 11g (also known as Database Catalog)

→ SQL> describe emp; → describe the structure of the table emp.

describe is an SQL * Plus command.

→ SQL> desc emp → this will work

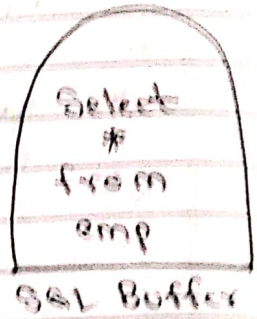
→ SQL> Password ; → SQL command.

→ In oracle 11g username is not case sensitive but password is case sensitive.

set linesize = 100
set pagesize = 25

→ select * from emp
- last SQL command is saved in SQL buffer.
- SQL buffer is a temperary text file.
- It is on client machine in BIN folder. It is called as AFIEDT.BUF

Name after all team members ← AFIEDT.BUF file



SQL Buffer

- / to execute (forward slash)
- List L It shows the last line of command which is stored in SQL buffer.

L1 → shows first line
L2 → " second line & so on.

- run command → list & execute.

→ R : It will list the last line from buffer & execute the table.

→ select * from emp
change
dept

→ L4
c/emp/dept
execute

This will change select * from dept

→ L2
c/*/ENAME
/

It will change * to ENAME & execute

→ L2
a, sal
/

a = append
It will add sal with the ENAME & execute.

→ L2
i, deptno
/

i → Insert, it will add deptno in the buffer & execute.

→ L3
del
/

It will delete 3rd line from buffer & next line will be renumbered.

- Editing command of SQL* Plus:-

C - Change
A - Append
I - Insert
Del - Delete.

→ save <filename>
save abc → to save file in buffer which in bin folder

To save somewhere else,
save c:\vital\abc.txt

→ @ <filename>
@ abc It will search a file in bin folder

To search somewhere else, give path
@ c:/vita/abc.txt.

→ get <filename> It will save to the buffer again (opposite of save)
get abc

→ save abc replace It is overwrite the existing file.

• To change system editor

→ define _ editor = notepad
define _ editor = vi
define _ editor = edit.com

It will change system editor to notepad for the current session

→ ed <filename> edit the abc file & will open in notepad cause we changed the editor
ed abc
↳ edit

PROMPT :- Use to print any statement on screen.

→ SQL> ED It will open the buffer file in the notepad.

→ Do not remove the / from editor
→ Do not add ~~the~~ one more select statement
→ Do not put sql*plus command here, only SQL command.

→ host <os command>

→ host dir *.sql /p → page wise.
  | shows all directory
  L→ having sql file

→ host dir c:\vita

→ host calc open calculator

→ host c:\vita\helloworld.exe open file

→ spool abc to record session
→ spool off It will stop recording, it won't be save if you forget to spool off.

→ ed abc.lst It will open this file in notepad

→ spool abc If you spool again it will overwrite without giving any warning.

Query :- SQL> @ demobld7.sql

@"C:\users\admin\Desktop\DBT\Oracle Demobuilder\demobld"

→ spool abc_2019_03_30_16_34

→ spool abc append (to save everything in one file)

(Whether you asce or desc, the null value are ignored.)

EMP

EMPNO	EMPNAME	SAL	CITY	DEPTNO
1	ADAMS	1000	Mumbai	10
2	BLAKE	2000	Delhi	10
3	ALLEN	2500	Mumbai	20
4	FORD	3000	Delhi	30
5	KING	4000	Mumbai	40

UPDATE

- It is an DDL command.

→
```
update emp
set sal = 10000
where empno = 1;
```
Salary of ADAMS will be 10000 now

→
```
update emp
set sal = sal + sal * 0.4
where empno = 1;
```
Salary of ADAMS will increase by 40% of sal.

→ update emp
set sal = 10000, city = 'Pune'
where empno = 1;
It will change sal as well as city.

→ update emp
set sal = 10000
where city = 'Mumbai';
It will change the sal of employee from mumbai city.

→ update emp
set sal = 10000, city = 'Pune'
where city = 'Mumbai';
It will change sal and city also.

→ update emp
set sal = 10000;
It will change sal of all the rows.

- We can UPDATE multiple rows and multiple columns simultaneously but only one table at a time.
- Seperate UPDATE command would be required for every table.

DELETE

It is a DML command.

→ delete from emp
where empno = 1;
It will delete the empno 1 row.

The keyword FROM is ANSI SQL
The keyword FROM is optional in Oracle
The keyword FROM is compulsory in MySQL.

→ delete from emp
where city = 'Mumbai';

→ delete from emp;
All the rows will be deleted, we can add (insert) later.

drop table emp;
It will drop the table.
↓
DDL command.

→ We cannot use where clause with drop table.

→ If you want to drop multiple table, you have to drop each table separately.

MySQL

→ When you install MySQL 'root' user is automatically created.
→ root user in MySQL is having DBA privileges.

To connect to MySQL database :-

1) OPen MySQL Workbench.
2) MySQL Connection (Click on + sign to create a new connection)
3) Connection name :- Root user Test Connection.
4) Connection method :- Choose TCP/IP.
5) Hostname : localhost
6) Portno : 3306
7) Username : root
8) Password (Store in Vault ... Push button) → click on it to enter the password.

In MySQL, SCHEMA is synonym for database.

9) Test Connection push button :- click on it
10) Ok push button.
11) Some basic commands post login :-

```
show database;
ctrl + enter
use <databasename>;
    or
use <schemaname>;
```

use mysql
select * from user
show tables;
desc emp;

- To create a Schema/Database and a user:-

create database vita_db;
or
create database vita_db;
show database;

Create user <username> identified by <password>

create user sai identified by
↓ < >

create user sai identified by 'sai123';

To grant permissions:-

click on server (menu at the top) → user and privileges → click on it → select the username that you created from the user Account list on LHS

click on Administrative Roles tab

select DBA Role and click on Apply push button

click on Schema privileges tab

click on add entry ... push button.

click on radio button selected schema → choose vita_db → click on ok.

Select All push button → click on it
Select Grant option checkbox also
Apply push button → click on it.

Exit from MySQL Workbench and create a new connection for lai user with default schema as vita_db

Use vita_db ← can also give this command & connect to it later.

UPDATE & DELETE command without where clause will not be allowed in MySQL Workbench.

Changes to make it work

Edit (menu at top) → Preferences → SQL editor → "Safe Update" checkbox at the bottom → Uncheck it → Click ok

Query (menu at top) → Reconnect to server

TRANSACTION PROCESSING

- Commit will save all the DML changes since the last committed state

SQL>	Insert	Update
	Insert	Update
	Insert	Delete
	Commit;	Delete
		Commit;

SQL> Commit;
SQL> Commit work;

work is ANSI SQL
Optional in Oracle & MySQL

Commit :
- When user issue a commit ; It is known as end of transaction.
- Commit will make the transaction permanent.

Work = $T_1 + T_2 + T_3 \cdots T_n$
- Transaction is a unit of work.
- When to issue a commit depends upon the logical scope of work
- Rollback will undo all DML changes since the last committed state.

→ rollback
or
rollback work;

- Only the DML command are affected by Rollback and Commit
- any DDL command is automatically commit
- When you EXIT from SQL*PLUS, it automatically commits.
- Any kind of power failure, system failure, network failure, PC reboot, window close, improper exit from SQL*Plus, etc your last uncommitted transaction is automatically rollback.

To tryout Rollback and Commit in MySQL

Query (menu at the top) -> Auto-Commit transaction -> Uncheck it

In Oracle

SQL> set autocommit on
SQL> set autocommit off
by default it is off

SQL> show autocommit — to check status

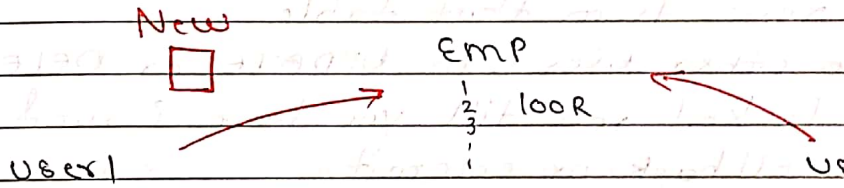
INSERT
INSERT
INSERT
SAVEPOINT ABC ;
UPDATE
UPDATE
SAVEPOINT PQR ;
DELETE
DELETE
~~SAVEPOINT TO PQR~~
ROLLBACK TO PQR ;

- you can rollback to a savepoint
- You cannot commit to a savepoint
- Commit will always save all the DML changes since the last committed state
- savepoint is a point within the work
- Savepoint is sub-unit of work
- savepoint is similar to bookmark
- When you rollback or commit, the intermediate safepoint will be cleared.
- If you want to use those savepoint again, you will have to reissue them in some new work.
- Savepoint is use in large & small transactions.
- for the savepoint name, max 30 characters, for the savepoint name, give a meaningful name
  eg:- savepoint JOURNEY_DETAIL, savepoint PASSENGER_DETAIL, savepoint PAYMENT_DETAILS etc.

- Within a transaction, you can have 2 savepoints with the same name
- The latest savepoint overwrite the previous one
- The older savepoint no longer exists.

Read and Write Consistency

- When you Select from a table you can view only the committed data of other users plus/union changes made by you.



User1	User2
Select 100R	Select 100R
Insert 10R	Select - 100R
Select 110R	
Commit	
Rollback	
Select -100R	

- When you UPDATE or DELETE a row, that row is automatically looked for other users.
- Row locking is automatic in oracle & MySql
- When you UPDATE or DELETE a row, that row becomes READ-ONLY for other users.
- Other users can SELECT from that table; they will be able to view the old data before your changes.
- Other users can INSERT ~~or DELETE~~ rows in the table.
- no ~~other~~ user can UPDATE or DELETE other rows from that table
- No other user can UPDATE or DELETE your locked row till you have issued a rollback or commit.
- LOCKS are automatically released when you rollback or commit.

OPTIMISTIC ROW LOCKING

- Automatic row locking mechanism in Oracle and MySQL.

PESSIMISTIC ROW LOCKING

- Before issuing update or delete we lock the ~~door~~ row manually in advanced ~~before issuing update or delete~~.

To lock the rows manually we have to use select statement with the for UPDATE clause.

→ SQL> select * from emp
for update;

It will lock ~~the~~ manually all the rows.

→ SQL> select * from emp
where deptno = 10
for update.

→ SQL> select * from emp
where deptno = 10
for update wait 60; second.

→ SQL> select * from emp
where deptno = 10
for update nowait;

- Locks are automatically released when you Rollback or commit.

To try out row locking in Mysql

Click on Query (menu at top)
New tab to current server
Click on it

You now have 2 query windows to try out locking.

In MySQL :-
Sql exs 1, 2

In Oracle
Sql Progms
1, 2, 3, 4, 5, 6

To Abort the operation in mysql

Click on Query (menu at the top) → stop → click on it.

In MySQL
WAIT & NOWAIT is not available.

Days

Oracle 11g - SQL Character function

EMP

FNAME	LNAME		
Arun	Puran	Nutan	Puran
Tarun	Arun		
Siran	Kiran		

In Oracle
fname Varchar2 (15)
Lname Varchar2 (15)

In MySQL
fname Varchar (15)
Lname Varchar (15)

|| → Concatenation operator.

→ select fname || Lname from emp;

o/p :- ArunPuran
TarunArun
SiranKiran
NutanPuran

→ select fname||' '||lname emp;

o/p :- Arun Puran
Tarun Arun
Siran Kiran
Nutan Puran

→ select 'Mr.'||fname||' '||lname from emp;

o/p :- Mr. Arun Puran

- || work in oracle
- || does not work in MySQL.

In MySQL
We can; concat(str1, str2)
→ select concat(fname, lname) from emp;

In Oracle
Nested function (max upto 255 level for fun)

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

o/p :- Arun Puran.

In MySQL

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

o/p :- Arun Puran.

→ select upper(fname) from emp; display in uppercase.

o/p :- ARUN
TARUN
SIRAN
NUTAN

→ update emp set fname = upper(fname);
It will update all the column of fname to uppercase.

Case insensitive Query in oracle.

→ select * from emp where upper(fname) = 'ARUN';

→ select lower(fname) from emp;

o/p :- arun
tarun
siran
nutan

→ update emp set fname = lower(fname);
all the rows of fname will be in lower case.

Case Insensitive query in oracle

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

(update the table name to ENAME.)

select initcap(ename) from emp;

o/p: Arun Puran
Taxun Arun

first letter in upper case & rest in LC

→ select * from emp where ename = initcap(ename);
It will change all the rows in emp

- initcap work in Oracle
- initcap not supported in MySQL

→ select lpad(ename, 25) from emp;

ex:- XXXXXArun Puran

15 25

Right justification of character data.

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

eg:- (* * * * * Arun Puran)
25

It can be use for cheque printing.

→ select lpad (ename, 25) from emp
Not supported in MySQL

Solution for mySQL
→ select lpad (ename, 25, ' ') from emp;

→ select rpad (ename, 25) from emp;

Arun Puran $ $ $ $ $
25

- left justification of numeric data
- to convert varchar2 to char (to convert variable length to fixed length)
- cheque printing

- Centre justification (R&D)
  * * * Arun * * *

→ select ltrim(ename) from emp;
It will remove the blank spaces from left hand side.

update emp set ename = ltrim(ename);
We can do this, but not recommended.

select ltrim(ename,'*') from emp;

select ltrim(ename,'* ') from emp;
It will remove * as well as space.

(You cannot specify character(s) of our choice in MySQL.)

→ Select rtrim(ename) from emp;
→ select rtrim(ename,'*') from emp;
→ Select rtrim(ename,'* ') from emp;

- to convert char to varchar2 (fixed length to variable length)
~~cannot specify~~
- right justification of char data
eg :- lpad(rtrim(ename)....)

→ select trim(ename) from emp;

- remove blank spaces from both the sides.

→ select substr(ename, 3) from emp;

3 → starting position

o/p :- un Puran
run Arun
run Kiran
tan Puran

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

3 → starting position
2 → number of character

o/p :- un
ru
ru
ta

Solution for initcap in Mysql

concat(upper(substr(ename, 1, 1)), lower(substr(ename, 2)))

→ select substr (ename, -3) from emp;
extract from right & print

o/p run
run
run
than

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

o/p :- ru
ru
ru
ru

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

un → xy o/p :- Arxy Purxy

→ select replace (ename, 'un', 'xyz') from emp;

→ select replace (ename, 'un') from emp;

Not supported by MySQL

un →
o/p :- Ar Pur
Tar Ar
Sir Kir
Nutan Pur

Third parameter is compulsory in MySQL.

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

u → x
n → y

single char is affected char by char.

o/p :- Arxy Pxrxy

→ select translate (ename, 'un', xyz) from emp;

u → x
n → y
→ z

o/p same as above.

→ select translate (ename, 'un', 'x') from emp;

u → x
n →

n will be removed from ~~char~~ ename.

→ select instr (ename, 'un') from emp;

o/p: 3
4
4
10

- Instr is char fun return numeric value.
- only for first occurrance
- If string is not found it returns zero

We can use instr to check if one string exist in another one.

