SELF JOIN :-

→ Joining a table by itself is known as self join. whenever we have some relationship between any two columns in the same table then we need to use self join.

Select x.empno, y.ename, x.ename manager from emp x, emp y where x.empno = y.mgr;

CROSS JOIN :-

→ cross join is the combination of or product of two or more than two variables ie m number of rows in one table and n number of rows in another table.

Syntax :-

ANSI style

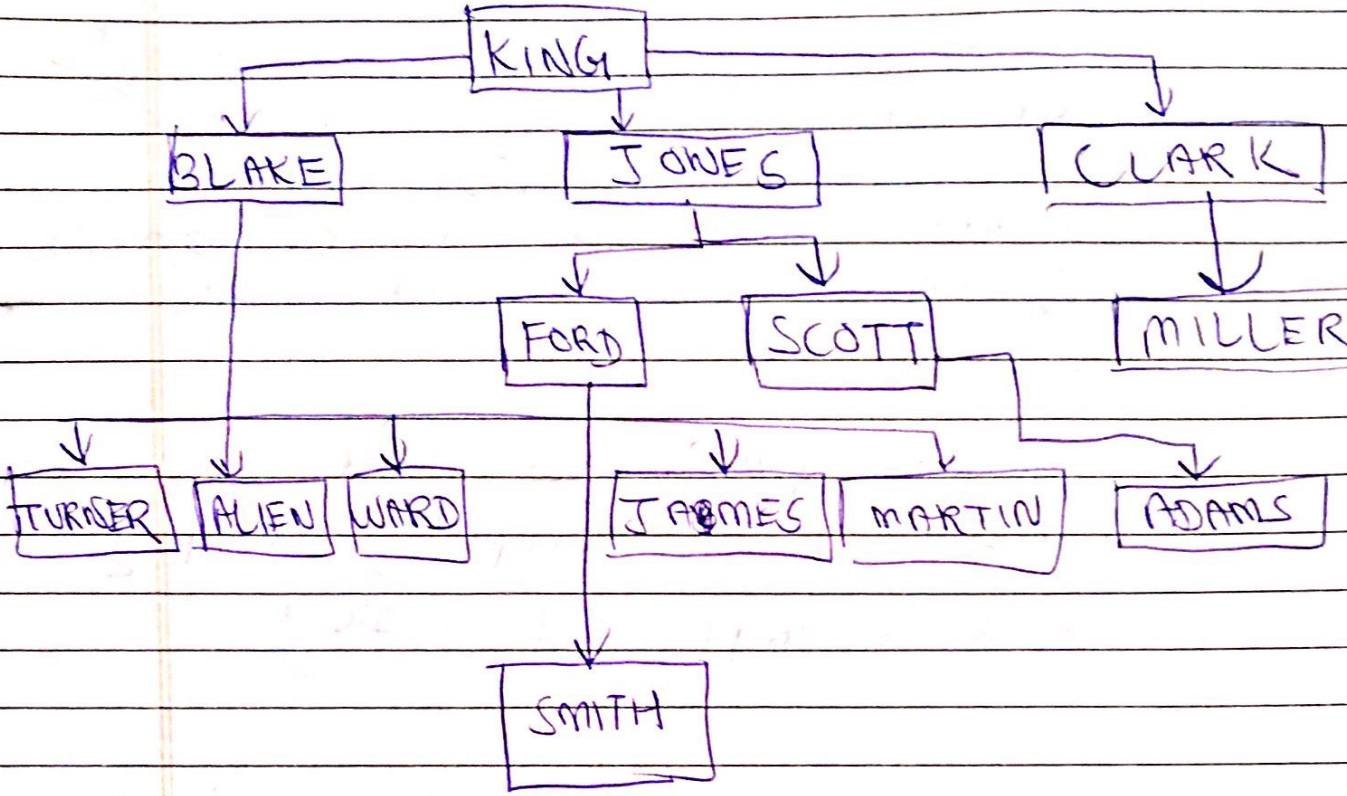
Select * from emp1 cross join dept1

Non ANSI style

select * from emp1, dept1;

→ Table 1 has 5 records, table 2 has 4 records then after cross join total number of records are 20.

HIERARCHIAL QUERIES



Syntax :-

```
Select ename || 'manager is' ||
prior ename "walk top down"
from emp start with ename =
'KING' connect by prior
empno = mgr;
```

/

```
King manager is
Blake manager is king
Jones manager is king
:
:
```

Hierarchial queries :-

→ using hierarchial queries you can retrieve data based on natural hierarchial relationship between rows in a table.

→ A relational database does not store records in hierarchial way.

→ hierarchial relationship exists between rows of single table, process called tree walking

→ A hierarchial query is a method of reporting

NOTE :-

hierarchial trees are used in various fields such as corporate management, manufacturing & scientific research.

SELECT :- it is SQL statement

LEVEL :- for each row returned by hierarchial query, the LEVEL pseudo column returns 1 for root row, 2 for child of a root and so on

From clause :- Specifies the table containing the columns, you can select from only one table.

