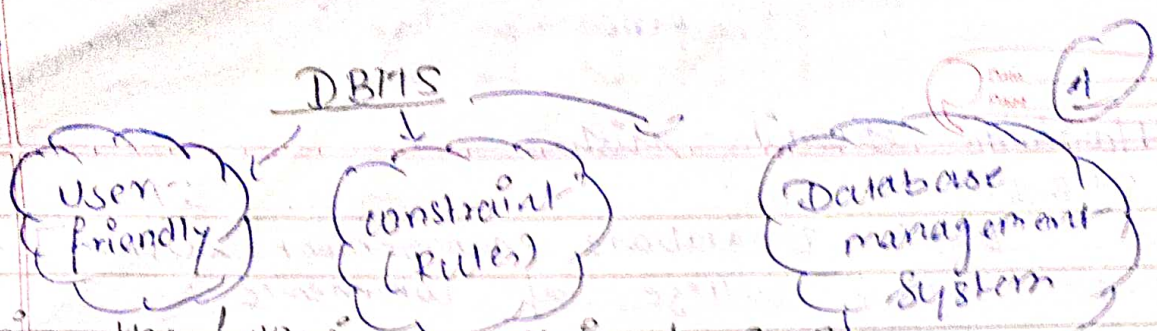
14/06/23



- Explain the following terms in 1 - 2 lines in terms of DBMS.

⊛ DRAWBACK OF FILE SYSTEM

- **DATA REDUNDANCY**

A situation that occurs in a database when a field needs to be updated in more than one table

- **DATA INCONSISTENCY**

A situation where various copies of the same data are conflicting, wastes storage space and duplicates effort.

- **DATA ISOLATION**

A property that determines when and how change made by one operation become visible to other concurrent user and systems.

- **DATA SECURITY**

security can be a problem with a file-based approach. (Protection from unauthorized access)

⊛ DBMS

- Databases are the collection of data in order to store and retrieve data.
- Data can be numerical, text and alpha-numeric, Audio, video & images.
- DBMS is a collection of programs that enables the user to create and maintain a database.

⊛ COMMON TERMINOLOGIES

- **TUPLE:** is also known as rows.
- **COLUMN:**
- **TABLE:** (i) collection of rows & column is known as table.

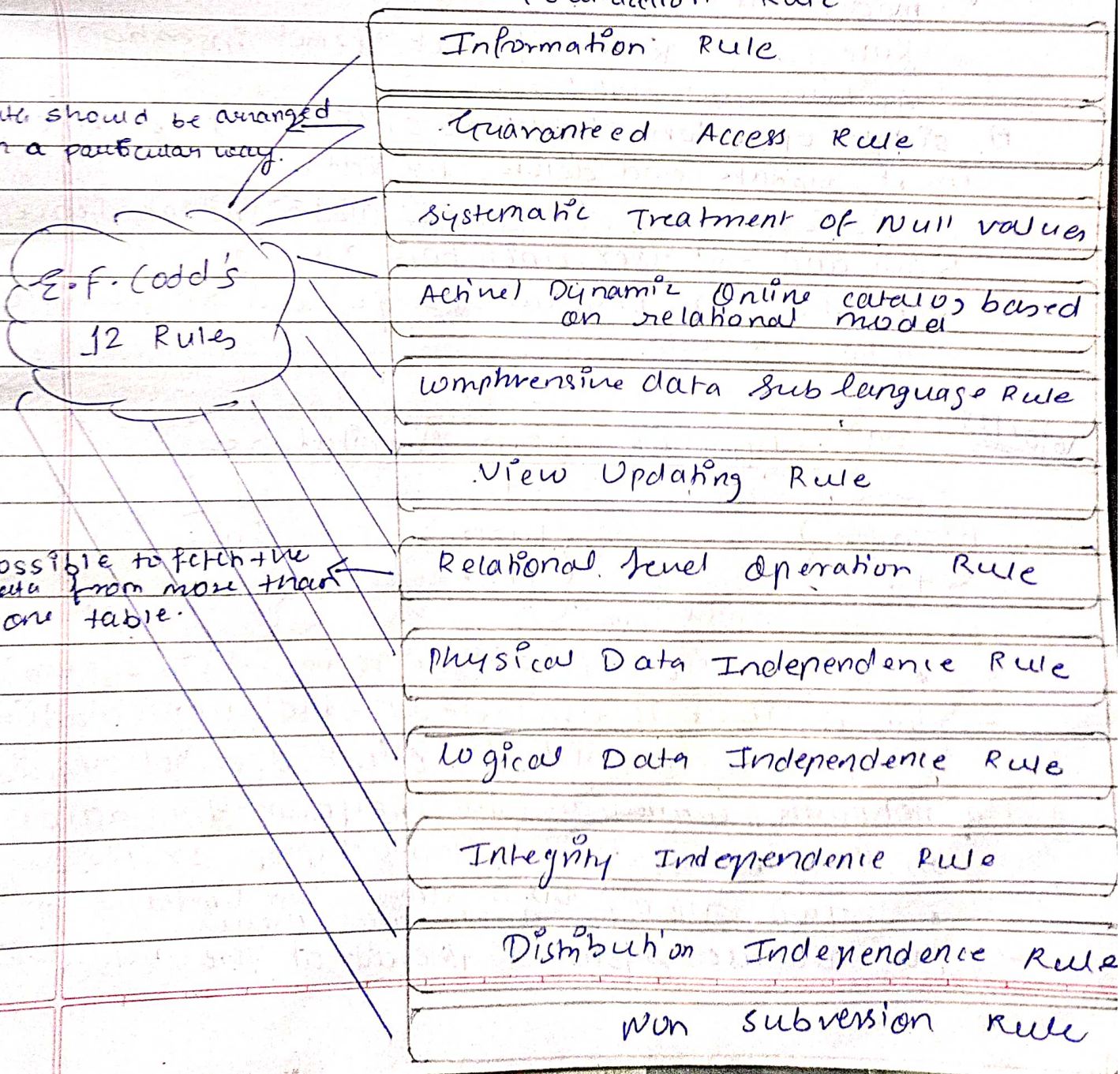
Relational model given by E.F. Codd which followed by 12 Rules in DBMS.

