Primal - basic, original, fundamental.

⇒ Physical view तो सबसे 3rd level view तो आता।
(Database storage Related).

⇒ In partial case, External view is Ans.

3. In Relational Model, Cardinality is →.

a) No. of tuples
b) No. of attributes.
c) No. of Tables.
d) No. of Constraints.

⇒ If degree, then no. of attributes.

The no. of rows in a Table, called as Cardinality

4. _____ Constraint is used to maintain Consistency among tuples in two relations.

a) key
b) domain.
c) Referential Integrity
d) Entity integrity.

⇒ domain basically deals with that which type of data we put into Table. (integer, varchar, character, etc.)

⇒ Entity integrity, related with primary key.

⇒ key deals with uniqueness.

Top 5 Comp. Ques on Advance DBMS :→
(Big Data & Data Warehouse)

Q-1: What do data warehouse support?

a) OLAP
b) OLTP
c) OLAP & OLTP
d) Operational Database.

⇒ Data Warehouse, where we Integrate data at one place from all diff sources.
Ex:- Big BAZAAR, → their outlets mostly found in Every City.
&
(at the end of day, all data Integrated from all cities & kept) i.e, data Warehouse.

- & we store this data, to analyse on it later. So that, they can also play ADS - to diff users acc. to their data of purchasing items.
(Apply Mining algorithms on these data)

⇒ OLAP → online Analytical processing.
⇒ OLTP → online Transac" processing.
↳ It works on Current Data.

⇒ (that where to Remove item, when open new store, rename or add items) — all these based on Analysis.

(2.) Hadoop is framework that works with variety of related tools. Common group includes ——.

A) Map reduce, Hive & Hbase
B) Map Reduce, My SQL and Google Apps.
C) Map Reduce, Hummus, Iguana
D) Map Reduce, Heron, Trumpet

⇒ Hadoop, basically work on Big Data. (It is a Tool of Big Data).

↳ like Google, Facebook. Big data deals with the multiple petabytes of the data. Now, to process this much of Data. We don't use normal tool like SQL Server, Oracle. We use Hadoop. (Bcz, this data is also unstructured).
&
↳ (Hadoop Ecosystem / framework includes many small tools like map reduce. (used to process the data, reduce the data (divide) & then works on bench processing — i.e, works on Multi-processing),
⇒ Hive → (If we write to SQL commands in Hadoop, then use Hive),
⇒ Hbase → (helps in data storage),
↳ also tool like Zookeeper, flum, pig, etc.

2) Google Apps, it is part of cloud.

3.) All of the following accurately describe Hadoop, Except :

A) open Source — (listed in Apache, we also install it).
B) Real time
C) Java based
d) Distributed Computing approach.

→ Hadoop, It is basically batch processing. Means we first need data & then we analyse on it.

→ (In Real time, we use SPARK).

4.) Which of the following does not comes under five V's of BIG Data ?

a) Volume (how much amount of data).
b) Velocity (with which velocity, data ↑).
c) Variety (Structured, unstr., semi-structu).
d) Visualization

Note :- let, if there is 'x' amount of total data.
where, x → is all data on Earth,
then,
(80% - 90%) of x is created in last 5-6 years. Mean,
Data is increasing with so much speed.

Unstructured → photos, videos,
Semi-Structured → XML based data.

⇒ Value → (ie, value of our data).
⇒ Veracity → (means, trustworthyness).
↓
(how much capable our data to believe on it.)

⇒ 5 V's of BIG DATA :→
1.) Volume
2.) Velocity
3.) Variety
4.) Value
5.) Veracity.

⇒ Visualiza^, is a part of Data Analytics. where we visualise our data with help of graphs (pie chart, bar chart, ... etc.)

---

106. Deferred Database Modification :→

→ This topic comes under Log based Recovery.
↓
(If there is any failure inside our System, then Either we can recover that System or not)

→ Log, is basically a file (small sized file). in which we store our actions. that (what Trans. performs, are stored in Logs)

deferred → delayed, postponed, belated

Date / /
Page No.

→ (like, History in our Browser)
when our System fails, to recover Trans.
⇒ (either we have to Roll back them
or Modify them).

- We do that, by seeing the log.

⇒ 2 Methods, by seeing the log :→
1.) Deferred Database Modifica"
2.) Immediate

(Ex.)