→ select instr (ename, 'un', 4) from emp;

4 → starting position

o/p :- Arun Puran → 9
Tarun Arun → 4
Sirun Kirun → 4
Nutan Puran → 10

→ select instr (ename, 'un', 4, 2) from emp;

4 → starting position
2 → Second occurrence

o/p :- 0
9
10
0

→ select instr (ename, 'un', -4) from emp;

9
9
10
10

→ select instr (ename, 'un', -4, 2) from emp

0
0
0
0

• Using instr function the 3rd & 4th paramcter are not allowed in MySQL

EMP
ENAME
Arun
Bannerjee
Charlie

→ select length (ename) from emp;

o/p 4
9
7

Give the length of ename

→ select ascii (ename) from emp;

only works for 1st letter

o/p :- 65
66
67

→ select ascii (substr (ename, 2)) from emp;

→ select ascii ('z') from emp;

o/p :- 122
122
122

Cause emp has 3 rows

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

o/p :- 122

→ select distinct ascii ('z') from tab;

o/p :- 122

→ select ascii ('z') from dual;

o/p :- 122

Dual :- Dual is a system table.
It is a dummy table.
It contain only 1 row & 1 column.

→ select 'Welcome to VITA' "MESSAGE"
from dual

→ select 8*12 from dual;

→ select substr ('New Mumbai', 1, 3) from dual;

→ select chr (65) from dual ;

o/p :- A

returns the character correspond to ASCII value.

- Oracle is available in 1290 language
  It is also available in Indian lang
  ज → 4 byte
  (Multi-byte character)
  Chinese → 8 byte
  Japanese → 16 byte
  Arabic → 32 byte

- SQL is available in 43 language
  It is not available in Indian lang.

Natural character

nchar → max 2000 bytes of natural character data

nvarchar → max 4000 bytes of natural character data

In MySQL

→ select char(65 using utf8) from dual;

- where utf8 is given character set for US English size default is binary character ~~size~~. set

→ Select * from emp where soundex(ename) = soundex('Aroon');

If the sound of the name is similar/ match it will print

It removes the vowel from both the string then it compares

try for simple names (not so accurate function).

- Number function

emp
SAL
1234.567
1875.019
1348.618
1652.651

sal number (7,3)
→ select round(sal) from emp; overloading

o/p :- 1284
1875
1849
1653

→ select round(sal, 1) from emp;

o/p :- 1234.6
1875
1348.6
1652.7

→ select round(sal, 2) from emp;

o/p :- 1234.57
1875.02
1348.62
1652.65

→ select truncate (sal, -2) from emp;

o/p :- 1200
1900
1300
1600

In Oracle

→ Select trune (sal, 0) from emp; } same
→ select trune (sal) from emp;

o/p :- 1234
1875
1348
1652

Ceil → Ceiling
If there is any value at all in decimal, then it will add 1 to the whole number

→ select ceil(sal) from emp;

o/p :- 1235
1876
1349
1653

use in billing, interest

Sign
Uses :-

(lateral thinking by Ayn)

→ select floor(sal) from emp;
remove decimal

o/p :- 1234
1875
1348
1652

→ select sign(-15) from dual; It may return
-1
o/p :- -1 1
0

Sign Uses :-
1) Checks if num is +ve or -ve
2) sign(temperature)
3) sign(stockmarketsensex)
4) x=10, y=5, sign(x-y)
5) x * sign(x)
6) sign(axis)
7) sign(bankbalance)
8) sign(marks)
9) sign(altitude)
10) sign(distance)
11) sign(SP-CP)

→ select mod(9,5) from dual; print the remainder
o/p :- 4

→ select mod(8.22, 2.2) from dual;
o/p :- 1.62

→ select sqrt(81) from dual;
o/p = 9

→ select power(10,3) from dual;
o/p = 10^3

→ select power(10, 1/3) from dual;

- ** does not work in SQL
- ** works in Oracle PL/SQL programming to do exponentiation we have to use power function.

→ select abs(-10) from dual;
o/p :- 10

→ sin(y)
sinh(y)
cos(y)
cosh(y)
tan(y)
tanh(y)

y → radians
If data is in degree
$\frac{\pi}{180}$

sinh(y), cosh(y) & tanh(y) will not work in MySQL.

$\ln(x) \rightarrow \log_e(x)$

$\log(n, m) \rightarrow \log_n(m)$

Oracle 11g – SQL Date function and formats

EMP
HIREDATE

15-OCT-15
31-DEC-15
15-JAN-16

- 1st Jan 4712 BC to 31st Dec 9999 AD
- Default format :- 'DD-MON-YY'
- 1970 is cutoff year (Year of the Epoch)
- date1 - date2

  Internally date is stored as a fixed length
  → number of days since 1st Jan 4712 BC
  → known as Julian date.
- 5th oct 1582 AD to 14th oct 1582 AD.
- 7 bytes of storage.
- date & time is stored together
- default value of time 12am midnight
  date1 - date2 → number of days between the 2 dates; remainder hours, minutes and seconds as a fraction of day.
- 78 date format.

→ select sysdate from dual;

In MySQL

→ select sysdate() from dual;

sysdate is a function and it returns the database server machine date & time.

→ select sysdate()+1 from dual;

→ select sysdate()-1 from dual

→ select sysdate - hiredate from emp;

819.808
793.808
735.808

→ select (sysdate - hiredate)/7 from emp;

→ select months_between(sysdate, hiredate) from emp;

→ select trunc (months_between (sysdate, hiredate)) from emp;

no of months

→ select trunc (months_between (sysdate, hiredate))/12 from emp;

no of years

→ select add_months (hiredate, 2) from emp;

o/p :- 15-DEC-15
24-FEB-16
15-MAR-16

→ select add_months (hiredate, 12) from emp;

will add a year

→ select last_day (hiredate) from emp;

last day of the month

o/p :- 31-OCT-15
31-DEC-15
31-JAN-16

uses :- Salary calculation
Attendance

→ select next_day (sysdate, 'Tuesday') from dual

date of coming tuesday

→ select next_day (sysdate - 7, 'tuesday') from dual

date of previous tuesday

Oracle 11g - SQL Data formats (DD-MON-YY)

→ select sysdate from dual;

01-APR-19

→ select to_char(sysdate, 'DD/mm/YY') from dual;

It will convert sysdate into format & convert into character format.

→ select to_char(sysdate, 'DD\mm') from dual;

o/p 01/04

We can use

DD	dd	Dd
mm	mm	Mm
YY	yy	Yy

→ select to_char(sysdate, 'DD-mm-YY') from dual;

/ → British seperator
- → American seperater
. → German seperator
: → French seperator
* → Italian seperator

→ select to_char(sysdate, "DD mm YY") from dual;

use :- pre printed stationary

Date :- [ ][ ][ ][ ]

→ select to_char(sysdate, "DD-mm-YYYY") from dual;

o/p :- 01-04-2019

we can give

1) YYY YYY

2) YY YY

3) Y Y

→ select to_char(sysdate, "DD-MONTH-YY") from dual;

o/p :- 01-APRIL-2019

→ select to_char(sysdate, 'DAY') from dual

o/p :- MONDAY.

→ select to_char(sysdate, 'DY') from dual;

o/p :- MON

→ select to_char(sysdate 'YEAR') from dual

o/p :- TWENTY NINETEEN

→ select to_char (sysdate, 'DDTH') from dual;

o/p :- 01ST

→ select to_char (sysdate, 'DDSP') from dual;

o/p :- ONE

→ select to_char (sysdate, 'DDSPTH') from dual;

o/p :- FIRST

format :-	YEAR	DD	mm	DD (1-31)
	year	DDTH	mmTH	D (1-7)
	Year	DDSP	mmSP	DDD (1-366)
		DDSPTH	mmSPTH	

W (1-5) - week of the month

WW (1-53) - week of the year

select to_char (sysdate, 'J') from dual

↓

Julian

J → Julian date (num of days since 1st Jan 4712 BC)

→ select to_date('10-JAN-2015','DD-MON-YYYY')
+1 from dual;
Add the date by 1

→ select to_date('04-OCT-1582','DD-MON-YYYY')
+1 from dual;
o/p:- 15-OCT-1582

→ insert into emp
values (to_date('2016/15/10','YYYY/DD/MM'));
to print or insert specific format

→ alter session set nls_date_format =
'DD/MM/YYYY';
It will change the date format for
current session.

(NLS :- National language support)

Global login
This file is on Client Machine
This file is startup file of SQL* Plus
globin.sql
This file is automatically executed when
you start SQL* Plus on your computer
.../sqlplus/admin>

Time format

→ select to_char(sysdate, 'DD/mm/YYYY HH:MI') from dual
01-04-2019 09:01

→ select to_char(sysdate, 'DD/mm/YYYY HH:MI:SS') from dual
01-04-2019 09:01:54

HH	am	pm
MI	AM	PM
SS	a.m	p.m.
HH24	A.M	P.M.

→ select to_char(sysdate, 'DD/mm/YYYY HH:MI:SS am') "MUMBAI"

→ select to_char(sysdate, 'DD/mm/YYYY HH:MM:SS am' 6.5/24 "LONDON"

sysdate + 1
sysdate + 1/24
sysdate + 1/(24*60)
sysdate + 1(24*60*60)

3) Select .... SYSDATE - .... HIREDATE
display the experience of employee
in years, remainder month,
hours, minutes, seconds.

ex:-
3 years 7 month 19 days 11 hours
35 minutes 12 Seconds

4) Select ... spell out the sal$ upto
5 million round, truncate

o/p :- Three thousand five hundred and one

→ Select to_char(to_date(123, 'J'), 'JSP' in_words from
dual;

o/p:- One hundred twenty three.

Day 6

Oracle 11g - SQL list functions (Independent of datatype)

Table: emp

ENAME	SAL	comm
A	5000	500
B	6000	.
C		700

Any Comparison done with null, returns null.

pessematic querying → searching for null value

To make this work

→ select * from emp where comm is null;

→ select * from emp where comm is not null;

→ select sal + comm from emp;

Any Operation done with null returns null.

In Oracle :-

→ select sal + nvl(comm, 0) from emp;

→ select nvl(sal, 0) + nvl(comm, 0) from emp;

nvl working (null value)

If commission is null then return 0 else return the comm.

- nvl can work with all datatype

→ nvl (comm, 0)
→ nvl (comm, 100)
→ nvl (city, Mumbai)

In MySQL

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

ifnull (comm, 0)
ifnull (comm, 100)
ifnull (city, 'Mumbai')
ifnull (orderdate, '01-APR-19')
ifnull (comm, tax)

Table:- EMP

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

Greatest

→ select greatest (sal, 3000) from emp;

o/p:- 3000
3000
3000
4000
5000

Use:- To set a lower limit on some value.

→ select greatest (sal*0.2, 500) "BONUS" from emp;

- We can compare upto max 255 value in greatest function
- It will work with all datatype.

→ greatest (num1, num2, num3, num4)
→ greatest ('str1', 'str2', 'str3');
→ greatest ('date1', 'date2', 'date3')

x = greatest (a, b, c, d); (In program)
It will compare a, b, c, d & assign greatest number to x.

least

→ select least (sal, 3000) from emp;

o/p :- 1000
2000
3000
3000
3000

Use :- to set an upper limit on some value
eg:- cashback = 10% of amount, max cashback is ₹ 2000.

eg:- → least (amount * 0.1, 2000)

→ least (value1, value2, value3 ... value 255)

→ least (num1, num2, num3, num4);
→ least ('str1', 'str2', 'str3', 'str4');
→ least ('date1', 'date2', 'date3', 'date4');

Decode

The most powerful function in sql

→ select decode(deptno, 10, 'Training' 20, 'Exports', 30, 'sales', 'others')
from emp;

If deptno 10 returns Training
" " 20 returns Export
" " 30 returns Sales
" anything returns Others

- If others is not specified & if we pass anything it will return null

→ select decode (deptno, 10, 'Ten', 20, 'Twenty', 30, 'Thirty', 40, 'forty') from emp;
spell out numbers

→ select ename, sal, sal * 12 "ANNUAL", decode (deptno, 10, sal * 12 * 0.4, 20, sal * 12 * 0.3, 30, Sal * 12 * 0.2, Sal * 12 * 0.1) "HRA" from emp;

if sal < 3000 then REMARK = 'low Income'
if sal = 3000 then REMARK = 'middle Income'
if sal > 3000 then REMARK = 'High Income'

→ select ename sal,
decode (sign (sal - 3000), 1, 'High Income',
-1, 'low Income', 'middle Income')
"REMARKS" from emp
order by 2 ;

- DECODE not supposted by MySQL
- In MySQL we can use the Case expression
  (Case also works in Oracle)

→ select case
when deptno = 10 then 'Ten'
when deptno = 20 then 'Twenty'
when deptno = 30 then 'Thirty'
else 'Others'
end "DEPTNUMBER"
from emp;

If we don't specify else, it will return null for others

vsize - returns size in bytes

→ select ename, vsize(ename) from emp;

- If data is in some other language we use vsize

→ select vsize(deptno), vsize(dname), vsize(loc) from dept

Add all using '+' will get the total size (bytes) of dept table.

- Environment function (Oracle)

→ select user from dual;

show username

→ select user, uid from dual;

→ select userenv('terminal') from dual;

returns computer name

→ select userenv('sessionid') from dual;

→ select userenv('language') from dual;

In MySQL - Environment function.

→ select user() from dual;

Implement this in project

→ insert into emp
values (1000, 10, user, sysdate, user env('terminal'));

- used to maintain logs (audit trails) of DML operation.

Oracle 11g - SQL Group / Aggregative function.

emp

EMPNO (parent column)	ENAME	SAL	DEPTNO	JOB	MGR (child column)
1	Arun	8000	1	M	4
2	Ali	7000	1	C	1
3	Kirun	3000	1	C	1
4	Jack	9000	2	M	
5	Thomas	8000	2	C	4

→ select decode (job, 'm', 'MANAGER', 'c', 'CLERK')
from emp;

→ Select case
when job = 'm' then 'm

Mon:

Single Row function

Group function

→ Select sum(sal) from emp;

o/p :- 35000

Assumption last row sal is null :-

o/p :- 27000

→ Null values are not counted by group functions.

→ select avg(sal) from emp;

o/p :- 27000/4 = 6750

→ select avg(nvl(sal, 0)) from emp; X

o/p :- 27000/5 = 5400

→ Select min(sal) from emp;

o/p :- 3000

→ select max(sal) from emp;

o/p :- 9000

→ select count(sal) from emp;

returns count of no of rows where not having null values

o/p :- 4

→ select count(*) from emp; returns total no of rows in the table.
o/p :- 5

→ select count(*) - count(sal) from emp;
o/p :- 1

select sum(sal) from emp
where deptno = 1;
o/p :- 18000

select max(sal) from emp
where job = 'm';
o/p :- 9000

Count Query

select count(*) from emp
where sal > 7000;
o/p :- 2

• We cannot use Group function in the WHERE clause.

→ select * from emp // error
where sal > avg(sal);

→ select max(sal)/min(sal) from emp
o/p :- 3

→ select sum(sal)/count(*) from emp;
o/p :- 5400 faster (one loop)

→ select avg(nvl(sal,0)) from emp;
o/p :- 5400 slower (two loop)

1) sum(column) } only work with oracle.
2) avg(column) }
3) min(column) } Works with all datatype
4) max(column) }
5) count(column) }
6) count(*) } only work with oracle.
7) stddev(column) }
8) variance(column) }

SUMMARY REPORT :-
→ select count(*), min(sal), max(sal), sum(sal), avg(sal) from emp;

* 3 limitation of Group function

1) We cannot select regular column along with group function.
→ select ename, min(sal) from emp;
In oracle it gives error
In MySQL it works but the output is meaningless.