(ii) intersection of rows & column is known as cell

- **SCHEMA:** how the data should be organised & represented logically along with constraints is called as schema.
- **DATA REDUNDANCY:** Avoiding duplicacy of the data in database.
- **KEYS:** also known as 'constraints' which define restriction over the values of a table.

Q Write 12 Rules given by E.F. Codd. Explain any 4 in your own words. Journal 2nd Que.

Foundation Rule



Explanation

Page 3

Rule 3 : Rule of Systematic Null value support
Null values are completely supported in relational databases. They should be uniformly considered as 'missing information'. Null values are independent of any data type.

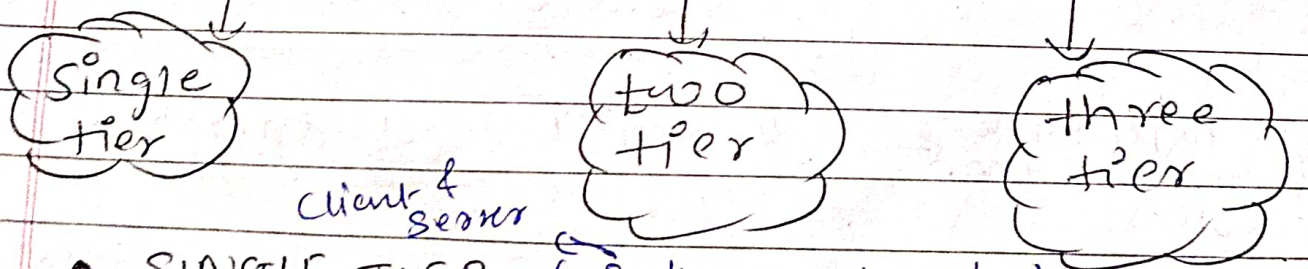
Rule 6: Rule of Updating views.
Views should reflect the updates of their respective base tables and vice versa. A view is a logical table which shows restricted data. Views generally make the data readable but not modifiable.

Rule 7: Rule of set level insertion, update and deletion.
A single operation should be sufficient to retrieve, insert, update and delete the data.

Rule 8: Rule of Physical Data Independence.
Batch and end user operations are logically seperated from physical storage and respective access methods.

06/23

* Types of DBMS Architecture



- SINGLE TIER (Both are found in one system)
- The database is available on the client machine
- Any request made by client does not require a network connection to perform the action on the database. Eg. Mam's laptop. or PC
- TWO TIER [Direct access when limited no. of users are there]

The database system is present at the server

db [database] = collection of tables

machine and the DBMS application is present at the client machine, these two machines are connected with each other through a reliable network. Ex. Tally

- THREE TIER

- another layer is present between the client machine and server machine.
- the client application does not communicate directly with the database systems present at the server machine. Ex. cache i.e. accessing via it.