A : 100 200
B : 200 400
Database (H.D).

T_1
R(A)
A = A + 100
W(A) — 200
R(B)
B = B + 200
W(B). — 400.
Commit.

Transac" log :→

< T_1, Start >.
< T_1, A, 200 >. new value.
< T_1, B, 400 >. new value.
< T_1, Commit >.

Redo →

⇒ Here, it don't update in Database hand-to-hande. It updates in Database after Commit. (bcz deferred - late, postponed).

After Commit, database updated.

⇒ How we Recovery in Deferred ? Let's, Say System fails after Commit.
→ So, when Recovery Manager comes, it first check trans. log file & check (Start & Commit) in it.

Then, Recovery Manager: REDO.

- Means, update the new values in the database.
- (Let, due to fail → our database also not there).
then,
- Recovery Manager puts the A & B value from Log in database.

A = 200
B = 400.

- (If it already in database, then overwrite हो जाएगा।).

Case II.

T1	Transac^n log.
A = 100, B = 200	
W(A). — 200.	<T1, Start >.
W(B). — 400.	<T1, A, 200 >.
fail.	<T1, B, 400>.
	Roll back.

- Now, fails before Commit. So, now database value is same as before.

Now,
after failure occur, when Recovery Manager opens the log file. → It sees T1 starts but not commit. So, Here Recovery Manager don't do anything. He Simply Roll back.

Means, पुरानी value को ही रखो, कुछ भी Opera^n करने की जरूरत नहीं है।

⇒ So, Deferred Modification also known as No UNDO/REDO Method.

Ex:-

<T1, Start>.
<T1, A, 200>
<T1, B, 400>
<T1, commit>. → (REDO)

<T2, Start>.
<T2, C, 500>. → (No Action).

Roll back नहीं होगा।

In Deferred, we store only new values.

---

107) Like Command in SQL :-

⇒ We use Like Command, generally to search the Data.

Q:- 1.) Find Employee detail whose name starting with 'A'.

2.) Find Emp detail whose name ending with 'n'.

3.) whose name contains 'ee'.

4.) whose name contain 'a' in 2nd place.

5.) whose name contain 'a' in 2nd place & name should contain total five characters.

% → Any value (कुछ भी chara. हो),
↳ यदि 'a' character से हो, कोई problem नहीं।

Emp.

ID	Name
1.	Varun
2.	Arun
3.	Karuna
4.	Amrit
5.	Ranjeet
6.	Ajeet

% → any value
& length.

_ → reserved for a value.

1.) Select * from Emp where name like 'A%';
Output → Arun, Amrit, Ajeet

2.) —"— like '%n';
Output → Varun, Arun.

3.) —"— like '%ee%';
Output → Ranjeet, Ajeet

Note:→ If Name → Saar, Arunee, ... any thing
↳ then, these also comes, bcz sirf 'ee' we need.

% → also 'a' characters include.

4.) —"— like '_a%';
Output → Varun, Karuna, Ranjeet.

5.) —"— like '_a___';
Output → Varun

In, C++ Here
=) =
=) :=
|| -> or

Date
Page No.

10.8: Basic PL-SQL programming with Execution :->

→ Program 1: Find the Sum of 2 numbers.

=> declare
Declare the. a int ;
b int ;
c int ;
begin
Execut able part a := &a ; // value given by user. (C to take input from user)
b := &b ;
c := a+b ;
dbms_output.put_line ('Sum of a and b =' || c);
end ;

// cout in C++
// : in C++

(#) Program 2: Greatest of 2 numbers →

→ declare
a int ;
b int ;
begin
a := &a ;
b := &b ;
if (a > b)
then
dbms_output.put_line ('a is greater') ; // a
Else
dbms_output.put_line ('b is greater') ; // b

SQL Live में हमें value पहले ही देनी पड़ती है उसमें बाद में user value नहीं दे सकता।.


```
end if;
end;
```

⇒ (both code works fine). ✓

(Q9) PL-SQL :- (while, for loop) :-

program 3:- for loop :-

```
declare
a number (2);
begin
for a in 0..10
loop
dbms_output.put_line (a);
end loop;
end;
```

// = 0 to 10.
// By default, increment of 1 by 1.

program 4:- while loop :-

```
declare
a int;
b int;
begin
a:=0;
b:=&b;
while a<b
loop
a:=a+1;
dbms_output.put_line (a);
end loop;
end;
```

// print from a to b.