2) We cannot select single row function along with a group function.

→ select upper(ename), min(sal) from emp;

It works in MySQL

3) You cannot use group function in the WHERE clause

```
select * from emp
where sal > avg(sal);
```

error in all RDBMS

Oracle 11g - SQL Group by clause (used for grouping)

→ Select sum(sal) from emp;

o/p :- 35000

→ select sum(sal) from emp
where deptno = 1;

o/p :- 18000

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

o/p :-

DEPTNO	SUM(SAL)
1	18000
2	17000

Steps :- How group by clause work internal

1) The rows are retrieved from server HD to server RAM.
2) Sorting will be done
3) Grouping is done department wise.
   It breaks the array & store randomly by group no
4) Summation / calculation (sum)
5) HAVING clause
6) ORDER BY clause

SELECT clause → select deptno, sum(sal)
FROM clause → from emp
GROUP BY clause → group by deptno

Rule #1

Besides the group function, whichever column is present in SELECT clause, it has to be present in the GROUP by clause

select deptno, sum(sal) from emp;
error

Rule # 2

Whichever column is present in GROUP by clause it may or may not be present in SELECT clause

→ select sum(sal) from emp
group by deptno

o/p :-
SUM(SAL)
18000
17000

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

→ select deptno, min(sal) from emp
group by deptno;

→ select deptno, max(sal) from emp
group by deptno

→ select deptno, count(*) from emp

"Any query with group by clause is known as 2-D Query because we can plot a graph from the output"

→ select deptno, sum(sal) from emp
where sal > 7000
group by deptno ;

o/p :-

Deptno	Sum(sal)
1	8000
2	17000

- WHERE clause is specified BEFORE the GROUP BY clause.

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

outer loop
Deptno
Job
inner loop

o/p :-

Deptno	Job	Sum(sal)
1	C	10000
1	M	8000
2	C	8000
2	M	9000

No upper limit on the number of columns in GROUP by clause

→ select .....
group by country, state, district, city;

classmate
Date
Page

- If you have large number of rows in the table and large number of rows in GROUP BY clause then it will be very slow (because that much sorting takes place internally

outer job
Deptno
inner

→ select ~~dept~~job, deptno, sum(Sal) from emp group by job, deptno ;

- The position of column in select clause & position of column in group by clause need not be the same.
- The position of columns in SELECT clause will determine the position of columns in the output.
- The position of column in GROUP BY clause will determine the sorting order, grouping order, summation order, and hence the speed of processing

select .....
group by state, city, district, country
slow

select .....
group by country, state, district, city
fast

→ set timing on
It will show the time taken by the process.

→ select deptno, sum(sal) from emp
group by deptno
having sum(sal) > 17000

o/p :- DEPTNO SUM(SAL)
1 18000

- WHERE clause is used for searching
- searching takes place in the DB server HD
- WHERE clause is used to restrict the rows
- WHERE clause is used to retrieve the rows from DB server HD to server RAM
- HAVING clause works in the server RAM
- HAVING clause work AFTER the sumation is done

It is recommended that only group function is use in HAVING clause

→ select deptno, sum(sal) from emp
group by deptno
having sal > 7000 Error
sal doesn't exist, Sum(sal) is present
↓
group function

→ select deptno, sum(sal) from emp
group by deptno
having deptno = 1;
will work but is inefficient

order by clause is last statement in select

→ select deptno, sum(sal) from emp
group by deptno
having count(*) = 3;

- In the HAVING clause we may use the group function that is not present in the SELECT clause. (that group function will not be displayed in the output but it is calculated by Oracle for the HAVING clause.

→ select deptno, sum(sal) from emp
group by deptno
having sum(sal) between 17001 and 24999;

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

o/p :-

DEPTNO	SUM(SAL)
1	18000
2	17000

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

o/p :-

DEPTNO	SUM(SAL)
2	17000
1	18000

```
select ..... from .....
where .....
group by .....
having .....
order by .....
```

Syntax for select statement.

In Oracle

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

o/p:

DEPTNO	SUM(SAL)
1	18000
2	17000

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

o/p

SUM(SAL)
18000
17000

→ select max(sum(sal)) from emp
group by deptno;

MAX(SUM(SAL))
18000

Nesting of group function is allow only in Oracle RDBMS.

In MySQL

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

o/p SUM(SAL)
18000
17000

→ select max(sum_sal) from
(select sum(sal) SUM_SAL from emp
group by deptno) as tempp;

o/p MAX(SUM_SAL)
18000

MATRIX REPORT

→ select deptno, count(*), min(sal),
max(sal), sum(sal) from emp
group by deptno
order by 1;

DEPTNO	COUNT(*)	MIN(SAL)	MAX(SAL)	SUM(SAL)
10	3	1300	5000	8750
20	5	800	3000	10875
30	6	950	2850	9400

SQL JOINS

child table (Details table)

DEPT

DEPTNO	DNAME	LOC
1	TRN	Bby
2	EXP	Dlh
3	MKTG	Cal

Parent table (master table)

EMP

EMPNO	ENAME	SAL	DEPTNO	JOB	MGR
1	Arun	3000	1	M	4
2	Ali	7000	1	C	1
3	Kiran	3000	1	C	1
4	Jack	9000	2	M	.
5	Thomas	8000	2	C	4

Data Redundancy :- Unnecessary duplication of data.

Joins Uses :- 1) To view the columns of 2 or more table.

→ Select dname, ename from emp, dept where dept.deptno = emp.deptno

dept → driving table
emp → driven table

tablename.columnname

o/p:-

DNAME	ENAME
TRN	Arun
TRN	Ali
TRN	Kiran
EXP	Jack
EXP	Thomas.

- (In order for the join to work faster preferably the driving table should be table with lesser number of rows)

- Common column in both the tables, the columnname need ~~be the~~ not be the same in both the table, because the same column can have a different meaning elsewhere.
- What matters is the datatype of the column has to match in both the tables and there has to be some relation on whose basis we are writing the join.

Day 7

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

→ select dname, loc, ename, job, sal from
emp, dept
where dept.deptno = emp.deptno;

→ select * from emp, dept
where dept.deptno = emp.deptno;

→ Select deptno, dname, loc, ename, job, sal from
emp, dept
where dept.deptno = emp.deptno;
It will give Error → Ambiguity error

→ select dept.deptno, dept.dname, dept.loc,
emp.ename, emp.job, emp.sal from
emp, dept
where dept.deptno = emp.deptno;
This will Work →

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

o/p :-

DEPTNO	SUM(SAL)
1	18000
2	17000

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

o/p :-

DNAME	SUM(SAL)
TRN	18000
EXP	17000

→ select initcap(dname), sum(sal), from emp, dept
where dept.deptno = emp.deptno
group by initcap(dname)
having .....
order by .....

Types of JOINS

1) Equijoin (also known as Natural join)
- join based on equality condition.
- shows me the matching rows of both the tables.
- Most frequently used join :(more than 90 %)

2) Inequijoin (Non-Equijoin)
- join based on inequality condition

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

It will show me all the non-matching rows of both the table.

- pratical use :- Used in Exception report

3) Outerjoin

Right Outerjoin

→ select dname, ename from emp, dept where dept.deptno = emp.deptno (+);

- (+) shows me the matching row of both table & non matching row of outer table.
- Outer table :- table which is on outer side of (+) sign.
- Outer table :- table which is on opposite side of (+) sign.

o/p :-

DNAME	ENAME
TRN	Arun
TRN	Ali
TRN	Kiran
EXP	Jack
EXP	Thomas
MKTG	

pratical use :- Used in Master-Detail Report

- Dept loop is Do-while loop
- Emp loop is for loop.

left Outerjoin

→ select dname, ename from emp, dept where dept.deptno (+) = emp.deptno ; display all the rows of emp

- Dept loop is for loop
- Emp loop is Do-while loop

a) Half Outerjoin

- '+' sign on any one side of WHERE clause.

i) Right Outerjoin

'+' sign on right side

ii) left Outerjoin

'+' sign on left side

b) full Outerjoin

shows matching rows of both the tables plus (union)
non matching rows of both the tables.

```
select dname, ename from emp, dept
where dept.deptno = emp.deptno (+)
union
select dname, ename from emp, dept
where dept.deptno (+) = emp.deptno;
```

o/p :-

DNAME	ENAME
TRN	Arun
TRN	Ali
TRN	Kiran
EXP	Jack
EXP	Thomas
MKTG	—

ANSI syntax for full Outerjoin :-

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

ANSI syntax for Right Outerjoin :-

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

ANSI syntax for left Outerjoin :-

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

- '+' ~~Outer~~ sign for writing Outerjoin is supported only in Oracle RDBMS, not supported in any RDBMS
- The ANSI syntax for right outer join & left supported by all RDBMS including oracle & MySQL but
- The ANSI syntax for full Outerjoin is supported in all RDBMS ~~including~~ except for MySQL

4) Cartesian join (Cross Join)

- join is without WHERE clause.

→ select dname, ename from emp, dept;

dept → driving table
emp → driven table

o/p :-

DNAME	ENAME
TRN	Arun
TRN	Ali
TRN	Kiran
TRN	Jack
TRN	Thomas
EXP	Arun
EXP	Ali
EXP	Kiran
EXP	Jack
EXP	Thomas
MKTG	Arun
MKTG	Ali
MKTG	Kiran
MKTG	Jack
MKTG	Thomas

[fastest join no WHERE Condition hence there is no searching involved.]

- Every row of driving table is ~~contain~~ combined with each and every row of driven table.

→ select dname, ename from emp, dept; fast
IO bet^n Server HD and Server RAM is less

→ select dname, ename from dept, emp; slow
IO bet^n Server HD and Server RAM is more

Practical use :- In the university, In Student table we have all the student name, in subject table, we have the subjectname, when we are printing the marksheet every student name is combined with each & every subjectname.

INNER JOIN

by default every join is inner join, putting a plus (+) sign is what makes it an Outerjoin.

SELF JOINING

- Join a table to itself

Used when parent column and child column both are present in the same table

→ select a.ename, b.ename from emp b, emp a where a.mgr = b.empno;

o/p:-

A.ENAME	B.ENAME
Arun	Jack
Ali	Arun
Kirun	Arun
Thomas	Jack

Server RAM

A		B	
ENAME	MGR	ENAME	MGR
Arun	4	Arun	1
Ali	1	Ali	2
Kirun	1	Kirun	3
Jack		Jack	4
Thomas	4	Thomas	5

- Self Join is the slowest join.
- Self Join is based on Recursion.

- When you give an alias to a table name a copy of table is brought to the Server RAM
- Not only will your select statement be slow but you will slow the processing speed for other users
- do not give an alias to tablename unnecessarily
- do so in the rare event of Self join.

All Joins are based on nested for loop except for the Outerjoin where one or both of the loops would be Do-While.

DEPT

DEPTNO	DNAME	LOC
1	TRN	BLY
2	EXP	DLH
3	MKTG	Cal

DEPTHEAD

DEPTNO	DHEAD
1	Arun
2	Jack

Joining 3 or more table

→ select dname, ename, dhead from emp, dept, depthead
where depthead.deptno = dept.deptno
and dept.deptno = emp.deptno ;

o/p :-

DNAME	ENAME	DHEAD
TRN	Arun	Arun
TRN	Ali	Arun
TRN	Kiran	Arun
EXP	Jack	Jack
EXP	Thomas	Jack

Types of Relationships :-

1 : 1 (Dept : Depthead) or (Depthead : Dept)

1 : Many (Dept : Emp) and (Depthead : Emp)

Many : 1 (Emp : Dept) and (Emp : Depthead)

Many : Many (Project : Emp) or (Emp : Projects)

Projects

PROJNO	CUSTNAME	DESCRIP
P1	Deloitte	CGS
P2	BNP Parilos	Marcos
P3	Morgan stanley	AMS
P4	ICICI Bank	PPS
P5	AMF	Website Dev

Projects_EMP → Intersection table.

Projno	Empno
P1	1
P1	2
P1	3
P2	1
P3	4
P3	2
P3	3

→ select custname, descrip, cname from
project_emp, emp, projects
where project.projno = project_emp.projno
and emp.empno = project_emp.empno
order by 1, 2, 3

Day 8

PAGE No.
DATE / /

Oracle 11g - SQL sub queries (Nested queries)

- Query within query (select within Select)

EMP

EMPNO	ENAME	SAL	DEPTNO	JOB	MGR
1	Arun	8000	1	M	4
2	Ali	7000	1	C	1
3	Kirun	3000	1	C	1
4	Jack	9000	2	M	.
5	Thomas	8000	2	C	4

Display the ename who is recieving min(sal)

→ select ename from emp → main query (outer)
where sal =
(select min(sal) from emp); → sub query (inner)

- main query is also known as parent query
- sub query is the child query
- max upto 255 levels for sub queries.

JOIN is faster than SUB QUERY

- When we write a join to solve the problem using one select when we use sub queries we solve the problem using 2 or more select.
- The more the number of SELECT, the slower it will be to make it work faster try to reduce the number of levels for subqueries.

PAGE No.
DATE / /

Display the second largest sal

→ select max(sal) from emp
where sal <
(select max(sal) from emp);

Display the rows who belong to the same DEPTNO as Thomas

→ select * from emp
where deptno =
(select deptno from emp
where ename = 'Thomas');

Display the rows same job as Kiran

→ select * from emp
where job =
(select job from emp
where ename = 'Kiran');

PAGE No.
DATE / /

Using subqueries with DML Command

→ delete emp
where deptno =
(select deptno from emp
where ename = 'Thomas');

→ Update emp
set sal = 10000
where job =
(select job from emp
where ename = 'Kiran');

- Above 2 command are supported by Oracle, not supported by MySQL.
- In MySQL we cannot update or delete from a table from which we are currently selecting

Solution for MySQL

→ delete emp
where deptno =
(select tempp.deptno from
(select deptno from emp
where ename = 'Thomas') as tempp);

→ update emp set sal = 10000
where job =
(select tempp.job from
(select job from emp
where ename = 'Kiran') as tempp);

ANY - logical OR
IN - logical OR

PAGE No.
DATE / /

Multi-Row sub queries

- Sub query returns multiple rows

To display the salary of Employee having salary Equal to Manager.

→
```
select * from emp
where sal = any
(select sal from emp
where job = 'm');
```

→
```
select * from emp
where sal in
(select sal from emp
where job = 'm');
```

To make it work faster;

```
select * from emp
where sal >=
(select min(sal) from emp
where job = 'm');
```

- Join is faster than sub query
- try to reduce the no of levels of subqueries
- try to reduce the no of rows returned by subquery

PAGE No.
DATE / /

Assumption 3rd row SAL is 13000.

Display the rows having salary greater than manager.

→ Select * from emp
where sal > all
(select sal from emp
where job = 'm');

ALL → perform logical AND

→ select * from emp
where sal >
(select max(sal) from emp
where job = 'm');

- SOME Operator is similar to ANY operator
- ANY is improvement over SOME Operator avoid using SOME operator, use ANY or IN Operator
- SOME Operator does not support blank padded comparison semantics
- ANY and IN operator support blank padded comparision semantics.