Where :- restrict the rows returned by the query without affecting other rows of hierarchial comparision with expression

Starts with :- specifies root row of the hierarchy

connect By :- Specifies the column in which the relationship b/w parent & child.

Prior :- check whether row exists or not, this clause is required for hierarchial query.

Row Num / Row Id :-

→ These two represent Pseudo columns whenever we insert the records in the table, oracle internally maintain seprate row id's and row numbers for each record of the table

→ Here rownum are in format of 1, 2, 3 whereas rowid is in encrypted format.

→ Rowid contains 18 characters in that first 6 characters represent data object character.

Next 3 char represents file number

Next 6 char represents block number

Last 3 char represents row number

Syntax to see the list of rownum/rowid of a particular table →

Select rownum, rowid from

<table - Name>

ROW NUM	ROW ID
1	AAAJNAAEAAAAAKQAAA
2	AAAJNAAEAAAAAKQAAB
3	AAAJNAAEAAAAAKQAAC

Select * from upt where Rowid like '%A'

it will display first record

NOTE :- We cannot check the condition on rownum using

=, =!, ≥, >

except [< or <=]

ORACLE database Architecture

→ Oracle database architecture consists of two structures one is logical structure second is physical structure.

→ The idea behind dividing the structure in two parts is :-

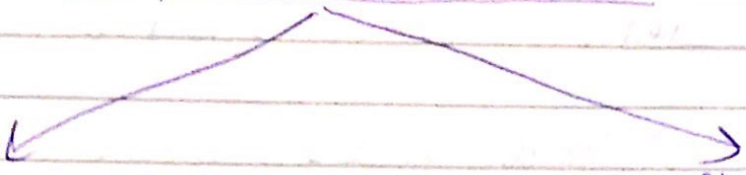
(i) If physical structure change, it won't effect to logical structure, so that we can make them independent.

(ii) But physical structure always depends on logical structure.

→ In an layman terminology logical structure is treated as building plan whereas physical

Structure is always building.

D/B Architecture



Logical Structure	Physical Structure
1) Table Space	1) CONTROL FILES
2) Schema	2) READ LOG FILES
	3) DATA FILES

LOGICAL STRUCTURE

1) Table Space :- i) It can be defined as logical storage unit which collectively stores database data.

ii) stores data logically stored in table spaces, physically stores in database files.

→ A database is made up of one or more table spaces.

→ Each table space is made up of one or more data files.

→ The sum of size of all the table spaces representing the storage capacity of database.

→ Every database has atleast minimum one table space ie system table space.

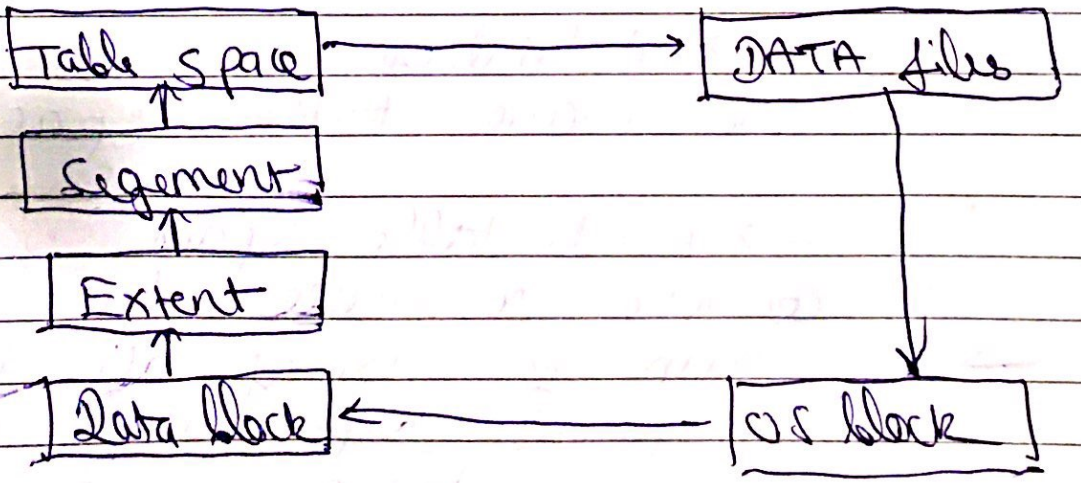
→ From oracle 10g onwards they introduce a new table space called SYSAUX table space.

Advantage of table space :→

→ It is used to control the availability of data. This possible by using online/offline options.

→ It is used to manage disk space for database data.

→ Table space makes it possible partial database recovery and backup.



Data Block :- It is a logical memory where data can store logically and it can also be called as logical ~~data~~ blocks, oracle blocks or pages.

Extent :- The next level of logical database (space) is called extent & it is nothing but the collection of continuous datablocks that are allocated for storing specific information.

Segment :- The next level of logical database space is segment (table) and it is nothing but, collection of extents.