→ Output → 1 to 10

than,
→ output → 0 to 9
(Now, first print then increment)
→ output → 10

→ In oracle, sometimes we don't see the output - then write.

→ `set serveroutput on` // but in like SQL it is by default.

Code

→ (both code works fine). ✓

11:0 Single Row & Multi Row functions in SQL :→

→ **Single Row :→**
If our func. is applicable on single row, and only apply on single row. & gives an single output corresponding to that row. → then, it is single row func.

→ **Multi-Row :→ func.**
func. that apply on more than one row, & gives an single output corresponding to all these rows.

→ Round (Round-off value)
→ Mod. (gives remainder after division)
→ lower (Convert a string into lower letters)
→ InitCap. (Capital the Initial letter).
→ Concat (Add 2 string & make 1).
→ LPAD/RPAD (for left & Right padding).
→ NVL, NVL2 (to take care of NULL values).

SQL functions -

- Single Row func^n.
  - Number func.
    - Round
    - Trunc
    - Mod
  - Character functions
    - Case Manipulation
      - lower
      - upper
      - initcap.
    - Character Manipulation.
      - Concat
      - Substr
      - length
      - Instr
      - LPAD/RPAD
      - TRIM
      - Replace.
  - Date function
    - Months_between
    - Add_Months
    - NEXT_DAY
    - LAST_DAY
    - ROUND
    - TRUNC
  - General functions
    - NVL
    - NVL2
    - NULL IF
    - COAL ESCE
    - CASE
    - DECODE
  - Data - Type Conversion.
    - TO_CHAR
    - TO_NUMBER
    - TO_DATE
- Multi_Row func^n.
  - Aggregate.
    - SUM
    - MAX
    - MIN
    - COUNT
    - AVG

111. Character functions in SQL with Execution :→

⇒ **Character functions**

- **Case function Manipulation**
  - → lower
  - → Upper
  - → Init Cap
- **Character Manipulation**
  - → Concat ('Varun', 'Singla') Varun Singla
  - → Substr ('Varun', 2, 4) arun
  - → Instr ('Varun', 'u') 4th
  - → length ('Varun') 5
  - → LPAD ('Varun', 10, '*') *****Varun (10 - length)
  - → RPAD —"— Varun*****
  - → TRIM ('V' from 'Varun') Arun
  - → REPLACE ('Varun', 'V', 'T') Tarun.

(*) ANURAG
1 2 3 4 5 6

(#) **Execution** → If we want to implement these funcs, so, first we have to make schema (table).

⇒ **Output Table:→** select * from emp;

ID	NAME
1	Code Smasher
2	Varun Singla

⇒

```
Create table emp
(
id int,
name varchar2(20)
);
insert into emp values (1, 'Gate Smashers');
insert into emp values (2, 'Varun Singla');
Select * from emp;
Select lower(name) from emp;
Select upper(name) from emp;
Select initcap(name) from emp;
Select Concat(id, name) from emp;
Select SUBSTR(name, 2, 5), INSTR(name, 'v')
    from emp;
Select length(name) from emp;
Select lpad(name, 15, '*') from emp;
Select rpad(name, 15, '*') from emp;
Select trim('v' from name) from emp;
Select replace(name, 'v', 't') from emp;
```

Table Created.

2 ROWS

2 ROWS.

Select 1 by 1 Each Row, & See the Output

Output:

REPLACE(NAME, 'V', 'T')
Gate Smashers
tarun Singla.

Output:

LPAD(NAME, 15, '*')
******Gate_Smashers
*********Varun_Singla.

Here, ~~*~~ counts (_) Space also.

Output:

SUBSTR(NAME, 2, 5)	INSTR(NAME, 'v')
ate S	0
arun	1

(~~*~~ don't count (_) here).

i.e.,

GATE	SMASHERS
1 2 3 4	5 6 7 8 9 10 11 12

→ ate S

(112) # View in Database :→

(Oracle, SQL Server Views)

- What is View in Database ?