PAGE No.
DATE / /

Using sub-query in the HAVING clause.

DEPT

DEPTNO	DNAME	LOC
1	TRN	Bby
2	EXP	Dlh
3	MKG	Cal

EMP

EMPNO	ENAME	SAL	DEPTNO
1	Arun	8000	1
2	Ali	7000	1
3	Kirun	3000	1
4	Jack	9000	2
5	Thomas	8000	2

Display the DNAME that is having the max(sum(sal)) :-

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

DEPTNO	SUM(SAL)
1	18000
2	17000

→ select sum(sal) from emp;

→ select max(sum(sal)) from emp
group by deptno;

MAX(SUM(SAL))
18000

→ select deptno, sum(sal) from emp
group by deptno
having sum(sal) =
(select max(sum(sal)) from emp
group by deptno);

DEPTNO	SUM(SAL)
	18000

→ select dname, sum(sal) from dept, emp
where dept.deptno = emp.deptno
group by dname
having sum(sal) =
(select max(sum(sal)) from emp
group by deptno);

DNAME	SUM(SAL)
TRN	18000

PAGE No.
DATE / /

In MySQL

→ select max(sum_sal) from
(select sum(sal) sum_sal from emp
group by deptno) as tempp;

MAX(SUM_SAL)
18000

→ select deptno, sum(sal) from emp
group by deptno
having sum sal =
(select max(sum_sal) r
(select

PAGE No.
DATE / /

Correlated Subquery (using the EXISTS Operator)

→ Select deptno from emp;

o/p
1
1
1
2
2

→ select distinct ~~from~~ deptno from emp;

o/p
1
2

→ select dname from dept
where deptno in
(select distinct deptno from emp);

o/p TRN
EXP

→ select dname from dept
where deptno not in
(select distinct deptno from emp);

o/p MKTG

→ select dname from emp, dept
where dept.deptno = emp.deptno;

o/p
TRN
TRN
TRN
EXP
EXP

→ select distinct dname from emp, dept where dept.deptno = emp.deptno;

TRN

EXP

- Join is faster than sub query but if we have a join alongwith DISTINCT, to make it work faster use Correlated sub query (use the EXIST operator)

→ select dname from dept where exists (select deptno from emp where dept.deptno = emp.deptno);

o/p:- TRN

EXP

Explaination:-

- first main query is executed
- for every row returned by main query Oracle will run the sub query once
- sub query returns boolean TRUE or FALSE value back ~~bac~~ to main query
- If sub-query return TRUE value then main ~~main~~ query is eventually executed for that row
- if sub query return FALSE then the main query is not executed for that row
- Unlike earlier we do not use DISTINCT here, this speeds it up
- Unlike earlier the no of full table scan is reduced this futher speeds it up.

PAGE No.
DATE / /

→ select dname from dept where not exists
(select deptno from emp
where dept.deptno = emp.deptno);

o/p:- MKTG

Oracle 11g - SET Operators

- based on set theory.

Dr Codd - founder of RDBMS (1968)

EMP1

EMPNO	ENAME
1	A
2	B
→ 3	C

EMP2

EMPNO	ENAME
1	A
2	B
→ 4	D
5	E ←

DATA - intersection of row and column.

Dr. Codd Ideas → Concept of table, datatype, SQL, system tables, insert, update, delete, constraints, views, client server architecture, distributed databases, frontend software, laptop, pointing device etc.

→ select empno, ename from emp1
union
select empno, ename from emp2

o/p:-

EMPNO	ENAME
1	A
2	B
3	C
4	D
5	E

Union :- Will combine the output of both the SELECT and it will suppress the duplicates.

→ select empno, ename from emp1
union
select empno, ename from emp2
order by 1;

→ Select empno, ename from emp1
union all
select empno, ename from emp2
order by 1;

Union all :- Will Combine the output of both the SELECT and duplicates are not suppressed.

PAGE No.
DATE / /

o/p

EMPNO	ENAME
1	A
1	A
2	B
2	B
3	C
4	D
5	E

→ select empno#, ename from emp1
intersect
Select empno#, ename from emp
order by 1

o/p

EMPNO	ENAME
1	A
2	B

INTERSECT :- Will return what is common in both the select and duplicate are suppressed.

→ select empno, ename from emp1
minus
select empno, ename from emp2
order by 1;

o/p :- EMPNO | ENAME
3 C

MINUS :- What is present in the first SELECT and not present in the second SELECT and the duplicates are suppressed.

Max upper limit is 255.

select .....
union
(select .....
minus
select .....) → preference
union
(select .....
intersect
select .....)
union all
select .....
order by x ;

cc

PAGE No.
DATE / /

→ select job from emp where deptno = 10
intersect
select job from emp where deptno = 20;

o/p JOB
CLERK
MANAGER

→ select job from emp where deptno = 10
minus
select job from emp where deptno = 20;

o/p :- JOB
PRESIDENT

- Union, union all → supported by MySQL
  intersect, minus → not supported by MySQL
- Using set operator with multiple SELECT brackets for precedence cannot be specified in MySQL

Day 9

PAGE No.
DATE / /

Pseudo-Columns

- fake column
= Virtual Column

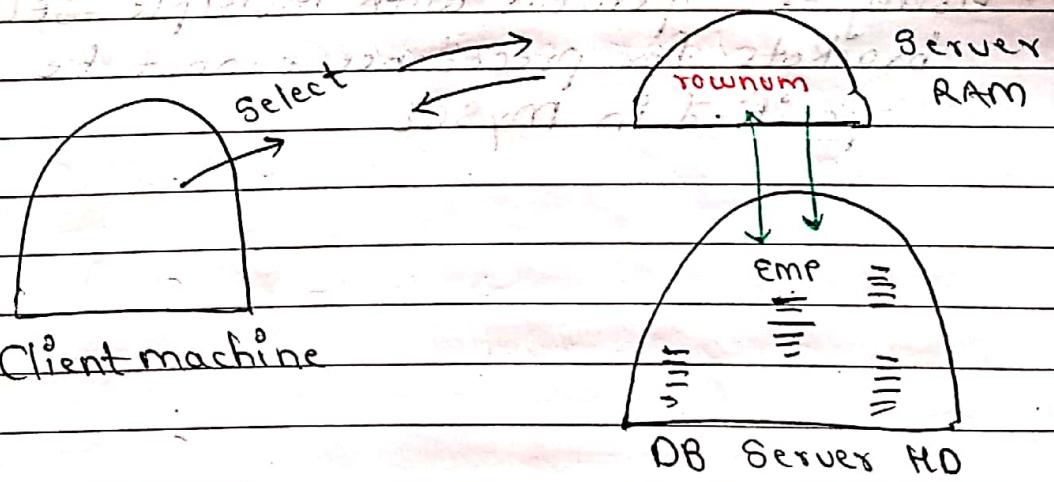
eg :- 1) Computed columns (ANNUAL = sal * 12)
2) Expression (NET_EARNING) = sal + comm - tax
3) function based column sum(sal)
initcap(ename).

Oracle supplied pseudo columns :-

→ select ename, sal from emp;

ROWNUM = row number

→ select rownum, ename, sal from emp;



PAGE No.
DATE / /

→ Select rownum, ename, sal from emp where rownum = 1 ;

→ select rownum, ename, sal from emp where rownum < 4 ;

→ Select rownum, ename, sal from emp where rownum = 4 ;

this will not work

→ select rownum, ename, sal from emp ~~where~~ order by ename ;

it will show ename by order but the rownum will not be in sequence.

→ select rownum, ename, sal from (select ename, sal from emp order by ename);

INLINE VIEW → If you use subquery with ~~form~~ from clause, it is known as inline view.

→ select rowid, ename, sal from emp;

- rowid is the row address
- rowid is the actual physical memory location in the DB Server HD where that row is stored.
- rowid is encrypted fixed length string of 18 character.

- no two rows of any table in the entire database can have the same Rowid.
- rowid works as an unique identifier for every row in the database.
- When we SELECT from a table of rows in the output will be in the ascending order of Rowid.
- If we update a row, the rowid may change & hence the rowid will change.

o/p :-

ROWID	ENAME	SAL
AAASNVAAEAAAATA	KING	5000

18 character

- We can use Rowid to UPDATE or DELETE the duplicate rows in a table.

Rowid is used internally by Oracle :-

- Rowid is used internally by Oracle to distinguish between two rows in the database.
- To implement row locking
- To manage indexes
- To manage the cursors (PL/SQL)
- Row management

PAGE No.
DATE / /

- Rowid is available in Oracle and we can view it.
- Rowid is available in MySQL but we cannot view it.
- feature of rownum is available in Oracle and we can view it.
- feature of rownum is not available in MySQL

Oracle 11g - SQL * Plus

- product of Oracle Corporation.
  - earlier known as UFI (User friendly Interface)
- earlier 3rd party product.
- Oracle v4 onwards, we have SQL*Plus
- SQL*Plus → Extension to SQL
- It is use to remove the limitations of SQL
- used as a reporting tool. (only text reports)
- used for setting the Oracle environment
  eg :- linesize, pagesize etc
- Oracle client software
- Interface with database for running SQL and SQL*Plus commands and PL/SQL programs.

eg :-

password	describe	clear screen
define_editor	L	I
A	C	del
ed	save	~~del~~ get
@	exit	~~get~~ prompt
/	run	host
quit	spool	

first few letters is sufficient
; is optional (enter key is end of command)

3 ways of ending SQL command

→ select * from emp ;

→ select * from emp
/

→ select * from emp
→ / 2 times enter key

→ desc - enter key
emp → table name

SQL	SQL*Plus
; compulsory	; optional
no shorcuts	first few letter are sufficient
ANSI & ISO standard	Oracle standard
Common for all RDBMS.	will work only with oracle RDBMS
last command is stored in the buffer	not stored in buffer

→ Help Index
shows all the help command

→ Help del

→ set sqlprompt = 'VITA>'
VITA>
VITA>
VITA>
It is for current session

→ select deptno, job, ename, sal,
hiredate from emp;
order by deptno, job;

→ set feedback off default is 6
It will give the clean output
& remove all the statement
in the output
by default it is ON

→ set feedback 2

→ set pagesize
no of lines per page

• default value of pagesize
is 14

→ set pagesize 25
for desktop

→ set linesize 40
40 character per line.

→ set wrap off
It will truncate the character all after the 40.

→ set linesize 60

- default value of linesize is 80.

→ set pause on
It will pause the output & execute after enter key. (pause acc to pagesize)

→ set pause 'text'
we can pass message also.

→ set pause 'Press enter to see more employees...!'

→ show pagesize
→ show linesize
→ show pause
→ show all

→ break on deptno
to suppress the deptno

→ break on deptno skip 1
skip a line on ~~every dept~~
after every dept no

→ break on deptno skip page
skip a page on every deptno

→ break on deptno dup skip page
It will show duplicate

→ break on deptno nodup skip page
It will show no duplicate

→ break

→ clear breaks

→ break on deptno on job

→ break on deptno skip page on job skip 1

→ break on deptno nodup skip page on job dup skip 1

PAGE No.
DATE / /

To display sum of sal dept wise

→ break on deptno skip page
→ compute sum of sal on deptno

→ break on deptno skip page
compute sum label 'Deptot' of sal on deptno

- for compute command to work, BREAK command is compulsory

→ clear breaks
→ clear computes

→ break on report
compute sum of sal on report
for grand total.

→ break on report
compute avg min max sum on report
It will show all the values

→ break on report on deptno on job
→ compute sum of sal label 'Jtot' on job
→ compute sum of sal label 'Dtot' on deptno
→ compute sum of sal label 'Gtot' on report

→ set time on
this will display the time next to SQL command.

→ set time off
by default it is off.

→ set timing on
It shows the execution time.

→ column sal heading 'SALARY'
It will affect all the table all the select having sal column It will be headed by SALARY.

→ column sal off
It will be switched off we can switch on after sometime when needed.

→ column sal heading 'SALARY'
format 99,999.99
the output of sal will be in this format.

→ column sal heading 'SALARY'
format $ 99,999.99
symbol ~~use~~ ~~use~~ can be use

PAGE No.
DATE / /

→ select 'RS' || "SALARY" from emp;

→ column ename heading 'NAME OF THE | EMPLOYEE'

for multiline column name

→ column ename heading 'NAME OF THE | EMPLOYEE' format a15

It will increase the length of the ename (varchar(15))

→ column sal off

→ column ename off

→ column sal on

→ column ename on

→ clear column

format & heading both will go

→ ttitle center 'CDAC VITA'

left

right

Center

for page heading

→ ttitle center 'CDAC VITA' skip 2

System Variables

SQL.PNO — page no
SQL.LNO — line no
SQL.RELEASE — version of oracle
SQL.SQLCODE — error number
SQL.USER — oracle user name

→ ttitle off
→ btitle off

→ set head off
column heading will not come, only data will come

→ set head on

→ spool abc
→ select ...
→ select ...

→ spool off

→ ed abc.lst — open file in notepad

→ Spool out
Stops the spooling & sends output to printer

→ pause text
to pause the processing

→ pause Beginning of Emp report

→ set autocommit on
→ set autocommit off
→ show autocommit

→ pause Beginning Dept Report
select * from Dept;
pause End of Dept Report
pause Beginning of Emp Report
select * from emp;
pause End of Emp Report

try this

→ disconnect

→ connect

→ termout off
to suppress screen output during spooling.

→ set echo on

It will display the command as it execute

→ set echo off

suppress display of SQL command.

Important Command

→ select * from emp
where deptno = &x;

'&' → It is use to take the input from user.

& works before compilation, not during runtime. (pre-processing)

→ set verify off

to suppress old & new message from coming.

→ select * from emp
where deptno = &deptnumber;

o/p:- Enter value for deptnumber :-

PAGE No.
DATE / /

→ Select * from emp
where deptno = &deptnumber and job =
&designation ;

o/p :- Enter value for deptnumber :- 10
Enter value for designation :- 'CLERK'

→ select * from emp
where deptno = '&deptnumber and job =
'&designation' ;

o/p :- Enter value for deptnumber :- 10
Enter value for designation :- CLERK

→ insert into emp
values (&empno, upper('&ename'), &sal);

→ accept variablename [datatype] prompt
['text']
datatype can be number, char, date

→ accept y number prompt 'Please
enter the dept number.'
The message will be the input
for user.

define — It will show all defined
define y variable in ram

undefine y — It will remove the
variable y from
RAM

PAGE No.
DATE / /

Create a file :- empreport.sql

```
cl scr
prompt Welcome to VITA
prompt Please choose a deptno
prompt for eg:- 10, 20, 30
accept y number prompt 'Dept number
ttitle center 'Employee of Dept
                number' y skip 4
break on report
Compute sum of sal on report
cl scr
spool abc
select * from emp where deptno = &y
spool out
pause End of deptReport
undefine y
```

Assignment :- Create a matrix report
file :- matrixreport.sql
Display sum of sal → Dname wise & jobwise.
Assume existence and hardcode
ANALYST, CLERK, MANAGER, PRESIDENT, SALESMAN in your SELECT statement
Display 0 where no value
Hint :- Use Decode.

o/p :-

	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN
TRAINING	3000	0	3000	5000	0
EXPORT	4000	3000	4000	0	0
MARKETING	0	4500	5000	0	2000
	7000	7500	12000	5000	2000

Day 1

PAGE No.
DATE / /

Oracle 11g - PL - SQL

- Procedural language SQL
- Programming language of Oracle
- Used for database Programming
  eg :- HRA_CALC, TAX_CALC, ATTENDANCE_CALC etc.
- used for processing (server side data processing)
- PL-SQL is the #1 database programming language in the world. (63% of database programming).
- PL-SQL is the #2 programming language in the world.
- can be called in SQL*Plus, Oracle forms, Oracle Reports, Oracle menus, Oracle Graphics etc.
- Can be called through any front-end software (only the drivers are ~~request~~ required for database connectivity)
- 4 GL (support OOPS)
  - Program is case insensitive.