→ Each segment has initial area of disk space called initial extent segments are broadly divided into four types :-

(i) Data segment
(ii) Index segment
(iii) Rollback segement
(iv) Temporary segement

Data segment → The data where we are storing within the database is known as ~~database~~ data segement.

Index segment → The segement contains the information about the indexes.

Rollback Segment :-

It stores the rollback information. This information is used when the data has to be rollback.

Temporary Segment :-

→ It is created when a SQL statement needs a temporary work area.

→ These statements gets destroyed once SQL statement gets executed.

③ SCHEMA :-

It is the collection of logical objects such as table, indexes, clusters, views, sequences, stored procedures, functions, packages and triggers grouped together is known as schema. And these are all called as schema objects.

DCL → (data Control Langage)

→ It is the fourth sub-langage in SQL, which containe two commands :-

1. GRANT
2. REVOKE

→ Steps to enter into DBA account :-

1. Enter user-name : / as sysdba
2. SQL > show user
   USER is "SYS"
   syntax to creat user account, user name without any permissions :-
3. SQL > create user akshay identified by twinkle;
   twinkle → password

Syntax to creat the user account with permissions :-

grant creat session, creat table to sumit identified by pandey;

GRANT command :-

→ This command is used to granting the privileges and roles to the specific user.

→ Here, privilege can be defined as any command permissions such as create table, create any table, alter table, alter any table etc.

Role :- collection of privileges can be called as role.

→ Roles are divided into two types :-

(i) system-defined / pre-defined
(ii) user defined / DBA defined roles

Syntax of grant command :-

Grant priv 1, role 1, priv 2, rol 2 ..... priv n, roln to <User name>;

Eg :- Grant connect to Akshay;

It contains all the roles, all DDL, DML, TCL, PL/SQL operations can be program by akshay.

grant resource to akshay;

→ resource contains 9 privileges.

→ Select * from session_privs;

↳ This is how we can see how many privileges are given to me.

Granting the privileges to user on another users table :-

→ Grant Select on scott.emp to Akshay;

Now akshay can Select Scott's emp table. but he cannot program any operations.

Now if you want to program any operation DBA will give you permission

→ Grant update, insert, delete on scott.emp to akshay;

Revoke :-

This command is used to taking back the permissions from the ~~other~~ user.

Eg :- Revoke select, updat, insert, delete on scott.emp from akshay;

Syntax :-

revoke priv 1, role 1, priv 2, role 2 ... priv n, role n from <user-name>;

Granting the permmision to akshay by scott :-

→ First, go into scott account and then,

Eg :- grant select, update, insert, delete on emp to akshay;

Syntax to create DBA defined roles :-

Role1

→ Create role myrole1;

→ Create role <role name>;

Now,
I want to ~~give~~ insert privileges into ~~role~~ myrole1

Syntax:-

→ grant select on scott.emp to myrole1;

→ grant select on scott.emp1 to myrole1;

→ grant select on scott.dept1 to myrole1;

Now,

Role2

Create role myrole2;

Grant insert on scott.emp to myrole2;

Grant insert on scott.emp1 to myrole2;

Grant insert on scott.dept1 to myrole2;

Here myrole Contains one privileges on three tables.

Myrole3

→ Create role myrole3;

→ grant delete on scott.emp to myrole3;

→ grant delete on scott.emp1 to myrole3;

→ grant delete on scott.dept1 to myrole3;

Now,

granting roles to the user By DBA:-

→ grant myrole1, myrole2, myrole3 to akshay

Now,

Syntax to merge the roles

→ grant myrole1, myrole2 to myrole3;

→ Now,

→ grant myrole3 to akshay;

Syntax to change the password to the user :-

Syntax :-

Login in to akshay account

③ SQL> password

old password:
new password:

→ user Name & password
Contains minimum 1 character
max 30 characters

Syntax to change the password with DBA :-

→ alter user akshay identified by tiger;

Granting the privileges with grant options :-

grant insert on scott.dept to u3 with grant option;

→ grant option is meant for object level privileges.

Granting the privileges with admin option →

grant Create Session to u1 with admin option;

Syntax to see all the list of privileges under a particular user :-

Select * from session_privs;

Select * from session_privs;

~~kill the session~~

Syntax for kill the user session →

→ desc v$session;

→ Select sid, serial#, username from v$session where username not in ('SYS');

Now if dBA want to kill the session of sumit or akshay then

~~alter system kill session '141, 59';~~

Syntax :-

alter system kill session 'sid, serial#';

Eg :-

alter system kill session '141, 59';

Syntax to see list of all the users :-

Select * from all_users;

OR

Select * from dba_users;

desc dba_users;

desc all_users;

Syntax to see the user id of a particular user :-

select uid from dual;

Syntax to lock the user account

→ alter user <user name> account lock/unlock

~~Syntax to see all the user defined roles :-~~

~~→~~

~~select * from~~

Syntax to see the list of privileges inside the role :-

desc role_sys_privs;

Syntax to drop the user account :-

drop user <user-name> cascade;