→ Virtual Table :→

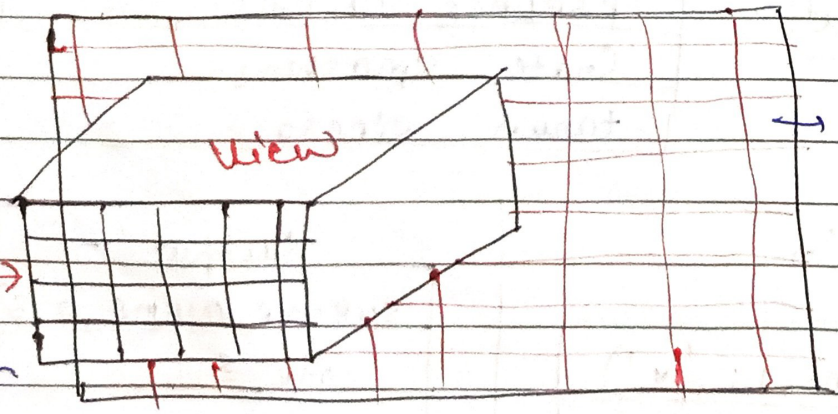
(The Table we create,
Create Table xyz
It takes physical space in memory (H.D.))

⇒ But, View is the virtual Table.
It looks like a Table, but it is not actually a table. It don't take space. any.

⇒ View is the result set of a stored query.

Query :→

```
Create view V1 as Select id from Student;
```



This view has no physical existence. We don't store it anywhere.

V1 - View -

(#) The Execute code of this query, Stores after compile. &

After compile, when we write

Select * from V1

then,

it shows the data like a table from the View

(#) So, actually we just store this little query, rather than Result. We don't store Result.

That's why it's a virtual Table.

⇒ Read-only (vs) updatable Views :→

⇒ If we made any changes in Base Table & that col^n is also in our view, then, obviously that also changes.

⇒ Same, If we delete from Base Table. Then also deletes from View.

(#) ⇒ But, if we change anything in View :→

And we want that changes to execute also in the Base Table, then updatable Views. &

If we Shut down the (Insert, Delete & update) (DML commands) on view, i.e, we disable these commands. So, that it can't Operate on View. Then, we made Read-Only Views for it.

Snapshot – जल्दी से photo लेना।

Date ___/___/___
Page No. ______

7. Materialised View :–
type of updated version.

(If our data is on the remote server, & I want a copy of that on my local server / machine. i.e, I want a snapshot of that remote data on my local Server. So, that is called Materialised View.
→ (This takes space, but takes less space as comparatively to that data).

(#) We can't apply any DDL command (Alter, etc.) on view.
But, apply only DML commands if we make the updatable view.

(#) We can insert the data of more than 1 table, in a View.
i.e.
(View can also take data from Multiple Tables.)

(#) We also can take any particular row from Table to make view. like
where address = 'Delhi';
So, all Delhi students come into the view.

(#) Advantages of View :–

1.) To restrict the Data Access.
(like, we don't give the access of original Table to our user, we just give view to them of some data.)

2.) To make complex queries Easy.

Ex.1) like, we some data from 3 tables.

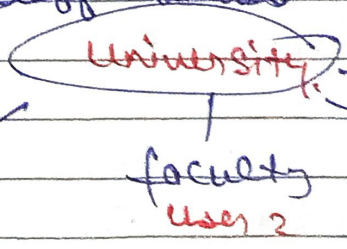
then,
By default, we apply Join as Nested Query.

So,
Its better to just make the views from these tables. Rather, than to write complex Queries

3.) To provide data Independence,
↓
(We just a give a particular access to a user of a Table, rather than giving the full database access).

4.) To present diff. views of the same data.

Ex.1



(They all have diff. privileges).

So,
→ We give diff. views to all of them of the same Data (Table).

(A) SQL CHEAT SHEET :→

(B) Examples →

1.) Select all rows from table with filter applied
→ Select * FROM tbl where Col1 > 5;

2.) Select first 10 rows for 2 columns
Select Col1, Col2 fprom tbl limit 10;

3.) Select all rows with multiple filters applied
→ Select * from tbl where Col1 > 5 AND Col2 < 2;

4.) Select all rows from col1 and col2 ordering by col1
→ Select col1, col2 from tbl order By 1;

5.) Return count of rows in table
→ Select Count (*) from tbl;

6.) Return Sum Of Col1
→ Select SUM (col1) from tbl;

7.) Return max value from Col1
→ Select MAX (col1) from tbl;

8.) Computes summary statistics by grouping col2
→ Select AVG (col1) from tbl GROUP By col2;