1st PL/SQL Program

PL/SQL block

```
BEGIN
    INSERT INTO DEPT VALUES (1, 'a', 'B');
END;
/
```

→ It indicates the end of PL/SQL program.
It is compulsory for every program.
It has to be first character on last line

- PL/SQL program allows you to put SQL commands inside PLSQL program (hence the name PL/SQL)
(SQL with a procedural option

```
BEGIN
   ==
   BEGIN
   ==
   END;
   ==
END;
/
```

execution is top to bottom

- Block level language (because we can have block within block)
- Benifits of block level language
  1) Modularity
     bigger program can be cut down into small pieces
  2) Control Scope of variable
     (for data hiding → encapsulation
  3) Efficient Error management
     (with the help of Exception we can localize the error)
- Screen input and screen output is not allowed (used only for processing

PAGE No.
DATE / /

```
BEGIN
    SELECT * from EMP;
END
```

Not allowed.

- SQL command that are allowed inside PL/SQL :-

insert, update, delete, merge, upsert, rollback, commit, savepoint, select

We can select statement as a part of subquery

```
delete emp
    where deptno =
    (select deptno from emp
     where ename = 'Tom');
```

- DDL command not allowed inside PL/SQL
- DCL command not allowed inside PL/SQL
- SQL * Plus not allowed inside PL/SQL

To store output of PL/SQL program :-

```
create table tempp
(
  fir number(4)
  sec char(15)
);
```

- Create a seperate output table on a per-application basis

Program

→ Begin
insert into tempp value(1,'Hello');
End;
COMMIT;
/

o/p :- TEMPP
FIR SEC
1 Hello

SQL> ed filename — save everytime in a file

SQL> ed P1 — read, compile, plan &

SQL> @ P1 — execute

After execution;
"PL/SQL procedure Successfully completed"

SQL> select * from tempp;

SQL> set head off

o/p 1 Hello

PAGE No.
DATE / /

Program

→ Declare
x number(4); local variable

Begin
x := 10;
insert into tempp values (x, 'Hello');
End;
/

Variable declaration has to done between DECLARE and BEGIN.

In PL/SQL when you declare a variable if you don't initialize it then it store NULL value.

:= is the assignment operator.

o/p:-

TEMP

FIR	SEC
10	Hello

Can declare a variable & assign a value to it at same time.

→ Declare
x number(4) := 10; allowed.

PAGE No.
DATE / /

Program

→ Declare
x number(4);

Begin
x := &x;
insert into tempp values
(x, 'Hello');
End;

1) &x → user input
2) It will replace &x with user input

PAGE No.
DATE / /

Program

```
truncate
clear
prompt Namaskar. Welcome to VITA.
accept y number prompt 'please enter
    dept number of your choice:'
Declare
    x number(4);
Begin
    x:= &y;
    insert into tempp values (x, 'Hello');
End;
/
select * from tempp;
prompt Thankyou. Visit again.
set verify off.
```

Compiler present in SQL prompt

SQL Compiler	Javascript
SQL * plus	JSON
PL/SQL	BSON
C	COBOL interpreter
C++	
Java (JVM)	
FORTRAN	
PASCAL	
ADA	

Program.

→ Declare
x char(15);
Begin
x := &name;
insert into tempp values(1, x);
End ;
/

o/p :-
Enter value for name := 'Jack'

1) for char, varchar2 ; date
use ' '

2) Begin
x := '&name'; allowed
no need of giving ' ' at o/p.

3) In Oracle number to char is implicit conversion (automatic datatype conversion)

→ Enter value for name : 10
allowed

PAGE No.
DATE / /

Program

→ Declare
x number(4);
Begin
x := '90';
End;
/

allowed.

Program

→ Declare
x constant number(4) := 10;
Begin
insert into tempp values(x, 'Hello');
End;
/

1) We cannot change the value of constant afterwards.
If you're creating a constant you will have to declare it and assign a value simultaneously

→ Begin
x := 20;

error

→ Declare

Begin

-- Single line Comment

/* Multi-line
Comment */

End;
/

1) 1 comment minimum every 2 statement
2) Do not use & inside the comment cause & operate / works before compilation.

Program

```
Declare
   x char(15);
   y number(6,2);
   z constant number(2,1) := 0.4;
   hra number(6,2);

Begin
   x := upper('&ename');      → KING
   y := &salary;              → 3000
   hra := y * z;
```

insert into tempp values (y, x);
insert into tempp values ('hra', 'HRA');
End;
/

o/p:-

TEMP	
FIR	SEC
3000	KING
1200	HRA

dbms_output.put_line ('text');
only works in SQL*PLUS.
Package name . procedurename.

1) dbms_output is package.
2) It helps - PL/SQL interactive

→ dbms_output.put_line ('ename = Mr. '||x);
dbms_output.put_line ('Salary = RS. '||y);
dbms_output.put_line ('HRA = RS. '||hra);

→ dbms_output.put_line ('Program run on '||
to_char (sysdate, 'DD/mm/YYYY HH:MIam)

→ to make this work
SQL > set serveroutput on

PAGE No.
DATE / /

Table

Emp

ENAME	SAL	JOB	DEPTNO
SCOTT	3000	CLERK	10
KING	5000	MANAGER	20

Program

```
Declare
    x number(4)
Begin
    select sal into x from emp
    where ename = 'KING';
    /* processing, eg: hra := x * 0.4, etc */
    insert into tempp values(x, 'KING');
End;
/
```

1) Select sal into x (5000)

2) select sal into x from emp
where ename = '&ename';

3) Ensure that king is present in the table & ensure that only one king is present.

4) x range should be according to the sal range.

PAGE No.
DATE / /

Program

```
Declare
    x number(4);
    y char(15);

Begin
    select sal, job into x, y from emp
    where ename = 'KING';

/* processing, eg: hra := 0.4*x, lower(y) */

    insert into tempp values (x, y);
End;
/
```

1) select sal, job into x, y

2) Declare
    x emp.sal%type;
    y emp.job%type;

'x' variable will have the same datatype and width as EMP table SAL Column

'y' variable will have the same datatype and width as EMP table JOB Column

In future even if you ALTER the structure of EMP table, the program will still work.

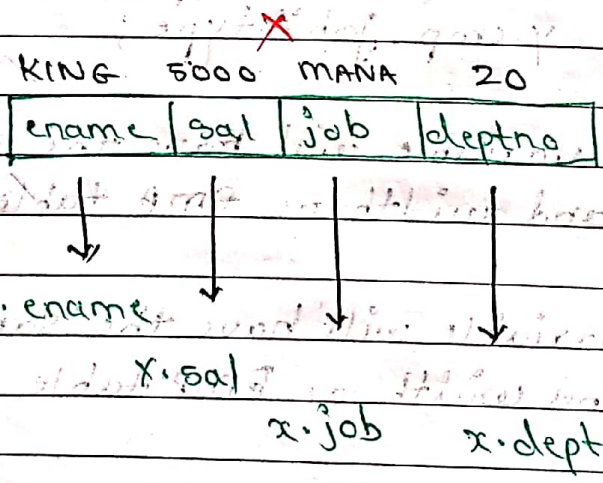
PAGE No.
DATE / /

3) Declare
x emp%rowtype ;
~~y emp%~~.

'x' will have the same datatype and width ~~as~~ for the row.

Program

```
Declare
    x emp%rowtype
Begin
    select * into from emp
    where ename = 'KING';
insert into tempp values (x.sal, x.job);
End;
/
```



o/p TEMPP

FIR	SEC
5000	MANAGER

x emp%rowtype ;

'x' become a structure type of variable similar to 'C' programming

Benefits :-

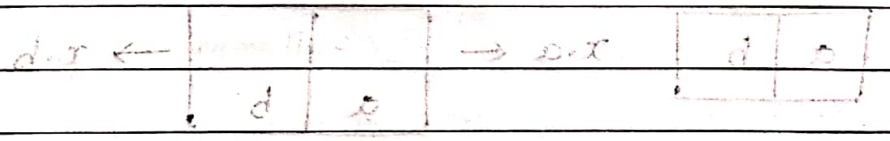
1) In future even if you alter the structure of EMP table the program will still work.

2) Multiple related values can be combined and stored in a single variable (group together the related fields)

3) If you want to copy the data from one place to another you can do it in a single command.

```
Declare
    x emp%rowtype;
    y emp%rowtype;
Begin
    y := x                    structure
```

4) y.sal := x.sal; for single column
   y := x ; recommended.



PAGE No.
DATE / /

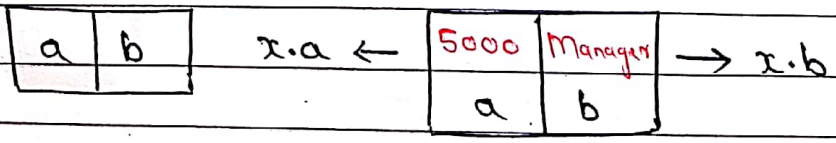
User defined Structure

Program

```
Declare
    type pqr is record          creation of datatype
    (
        a emp.sal%type,
        b emp.job%type
    );
    x pqr;
Begin
    select sal, job into x from emp
    where ename = 'KING';
    insert into tempp values(x.a, x.b);
End;
/
```

1) Similar to creating structure in 'C' programming
2) then you say some variable is of type of that structure.

PQR x



Server RAM

PAGE No.
DATE / /

3) Structure is known as PL/SQL ~~comm~~ record

Benefits:-

1) Multiple related values can be stored in a single variable
eg. ADDR_STRUCT could be made up STREET, CITY, STATE and PINCODE therefore entire address information can stored in a single variable.

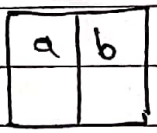
2) all address information from 'x' variable can be copied into 'y' variable using a single command.
eg:- y := x;

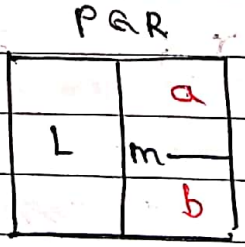
```
Declare
    type addr_strict is record
        (
        ------
        ------
        );
        x addr_struct;
        y addr_struct;
Begin
        y := x;
```

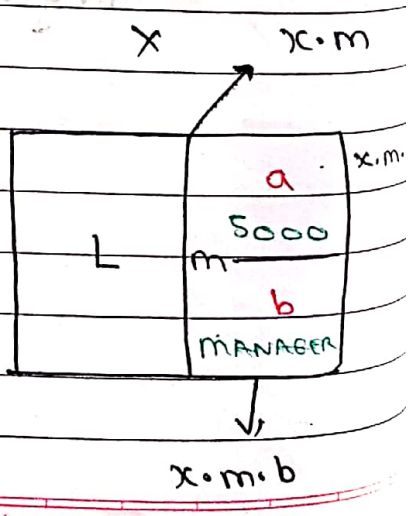
#Program (Structure within Structure)

```
Declare
    type abc is record        -- creation of
    (                            datatype
      a emp.sal%type;
      b emp.job%type
    );
    type pqr is record        -- creation of
    (                            datatype
      l number(4),
      m abc
    );
    x pqr;

Begin
    select sal, job into x.m from emp
    where ename = 'KING';
    insert into tempp values
        (x.m.a, x.m.b);
End;
/
```







1) Reusability of Code
eg:- ADDR_STRUCT can be nested in EMP_STRUCT, CUST_STRUCT, SUPP_STRUCT, STUDENT_STRUCT, etc.

2) If you want to copy ~~emp~~ STUDENT address into EMP address you can do it in a single command.

3) Standardization of Code
address information of Employees, Customers, etc, would be similar.

Day 3

PAGE No.
DATE / /

EMP

ENAME	SAL
KING	9000

TEMPP

FIR	SEC

Decision making using if statement

```
Declare
    x number(4);
Begin
    select sal into x from emp where
    ename = 'KING';
     if x > 5000 then
       insert into tempp values (x, 'Highsal');
   end if;
End;
/
```

1)
```
if x > 5000 then
    insert into tempp values (x, 'highsal');
else
    insert into tempp values (x, 'lowsal');
end if;
```

2) Nested if
```
if x > 5000 then
    insert into tempp values (x, 'highsal');
    else
if x < 5000 then
        insert into temp values (x, 'low sal');
        else
    insert into temp values (x, 'Medium Sal');
end if;
```

3) if x > 5000 then
insert into tempp values(x, 'highsal');
elsif x < 5000 then
insert into tempp values (x, 'lowsal');
else
insert into temp values (x, 'mediumsal');
end if;
End;
/

elsif

4) format

if ------ then
elsif ------ then
----
elsif ------ then
----
elsif ------ then
else
----
end if;

#Program

→ Declare
x number(4);
y char(15);

Begin
select sal,
decode(sign(sal-5000), 1, 'high sal',
-1 'low sal', 'medium sal')
into x, y from emp where ename =
'KING';
insert into tempp values(x, y);
End;
/

1) Decode is faster than if statement.
2) Decode is a single row function but it require a table to select from.

Program
check if number is divisible by 7

Declare
x number(4),
y char(15);
Begin
x := #

```
Select decode (mod (x,7), 0, 'Divisible',
        Not Divisible')
    into x, y from dual;
    insert into temp values (x, y);
End;
/
```

Program

```
Declare
    x boolean;
Begin
    x := TRUE;
    if x then
insert into tempp values (1, 'mumbai');
    end if;
End;
/
```

o/p :- 1 Mumbai

1)
```
Begin
    x := TRUE; false;
    if not x then
        insert into temp values (1, Delhi');
    end if;
```

2) Boolean is a logical datatype.
Boolean is datatype of PL/SQL only
You cannot have a boolean column
in a oracle table.

4) If x is null

```
If x is null then
elsif x then
-------
else
-------
end if;
```

PL/SQL LOOPS

for repetitive / iterative processing

1) While loop:
always check for some condition before entering the loop.