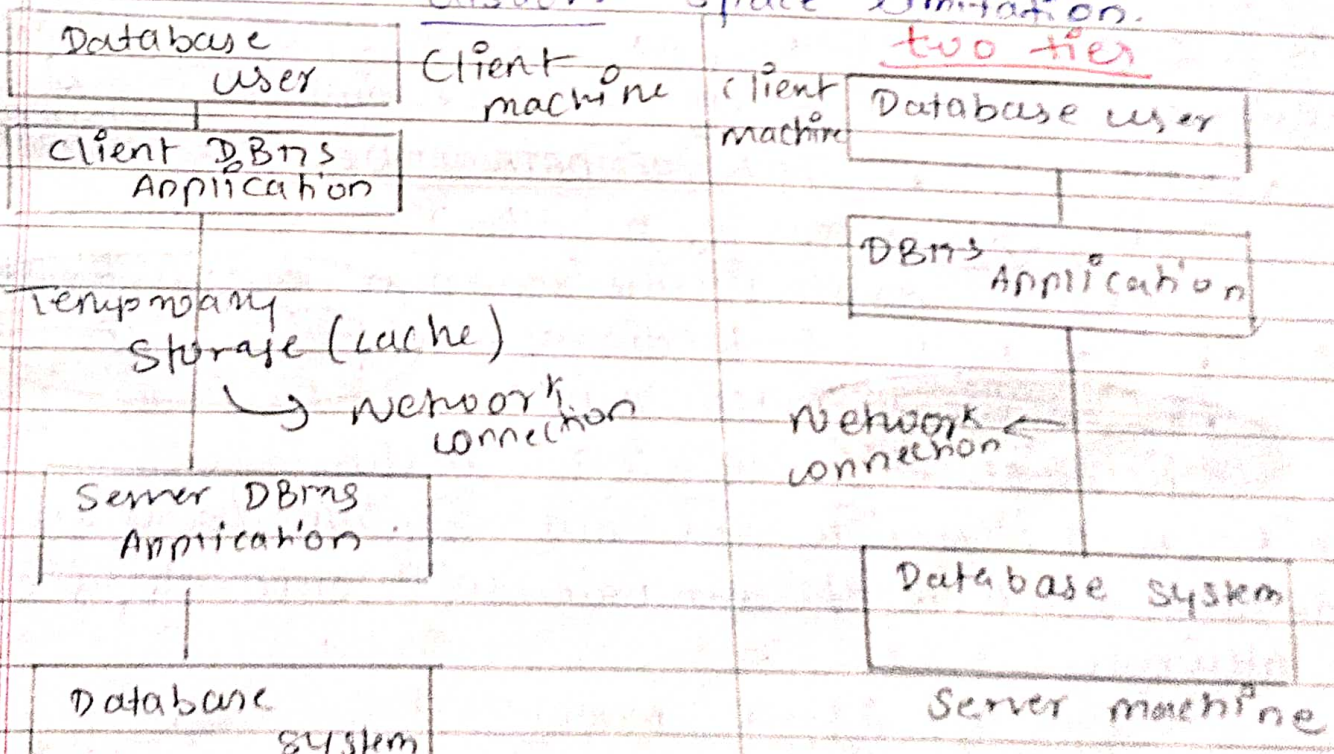
20/06/23 Ex: Every individual can work independently on the temporary storage area i.e. cache.

Single tier:- Adv: Secured and simple

two tier: Adv: fast access but limited no. of users.

three tier: Adv: temporary storage area provided. (more no. of user can work)

disadv: space limitation.



⊛ Levels OF DBMS Architecture

3 Levels.

External / View level	Conceptual / Logical level	Internal / Physical level
• Top level of the architecture. • Ex:- Blueprint or E.R. diagram	• The whole design of the database such as relationship among data, schema of data etc. are described in this level. • Maintained by DBA i.e. Database administrator Ex: Actual implementation	• This level describes how the data is actually stored in the storage devices. • Allocate space to data. • Lowest level of architecture Ex:- Maintaining.

⊛ DATA MODELS IN DBMS

It is a logical structure of database It describes entities, attributes, relationship among data, constraints etc.
(attributes → values; entities → column name)

- TYPES OF DATA MODELS:

(i) Object based logical Models
- Describe data at the conceptual and view levels.
- E-R. model / E-R diagram.
- object Oriented model.