Program

```
Declare
    x number(4) := 1;
Begin
    while x<=10
    loop                    start of loop
insert into temp values (x, 'in while loop');
    x := x+1;
end loop;                   End of loop
End;
/
```

i) x++, ++x, x--, --x are not allowed in PL/SQL

#Program.

```
Declare
    x number(4) := 1
    y number(4) := 1;

Begin
    while x<10
    loop
        while y<10          / y<x
        loop
            insert into tempp values(y, 'in y loop');
            y := y+1;
        end loop;
        insert into temp values (x, 'in x loop');
        x := x+1;
    end loop;
End;
/
```

try it out

Nested loop

o/p:-

```
in y loop
.
.
.
in x loop
.
.
.
```

18 rows

PAGE No.
DATE / /

Program

```
Declare
    x number(4) := 1;
    y number(4);

Begin
    while x<=10
    loop

        y := &sal;
        insert into tempp values
                (y, 'in x loop');
        x := x+1;
    end loop;
End;
```

1) & works before compilation
2) PL/SQL is not interactive

o/p :- 3000 in x loop

9 times

PAGE No.
DATE / /

Program

```
Declare
    x number(4) := 1;
Begin
    while x < 10
    loop
        insert into tempp values (x, 'x loop');
        if x > 5 then
            exit;
        end if;
        x := x+1;
    end loop;
End;
/
```

o/p :- 1 x loop
2 x loop
3 x loop
4 x loop
5 x loop
6 x loop

1) EXIT will exit the current loop
2) Similar to break statement in C
3) insert into temp . . .
   exit when x > 5;
   x := x+1;

same o/p

PAGE No.
DATE / /

Do - While loop

1) There is no condition to enter the loop but there is a condition to exit the loop

#Program

```
Declare
        x number(4) := 1;
Begin
        loop
        insert into temp values (x, 'x loop');
        exit when x>5;
        x := x+1;
    end loop;
End;
/
```

2) It will execute at least once

3) used for master-detail report (Outerjoin).

PAGE No.
DATE / /

For loop

1) Automatic incrementation of variable
2) for convience of programmer

#Program

```
Declare
    x number(4) := 1;
Begin
    for x in 1..10
    loop
        Insert into tempp values(x, 'in for loop');
    end loop;
End;
/
```

3) difference always 1
4) always increments by 1
5) for loop variable is the readonly variable.
6) for loop variable cannot be initialized inside the loop.

```
x := x + 2;    error
```

7) for loop variable need not be declared
8) for loop variable is created when enters the for loop & killed when it comes out of for loop
9) for loop variable is local variable & scope is limited to the forloop.

3) Begin
for x in &x .. &y
loop
Can also give blank space followed by 2 dots.

#Program

```
Declare
    x number(4) := 100;
Begin
    for x in reverse 1..10
    loop
    insert into tempp values (x, 'in for loop');
end loop;
end;
/
```

1) has to be from lower value to higher value.

o/p :- 10 in for loop
9 in for loop
.
.
.

2) Begin
for x in 10..1 It will not show any output
loop

PAGE No.
DATE / /

Program

```
Declare
    x number(4) := 100;
begin
    for x in 1.5 .. 9.3
    loop
     insert into tempp values (x, 'in for..');
end loop;
end;
/
```

1) In Oracle when you assign a float value to integer variable, then automatic rounding takes place.
2) float to int → Implicit conversion.

```
o/p    2  in for loop
       3  in for loop
       .
       .
       9  in for loop
```

#Program

```
Declare
    x number(4) := 100;
Begin
    for x in 1..10
    loop
        for y in 1..x
        loop
            insert into temp values
                (y, 'y for loop');
        end loop;
    end loop;
End;
/
```

o/p

```
1   y for loop
1   y for loop
2   y for loop
1   y for loop
2   y for loop
3   y for loop
:
:
10  y for loop.
```

PAGE No.
DATE / /

PL/SQL Goto Statement

used for transfer of control

Program

```
Declare
    x number(4) := 1;
Begin
    insert into tempp values (x, 'A');
    goto abc;
    <<pqr>>                                  label
    insert into tempp values (x, 'C');
    x := x+1;
    <<abc>>                                  label
    insert into tempp values (x, 'B');
```

outermost loop, Goto is the fastest way of doing it.

3) To exit from an If construct, Goto is the only way of doing it.

```
If ... then
    goto abc ;
END IF;
<<abc>>
```

4) To imitate the working of 'continue' statement of 'C' programming.

```
for x in 1..100
loop
    if mod(x,10) = 0 then
        goto pqr ;
    end if;
    ----------;
    ----------;
    ----------;
    <<pqr>>
    null;
end loop;
```

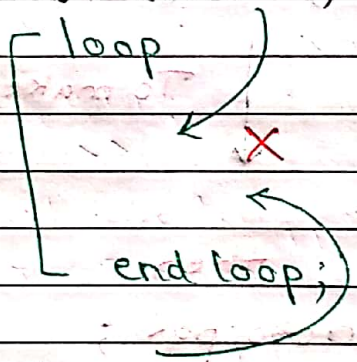
5) To exit the block.

```
Declare
Begin
```

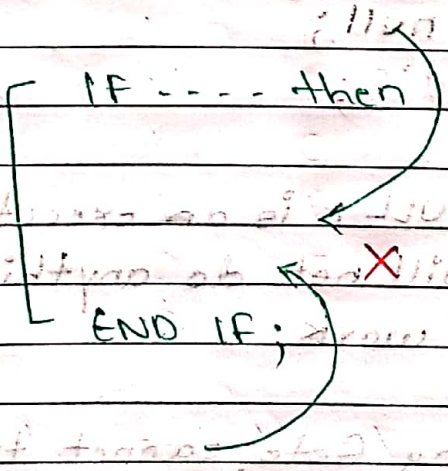
```
goto abc;
<<abc>>
End; null;
```

6) Rules for Goto Statement

1) Cannot transfer control from outside to a point within a loop (Reverse allowed)



2) Using Goto you cannot transfer control from outside to a point within an If ~~s~~ contruct (Reverse allowed)



3) Goto should always transfer control to some executable statement
(End;, End if;, End loop; are non executable statements)

```
If ... then
    goto pqr;
    ...
<<pqr>>          ← Error
End if;
```

To make this work

```
If ... then
    goto pqr;
    ...
<<pqr>>
null;
End if;
```

NULL; is an executable statement which will not do anything hence it will work.

4) Using Goto, cannot transfer control from a ~~point~~ main block to a point within a sub block (Reverse allowed)

Oracle 11g SQL - Indexes.

PAGE No.
DATE / /

EMP

RowID	EMPNO	ENAME	SAL	DEPTNO
X001	5	A	5000	1
X002	4	A	6000	1
X003	1	C	7000	1
X004	2	D	9000	2
X005	3	E	8000	2

1) To Speed up the search operations (for faster access)
2) To speed up SELECT statement with a WHERE clause.

→ Select * from emp
where empno = 1;

3) Index data is stored in DB server HD
4) Index will occupy space; but we are ~~connecte~~ concerned about performance.
5) Duplicate values are stored in Index
6) Null values are not stored in an Index

→ select * from emp
where empno is null;

It will do the full table scan & it will be slow.

Pessimistic Querying → searching for null value.

7) If you want to make this work faster for null value, store '0' instead of null then

→ select * from emp
where empid = 0 ;

8) Indexes are available in all DBMS, RDBMS, and some language also.

In Other RDBMS,

you need to manually invoke the index

→ use index ind_empno ;

9) In Oracle & MySQL the indexes are automatically invoked by the system as and when required.

Execution Plan :-

Plan created by oracle as to how it is going to execute the select statement.
(Oracle query optimizer tool is required)
Very Expensive

→ explain plan for select * from emp
where empno = 1 ;

In Other RDBMS

SQL> insert/update/delete .....
SQL> REINDEX

It has to reindex manually everytime we insert/update or delete.

10) In Oracle and MySQL, the indexes are automatically updated by the system for all the DML operations.

11) Oracle and MySQL are self-managing RDBMS

→ select * from emp
where ename = 'A';

IND_ENAME

ROWID	ENAME
X001	A
X002	A
X003	C
X004	D
X005	E

→ select * from emp
where sal > 7000;

IND_SAL

ROWID	SAL
X001	5000
X002	6000
X003	7000
X004	8000
X005	9000

12) No upper limit on the number of indexes per table.

13) larger the number of indexes, the slower would be the DML Operations.

14) In Oracle, you cannot long and Raw columns (Index)

15) In MySQL you cannot Text & Blob columns. (Index)

EMP

ROWID	EMPNO	ENAME	SAL	DEPTNO
X001	1	A	5000	1
X002	2	A	6000	1
X003	3	C	7000	1
X004	4	D	9000	2
X005	5	E	8000	2

→ select * from emp
where empno = 2;

→ select * from emp
where sal > 5000;

IND_EMPNO

ROWID	EMPNO
X004	5000
X002	6000
X003	7000
X005	8000
X004	9000

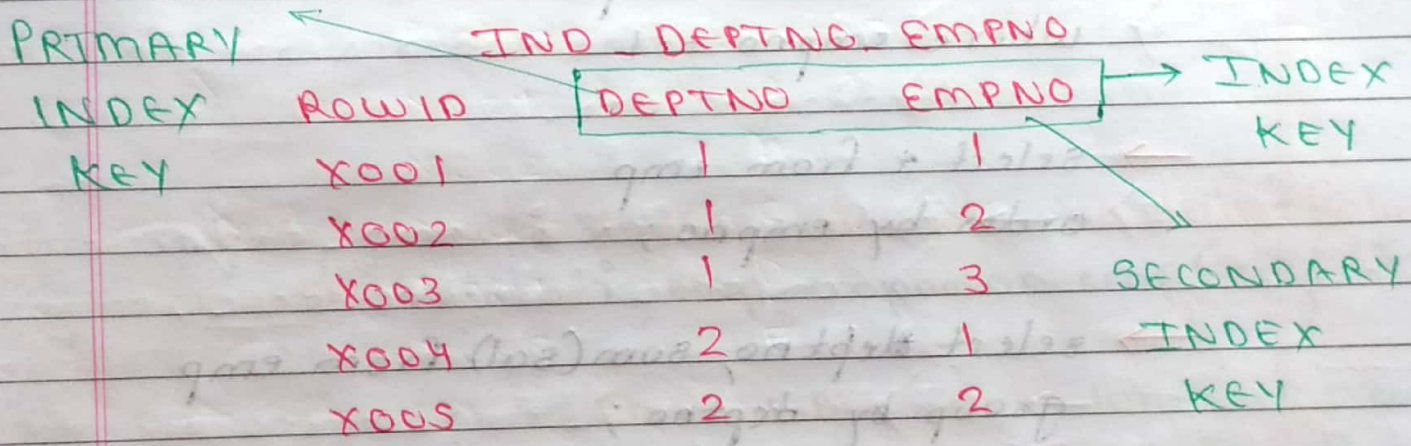
IND_SAL

ROWID	SAL
X001	1
X002	2
X003	3
X004	4
X005	5

Select * from emp
where empno = 2 and sal > 5000;

16) If you have multiple INDEPENDENT column in the WHERE clause, then create seperate indexes for each column; Oracle will use the necessary indexes as and when required.

COMPOSITE INDEX :- Combine 2 or more inter-dependent column in a single index.




```
select * from emp
where deptno = 1 and empno = 1;
```

INDEX KEY :- Column or set of column on whose basis the index has been created.

17) In Oracle you can combine upto 16 column in a composite Index
In MySQL you can combine upto 32 column in a Composite Index.

Condition when an Index should be created:

1) If you have SELECT statement with a WHERE clause, ~~OR~~ ORDER BY clause, GROUP BY clause, DISTINCT, UNIQUE, UNION/ INTERSECT / MINUS.

→ select * from emp
where empno = 1

→ select * from emp
order by empno;

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

→ select distinct empno from emp;

2) If your select statement retrieves < 25% of table data.

3) Primary key and Unique columns should always be indexed.

4) Common column in join operation should always be indexed.

~~select~~ .

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

I2

RowID	DEPNO
X001	1
X002	1
X003	1
X004	2
X005	2

I1

ROWID	DEPTNO
Y011	1
Y012	2
Y013	3

SQL> Create index indexname on table(column);
→ create i_emp_empno on emp(empno);

I_EMP_EMPNO

ROWID	EMPNO
X001	1
X002	2
X003	3
X004	4
X005	5

When we create a index, there is a entry added ~~on~~ in the system table.

→ select * from emp
where empno = 1;

→ create index i_emp_ename on emp(ename);

→ create index i_emp_sal on emp(sal);

→ create index i_emp_deptno_empno on
emp(deptno, empno);

Composite Index

PAGE No.
DATE

To Drop index

→ drop index i_emp_empno;

"If you drop the column/table then the associated indexes are dropped automatically."

1) By default all the indexes are in ascending order.

2) → create index i_emp_deptno_empno on emp (deptno desc, empno);

→ create index i_emp_empno on emp (empno desc)

→ create index i_emp_deptno_empno on emp (deptno desc, empno desc);

→ create index i_orders_onum on orders (onum desc);

→ select * from orders where onum = 1000000;

→ select * from user_indexes where table_name = 'EMP'; Case Sensitive

→ select * from user_ind_columns
where table_name = 'EMP';

→ create unique index i_emp_empno on emp
(empno);

perform an extra function, it won't allow the user to INSERT duplicate value for EMPNO.

If we enter the duplicate value it will raise an Exception (Insert/update)
DUP_VAL_ON_INDEX

3) Oracle will not allow you to create more than 1 index on the same column
(unless it is combined with other columns)

Types of Indexes :-

1) Normal Index.
2) Unique Index.
3) Clustered Index.
4) etc.

Oracle 11g - SQL - ALTER TABLE

1) It is a DDL command.
2) rename a table
3) drop a column
4) increase width of column.

Indirectly :-

1) reduce the width of column
2) change datatype of column
3) Copy rows from one table into another table
4) Copy a table
5) Copy structure of table
6) rename a column
7) change position of column in a table structure.
   (for storage consideration because of null values)

→ select * from emp; - Not Recommended

→ select empno, ename, sal from emp;
Recommended

Emp

EMPNO	ENAME	SAL
1	SCOTT	5000
2	KING	6000

→ insert into emp values (1, 'SCOTT', 5000);
Not recommended.

→ insert into emp (empno, ename, sal)
values (1, 'SCOTT', 5000) recommended

In Oracle :-
rename emp to employees;

In MySQL :-
rename table emp to employees;

→ alter table emp
add bonus number(7,2);

Emp

EMPNO	ENAME	SAL	BONUS
1	SCOTT	5000	
2	KING	6000	

→ alter table emp
drop column bonus

Increase width of column

→ alter table emp
modify ename varchar2(30);

Whenever you create table,
create extra column (num, varchar2,)
for future use. (Extension Table)

reduce the width of column

```
alter table emp
modify ename varchar2(20);
```

Not Allowed

We can reduce the width if content are null

```
update emp
set ename = null;
alter table emp
modify ename varchar2(20);
```

↓

Solution

```
alter table emp
add x varchar2(25);
update emp
set x = ename, ename = null;
alter table emp
modify ename varchar2(20);
```

Data testing with x column

```
update emp
Set ename = x;
alter table emp
drop column x;
```

Change datatype of Column.

→
```
update emp
set empno = null;
alter table emp
modify empno number(4);
```

COPY rows from One table into another table.

EMP2

EMPNO	ENAME	SAL

→
```
insert into emp2
select * from emp
```

COPY of a table.

- for software testing purposes.

→
```
create table emp_copy
as
select * from emp;
```

1) When you create subqueries, indexes on original are not copied into the new table

2) When you copy a table using subquery, constraints on original are not copied into the new table (except for the not null constraint)

→
```
create table emp_copy
as
select empno, ename from emp
```

Copy a structure of table

→ create table emp_struct
as
select * from emp ; Method 1

→ delete emp_struct ;
→ commit ;

Instead of delete & commit

→ truncate table emp_struct ;

DELETE	TRUNCATE
ANSI SQL	Extra in oracle & MySQL
Requires commit	Auto commit
DML Command	DDL Command
can specify WHERE clause	cannot specify WHERE clause.
Free space is not deallocated	free space is deallocated.
Delete trigger on table will execute	Delete trigger on table will not execute.

Emp Table (1 million rows)
1000 mB = 1GB

→ delete emp ; It still occupy 1000 mB
commit ;

free space is not deallocated
free space is not surrendered by Emp Table
free space is not made available for other table.

To deallocate the free space :-

drop table emp ;
create table emp - - - - -

Better Solution

→ Truncate table emp
It will delete & Commit and also deallocate the free space.

→ create table emp_struct
as
select * from emp Method 2
where 1 = 2 ;

Rename a column

→ create table emp2
as
select empno, ename, sal Salary
from emp;

Alias

→ drop table emp;

rename emp2 to emp;

Change position of Column

→ create table emp2
as
select ename, sal, empno
from emp;

→ drop table emp;

rename emp2 to emp;

Day

PAGE No.
DATE / /

Oracle 11g - SQL - Constraints

primary Key

EMP

EMPNO	ENAME	SAL	DEPTNO
1	A	5000	1
2	B	6000	1
3	C	7000	1
4	D	9000	2
5	E	8000	2

1) limitation or restriction imposed on a table

Different types of Constraints

A) PRIMARY KEY (Primary column)

1) column or set of column that uniquely indentifies the row.
2) Duplicate values are not allowed (has to be unique).
3) Null values are not allowed (mandatory column).
4) It's recommended that every table should have a primary key.
5) Purpose of Primary key is row uniqueness (with the help of primary key we ... wish between 2 rows of a table)
... and long raw cannot be
... Blob cannot be primary

8) Unique Index automatically created.
9) You can have only one primary key per table.
10) ~~CANOT~~

CANDIDATE KEY eg:- PAN ID, PP NO

1) It is not a constraint
2) It is a defination.
3) Beside primary key, any other column in the table that can also serve the purpose of primary key, is a good candidate for primary key is known as Candidate key.

COMPOSITE PRIMARY KEY

1) Combine 2 or more Inter dependent columns together to serve the purpose of primary key.
2) In Oracle you can combine 16 column in a composite primary key.
3) MySQL you can combine 32 column in a Composite primary key.

STEPS for Identifying the primary key

1) When you create a table, identify some key column and make it the primary key of your table.
2) If you cannot identify some key column then try for composite primary key
3) If you cannot identify Composite Primary key, then add an extra column to

the table and make it the primary key of your table.

4) If you declare a composite primary key then the index that is automatically created would be a composite Index.

SURROGATE KEY

1) It is not a constraint
2) It is a defination
3) If you cannot identify a primary key in the table, you add a extra column to the table to serve the purpose of primary key such a primary key column that is not an original column of the table is known as surrogate key.
4) For Surrogate key column CHAR datatype is recommended (because the primary key column is the best column for searching and with CHAR datatype the searching and retrieval will be very fast)

How to declare primary key

```
Create table emp
(
empno char(4) Primary key,
ename varchar2(25),
sal number(7,2),
deptno number(2)
);
```

Internally a Constraint is an Oracle created function, it perform the validation.

→ select * from user_constraints
where table_name = 'EMP';

CONSTRAINT_NAME	CONSTRAINT_TYPE
SYS18492	P

→ select * from user_cons_columns
where table_name = 'EMP';
will give column information
(will show which column is primary key)

To Drop Constraint

→ alter table emp
drop constraint sys18492;

To name constraint

→ empno char(4) contrainst pk_emp_empno
primary key
ename varchar(25),

constraint pk_emp_empno → Optional

→ alter table emp
drop constraint pk_emp_empno

1) Unique index that are automatically created, it happens to have the same name as that of the constraint name due to which created
2) If you try to drop this index then oracle will not allow it
3) however if you drop the constraint then that index will be drop automatically
4) later if you want to retain the index you will have to create a new index manually

→ Select * from user_index

pk_emp_empno

→ Drop index pk_emp_empno Not Allowed

How to give composite primary key

→
```
Create table emp
(
  empno char(4),
  ename varchar2(25);
  sal number(7,2),
  deptno number(2),
  constraint pk_emp_deptno_empno
  primary key (deptno, empno)
);
```

constraint pk_emp_deptno_empno → Optional

Constraint are of 2 types :-

1) Column level constraint
(specified on 1 column)

2) Table level Constraint
(Specified on 2 or more column)
(Composite) (has to be specified at the end of table structure)

How to change the primary key to another column.

→ alter table emp
add constraint pk_emp_empno
primary key (empno);

constraint pk_emp_empno → Optional.

NOT NULL Constraint

1) null values are not allowed
2) Duplicate values are allowed.
3) Can have any number of not null constraint per table (unlike primary key)
4) Cannot have a composite not null constraint
5) always a column level constraint

PAGE No.
DATE / /

→ Create table emp
(
empno char(4)
ename varchar2(25) constraint nn_emp_ename
not null,
sal number(7,2) constraint nn_emp_sal
not null,
deptno number(2)
);

constraint nn_emp-ename → optional
constraint nn_emp-sal → optional

→ alter table emp
modify ename varchar2(25)
constraint nn_emp-ename
not null;

To add constraint ename afterward.

Oracle v6 → only not null constraint.
Oracle v7 → all other constraints introduced.

Solution for Candidate key column

1) not null constraint + unique index
2) Indirectly we can have more than 1 Primary key per table.

ALTERNATE KEY

for a candidate key column, if you apply a not null constraint and create an unique index, then the candidate key column becomes an alternative to primary key, such a candidate key column is known as Alternate key.

SUPER KEY

When Candidate key becomes alternate key, then the primary key is known as Super key.

UNIQUE

1) Duplicate values not allowed.
2) Null values are allowed
3) Can have any number of null values.
4) In oracle, long & long raw cannot be unique.
5) In MySQL, Text and Blob cannot be unique.
6) Unique index automatically created
7) In Oracle can combine upto 16 column in a composite unique
8) In MySQL, can combine upto 82 column in a composite unique.

9) Can have any number of unique constraint per table (unlike primary key)

→
```
Create table emp
(
    empno char(4),
    ename varchar2(25),
    sal number(7,2),
    deptno number(2),
    mob_no char(15) constraint u_emp_mob_no
        unique,                          → column level
    contraint u_emp_deptno_empno unique
        (deptno, empno)                  → table level
);
```

constraint u_emp_mob_no → optional
constraint u_emp_deptno_empno → optional

To add constraint

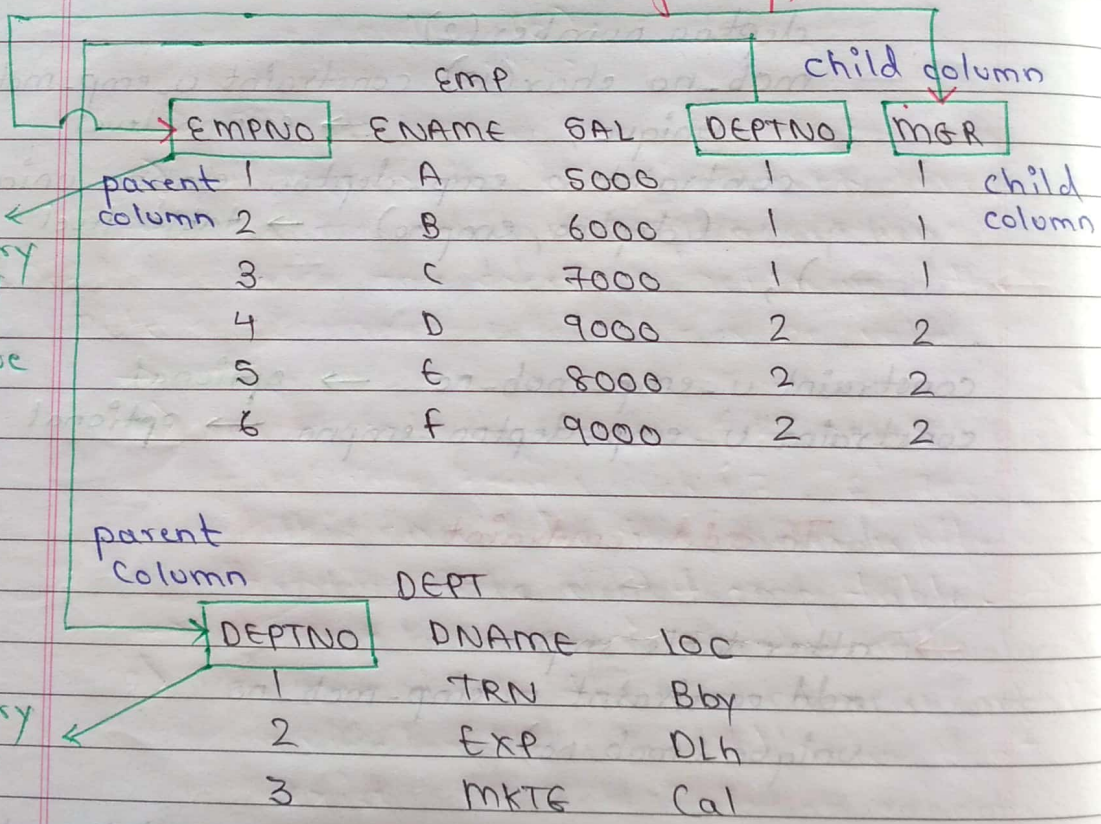
→
```
alter table emp
add constraint u_emp_mob_no
unique (mob_no)
```

10) Column level constraint can be written at table level but a table level composite constraint cannot be written at column level.

11) Column level constraint can be written at table level except for the not null constraint which is always a column level constraint and the syntax will not support you from it at table level.

12) one column may have 2 or more constraint
eg:- not null constraint + unique constraint
13) above is one more solution for candidate key.
14) In this situation also the candidate key becomes an alternate key

FOREIGN KEY (foreign key)



EMPNO	ENAME	SAL	DEPTNO	MGR
1	A	5000	1	1
2	B	6000	1	1
3	C	7000	1	1
4	D	9000	2	2
5	E	8000	2	2
6	F	9000	2	2

DEPTNO	DNAME	LOC
1	TRN	Bby
2	EXP	DLh
3	MKTG	Cal

1) Column or set of columns that references a column or set of columns of some table.
2) foreign key constraint is specified on the child column (not the parent column).

3) Parent column has to be Primary key or unique (this is a pre requisite for foreign key)

4) foreign key column may contain duplicate values (unless specified otherwise)

5) foreign key column may contain null values (unless specified otherwise)

6) foreign key column may reference column of same table also (known as self referencing)

→
```
create table dept
(
deptno number(2) constraint pk_dept_deptno
        primary key,
dname varchar2(15),
loc varchar2(10)
);
```

→
```
create table emp
(
empno char(4) constraint pk_emp_empno
        primary key,
ename varchar2(25),
sal number(7,2)
dept number(2) constraint fk_emp_deptno
    references dept(deptno), ON DELETE CASCADE
mgr char(4) constraint fk_emp_mgr
    references emp(empno)
```

while inserting values in emp table

→ insert into emp values ('7', 'G', 7000, 99, null);
error

while inserting value in MGR

→ insert into emp values ('7', 'G', 7000, null, 10);
error

7) All constraint are at SERVER LEVEL you may perform DML operation using any frontend software, the constraint will always be valid.

Dr. Codd 12th Rule

8) insert into dept values (4, 'SALES', 'BLR');
insert into emp values ('7', 'G', 7000, 4, null);
This should be the order while inserting into dept & emp table.

9) You can delete the parent row (master) provided child (detail) rows don't exist

10) You cannot delete the parent (master) row when child (detail) row exist
error :- unable to delete master row

PAGE No.
DATE / /

ON DELETE CASCADE :- If you delete the parent row then oracle will automatically delete the child row also.

To preserve the child rows :-

→
```
Update emp
set deptno = null
where deptno = 2;
```

→
```
delete dept
where deptno = 2;
```

You cannot update parent column if child rows exist

```
update dept
set deptno = 4
where deptno = 2;
```
Not allowed

To achieve on update cascade in Oracle.

```
update emp
set deptno = null
where deptno = 2;
```
↓
```
update dept
set deptno = 4
where deptno = 2;
```

```
update emp
set deptno = 4
where deptno = null;
```
↑

2nd Solution

→ insert into emp values (4, 'exp', 'DIH')

→ update emp
set deptno = 4
where deptno = 2;

→ delete dept
where deptno = 2;

3rd Solution

→ alter table emp
disable constraint fk_emp_deptno;
/* data correction */

→ alter table emp
enable constraint fk_emp_deptno;

To see constraint enable/disable

→ select * from user_constraints;

o/p - STATUS
Enable/disable

Day

PAGE No.
DATE / /

EMP

EMPNO (PK)	ENAME	MGR (FK)
1	A	1
2	B	1
3	C	1
4	D	2
5	E	2
6	f	2

→ Insert into emp
values ('7', 'f', 7000, 2, '7'); Oracle will allow

1) first it INSERTs, then it checks for the constraint.
2) Constraint checking is DEFFERRED (delayed) AFTER INSERT

→ Insert into emp
values ('8', 'f', 7000, 2, '9'); Oracle will not allow

1) first it INSERT then it checks for the constraint then it will Rollback and give an error message.

EMP 2

EMPNO	MGR
7	10
8	9
9	8
10	7

→ INSERT into emp
select * from emp2; Oracle will not allow

first it INSERT, then it checks for the constraints then it will Rollback ALL the rows, and give an error message

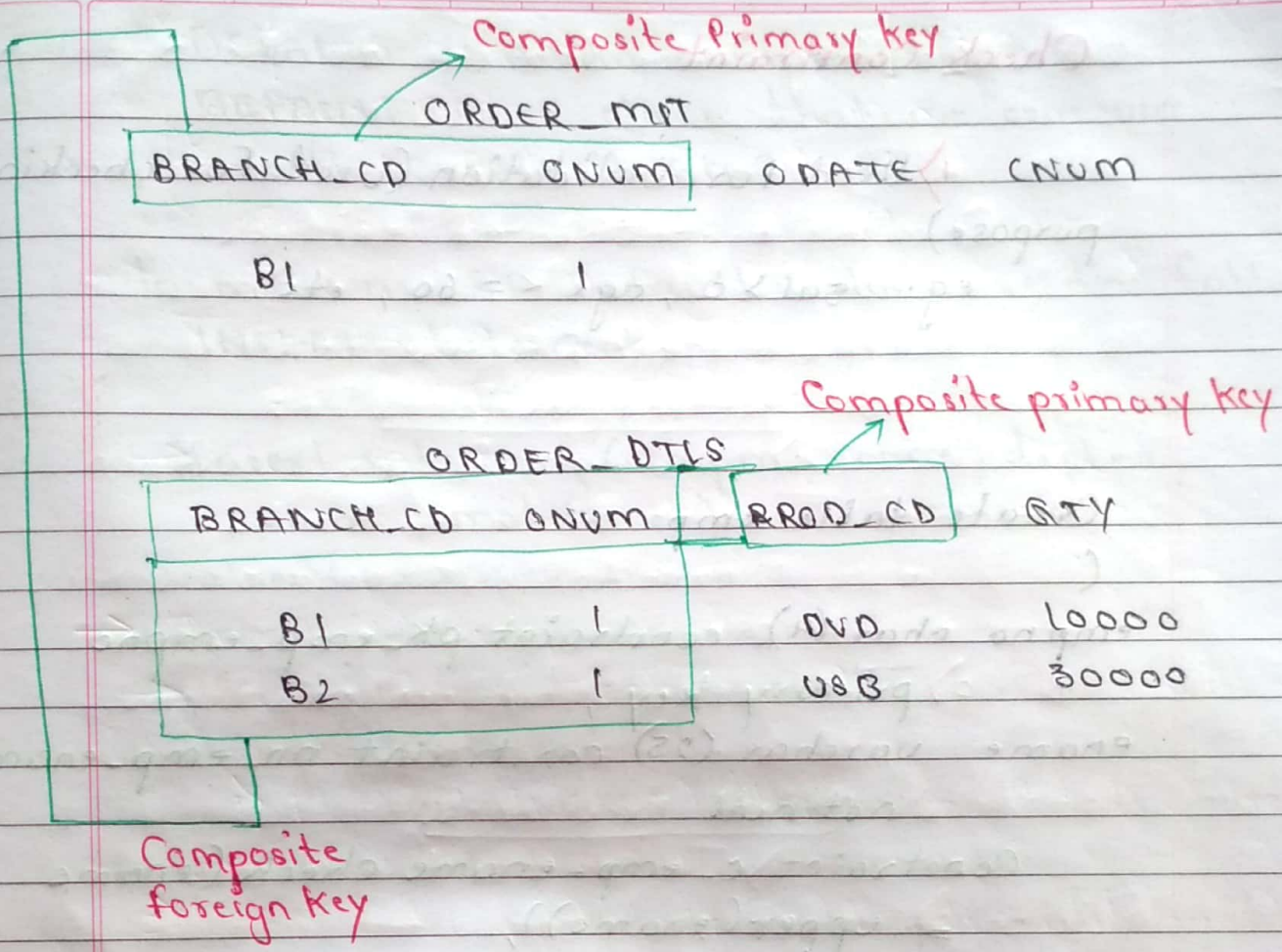
→ insert into emp Oracle will allow
select * from emp2

1) first it INSERT then it checks for the Constraint.
2) Constraint checking is DEFFERED after insert

Assumption :- fk constraint between MGR and EMPNO.
Assumption :- On Delete cascade has been specified.

Delete emp
where empno = 1;

AVOID ON DELETE & CASCADE in the event of self refrencing you may end up deleting a more rows than expected.



Composite foreign key:-

```
Create table order_mst ....;
Create table order_dtls
(
branch_cd char(4),
cnum number(6),
prod_cd char(4),
qty number(8),
Constraint pk_abc primary key(branch_cd,
    cnum, prod_cd);
Constraint fk_abc foreign key(branch_cd,
    cnum)
reference order_mst(branch_cd, onum)
); ON DELETE CASCADE
```

Check Constraint

1) Used for validation (used for checking purpose)

eg:- sal > 0, age <= 60, etc.

```
Create table emp
(
empno char(4) constraint pk_emp_empno
      primary key,
ename varchar(25) constraint nn_emp_ename
      not null
   Constraint c_emp_ename check(ename
      = upper(ename)),
sal number(7,2) default 7000
constraint c_emp_sal check(sal > 5000 and
c                sal < 95000),
deptno number(2) constraint fk_emp_deptno
   reference dept(deptno);
status char(1) default 'T'
constraint c_emp_status check(status in
                ('T','P','R')),
comm number(7,2) constraint nn_emp_
      ename not null,
mobile_no char(15) constraint u_emp_mobile
      no unique,
constraint c_emp_sal_comm check
      (sal + comm < 190000)
);
```

DEFAULT is not a constraint
DEFAULT is a clause that we can use with Create table

to make use of Default value, use the following INSERT statement :-

Insert into emp(empno, ename, deptno, comm, mobile_no)
values (. . . . . .

If you copy a table
eg:- Create table emp2
as
select * from emp
then indexes and constraints of original table are not copied into the new table except for the not constraint.

List of Operator we can use in the constraint

Relational Operator
logical "
Arithmetic "
Special "

eg:- IN, LIKE, BETWEEN, etc
Can call Single row function also
eg:- upper, lower, etc.
(Except Decode)

Null_ability is a feature of the datatype

Oracle 11g - SQL - Privileges.

Privileges are of 2 types

1) System privileges
   1) privileges to give certain command.
      eg:- create table, create sequence, create user, etc.
   2) Total of 80 system priveleges.

2) Object Privileges.
   1) To access the object of other users.

1) SYSTEM

In MySQL

1) When you install MySQL, root user is automatically created.
2) root user is having DBA privileges

In Oracle

1) When you create a database, 3 users are automatically created :-

SCOTT :- by default the password is tiger

1) It is a regular user
2) has connect, resource and create view permissions

SYSTEM :- by default the password is Manager

1) has DBA privileges
- Create user, assign privileges, exp_full_database, imp_full_database, database configuration, performance monitoring, performance tuning etc.

SYS :- by default the password is 'change_on_install'

1) This user cannot be dropped.
2) most important user
3) owner of database
4) owner of system table
5) Startup database, shutdown database, perform recovery etc is function of database.

SQL> Connect System

SQL> Create user dac1 identified by dac1

username / password

SQL> grant connect, resource to dac1

grant [privilege] [role] to [user] [role] [public] [with admin option];

SQL> grant create table to dac1;

SQL> grant create table, create view,
create session to dac1;

SQL> grant create table, create view to
dac1 with admin option;

SQL> grant create table to dac1;

SQL> grant create priv1, priv2, priv3
.... priv50 to user1, user2,
user3 ---- user140;

Role :- 1) Set of privileges.

DBA > create role student;

DBA > grant create table,
create view,
create sequence to student;

DBA > grant student to dac1, dac2, dac3.

2) One role can be granted to another Role.

→ grant create table to public;

PAGE No.
DATE / /

DBA > any user who has all 80 system privileges with admin option.

DBA > grant dba to dac1

DBA > pre defined Role (already available with the system);

DBA> revoke create table from dac2

DBA> select * from user_roles

DBA> select * from user_role_privs;

DBA> select * from all user;

2) OBJECT Privileges

To access the object of other users

Grant/Revoke (DCL)

grant [object prov] to [user] [with grant option]
(on object; all; role; public)

~~grant t from~~

SCOTT_SQL> grant select on emp to king;

SCOTT-SQL> grant insert on emp to king;

SCOTT_SQL> grant update ~~on~~ on emp to king;

SCOTT-SQL> grant ~~update~~ delete on emp to king;

SCOTT_SQL> grant all on emp to king;

SCOTT_SQL> grant select on emp to king;

SCOTT_SQL> grant select on emp to king, sam, huma;

SCOTT_SQL> grant select on emp to public

SCOTT-SQL> revoke select on emp from king;

SCOTT_SQL> grant select, insert on emp to king with grant option.

PAGE No.
DATE / /

Select

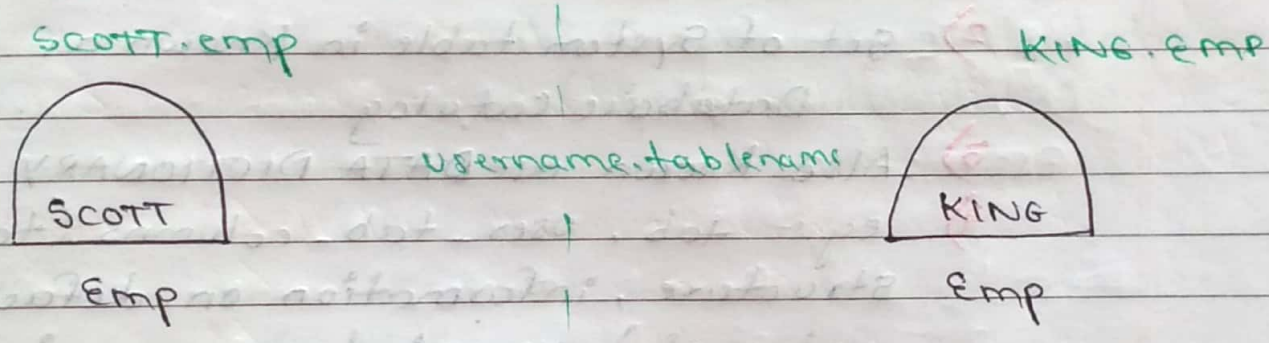
KING_SQL> grant select on scott.emp to sam;

→ select * from user_tab_privs;

→ select * from user_tab_privs_recd;

→ select * from user_tab_privs_made;

Access the table of other users.



Select * from emp;

select * from emp
select * from scott.emp

Insert into scott.emp
values ... ;

Update Scott.emp set...;

Delete Scott.emp
where ... ;

In Oracle

Username.tablename

In MySQL

Databasename.tablename.

SYSTEM TABLE

1) 2000 system tables in Oracle 11g (63 in MySQL)
2) Set of System table in known as Database catalog
3) Also known as DATA DICTIONARY
4) eg :- tab, user_tab_columns (table structure, information on default value etc) user indexes, user_ind_colum user_constraints, user_cons_columns, all users, user_roles, user_role & privs, user_tab_privs, user_tab_privs_recd, user_tab_privs_made, dual etc.
5) DDL for user is DML for system tables.
6) All system ~~a~~ table are READ_ONLY.
7) We can only SELECT from system tables.
8) Store data in upper case.

PAGE No.
DATE / /

DATA

It is of two type

1) User data
   a) user created
   b) user tables and indexes.

2) System data
   a) Oracle created
   b) Data that is stored in system tables
   c) also known as Metadata (data about data).

Oracle Unix

1) Most Secure Unix
2) Known as Unbreakable linux
3) has been designed specifically for testing the Oracle database.

views is present in Excel & foxpro.

PAGE NO.
DATE / /

Oracle 11g - STORED OBJECTS

1) object that are stored in the database
2) eg :- tables ; indexes.

VIEWS

1) present in all RDBMS and some DBMS.

user (AMIT)

EMP

EMPNO	ENAME	SAL	DEPTNO
1	A	5000	1
2	B	6000	1
3	C	7000	1
4	D	9000	2
5	E	8000	2

2) It is a handle to a table
3) Stores the address of table
4) view is a HD pointer. (Known as LOCATOR)
5) Used for indirect access to the table.
6) Used for SECURITY purposes
7) Used to restrict the access of users.

→ create view viewname ... ;

```
AMIT_SQL> create view v1
          as
          select empno, ename from emp;
          view created.
```

VIEWNAME & Tablename cannot be the same.

```
AMIT_SQL> select * from v1;
```

EMPNO	ENAME
1	A
2	B
3	C
4	D
5	E

```
AMIT_SQL> grant select on v1 to dac1;
```

```
DAC1_SQL> select * from amit.emp      Error
```

```
DAC1_SQL> select * from amit.v1;
```

EMPNO	ENAME
1	A
2	B
3	C
4	D
5	E

8) U... restrict the column access

System table (2000 tables) are views.
eg:- tab

9) form of Encapsulation (form of data hiding)
10) VIEW does not contain Data
11) Only the defination is stored, data is not stored.
12) View is a stored query
13) SELECT statement on which the view is based, it is stored in the DB in the compiled format.
14) VIEW is an executable format of SELECT statement
15) hence the execution will be very fast.
16) hiding source code from end user.

AMIT_SQL> grant select, insert on v1 to DAC1;

DAC_SQL> insert into amit.v1 values (6, 'f');

1) DML operation will perform on a view
2) DML operation performed on a view affect the base table
3) Constraint specified on base table will always be enformed even if you insert via the view.

To DROP VIEW.

→ drop view viewname

→ drop view v1

AMIT_SQL> Create view v1
as
select * from emp
where deptno = 1;
View created.

AMIT_SQL> grant select, insert on v1 to dbdal;

DBDAL_SQL> Select * from amit.v1;

EMPNO	ENAME	SAL	DEPTNO
1	A	5000	1
2	B	6000	1
3	C	7000	1

1) Used to restrict the row access.

DBDAL_SQL> insert into amit.v1 values (6, 'f', 6000, 1)

It will allow for deptno 1

DBDAL_SQL> insert into amit.v1 values (6, 'f', 6000, 2)

It will allow, cause we restrict select in where clause

AMIT_SQL> Create view v1
as
select * from emp
where deptno = 1 WITH CHECK OPTION;

DBDA_SQL> insert into amit.v1 values (6, 'f',
6000, 2)

this will not allowed, it will give an error.

2) view with check option
3) to enforce different checks for different users.

→ create view v1
as
select empno, ename, sal, deptno, tax
from emp;

error "TAX" invalid identifier.

→ create FORCE view v1
as
select empno, ename, sal, deptno, tax
from emp;

Warning:-
view created with compilation error

→ select * from v1;
view " " has error

→ alter table emp
add tax number(7,2);
table altered

→ select * from v1;
now it will work.

FORCE VIEW used :-
a) When table structure are not known or are undecided.
b) During offshore development
c) If you plan to create the table at runtime. eg:- create a new ORDERS table each year.

FORCE view is not available in MySQL

→ create or replace view v1
as
select ename, sal from emp;
It will create view if it not exist.

→ desc v1

→ Create or replace view v1
as
select upper(ename) u_ename, sal * 12
annual from emp;

→ select * from v1;

1) view based on computed column, expression, function, etc. you need to specify an alias for the virtual column
2) Can only SELECT from the view
3) DML operation not allowed.
4) DML operations for such view is possible in Oracle if you write a INSTEAD of trigger.

→ Create or replace view v1
as
select deptno, sum(sal) from emp
group by deptno;
Computed column not allowed in view, give 'an Alias' to work

→ Create or replace view v1
as
select deptno, sum(sal) Sum_sal from emp
group by deptno;

→ select * from v1;

1) view based on Group by clause

→ Create or replace view V1
as
select dname, ename from emp, dept
where dept.deptno = emp.deptno ;
(It will not allowed)

→ select * from v1 ;

1) View based on join
2) can only SELECT from the view
3) DML operation not allowed
4) DML operation for such view is possible in Oracle if you write INSTEAD of trigger.

VIEW based on VIEW

→ 1) to exceed the limits of SQL
eg:- UNION of > 255 SELECTS
SUB QUERIES > 255 SELECTS
NESTING of > 255 levels for function within function

2) To simplify the writing of Complex SELECTS
eg:- join of 40 tables.

→ select * from tab ;

→ select * from user_views ;

1) If you drop the table then views remain.
2) If you drop and recreate the table.
then the associated view need to be recompiled.
3) If you alter the table
then the associated views needs to be recompiled.

VIEWS are used in
1) Migration
2) EIM (Enterprise Integration Management)
3) EAI (Enterprise Application Integration)
4) Data Mapping
5) To convert 3D tables into 2D table
6) To convert 2D table into 3D table
7) To apply relational ~~view~~ method on object tables.
8) To apply object methods on Relational tables etc

Types of VIEW

1) Relational View
2) Object view
3) Materialized View (Known as Snapshot)
4) Inline View
5) etc.