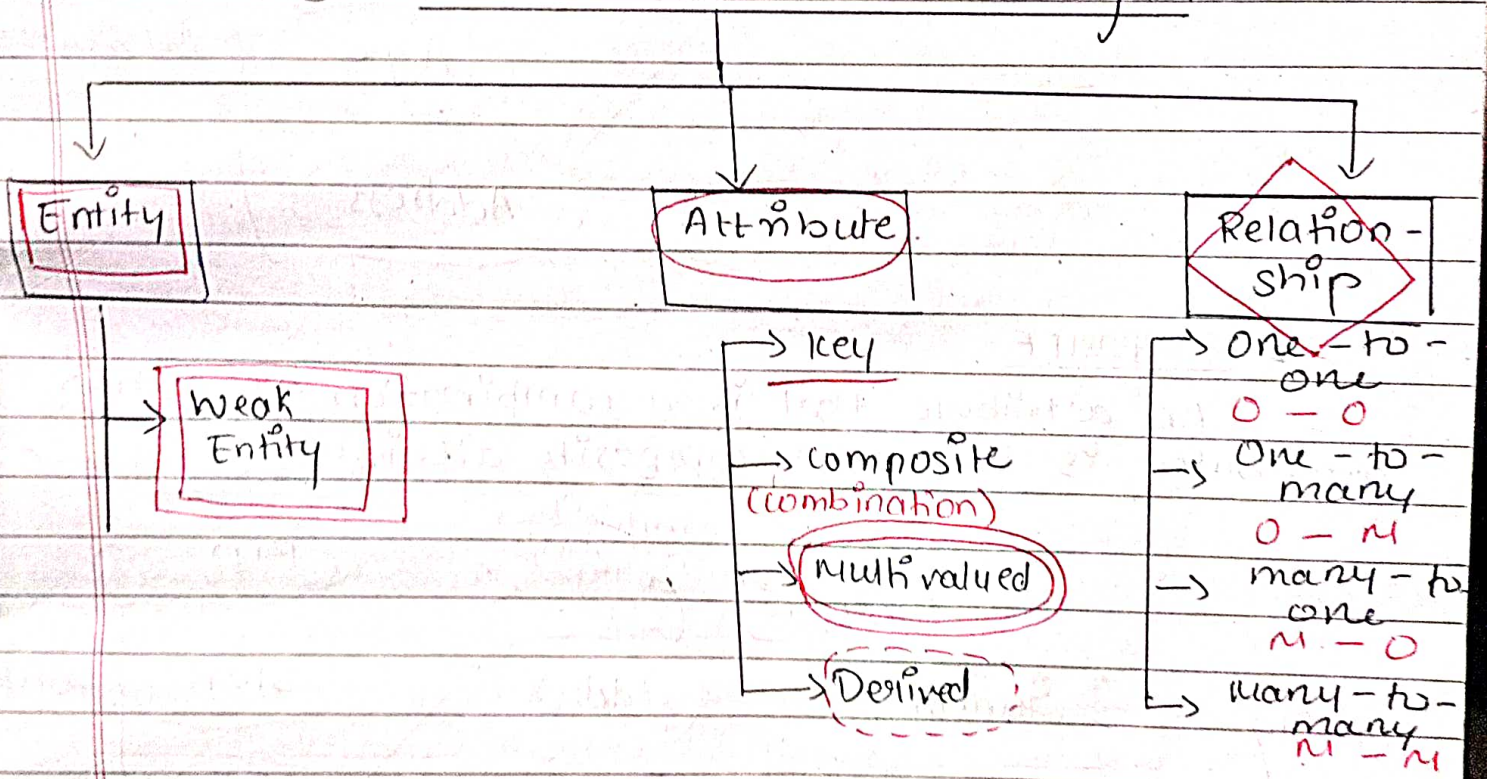
(ii) Record based logical models
Similar to the previous one but mainly focus on logical structure of database with records, fields and attributes.

⊛ E-R Model

- Stands for 'Entity - Relationship Model'.
- It describes the structure of a database with the help of a diagram, which is known as E-R diagram.

- It is a blueprint of database.
- Main components are:
  - Entity Set
  - Relationship Set
- An E-R diagram shows the relationship among entity sets.
- It shows the complete logical structure of the database.

✳ COMPONENTS OF E-R Diagram



✳ TERMINOLOGIES OF E-R DIAGRAM

- **ENTITY**

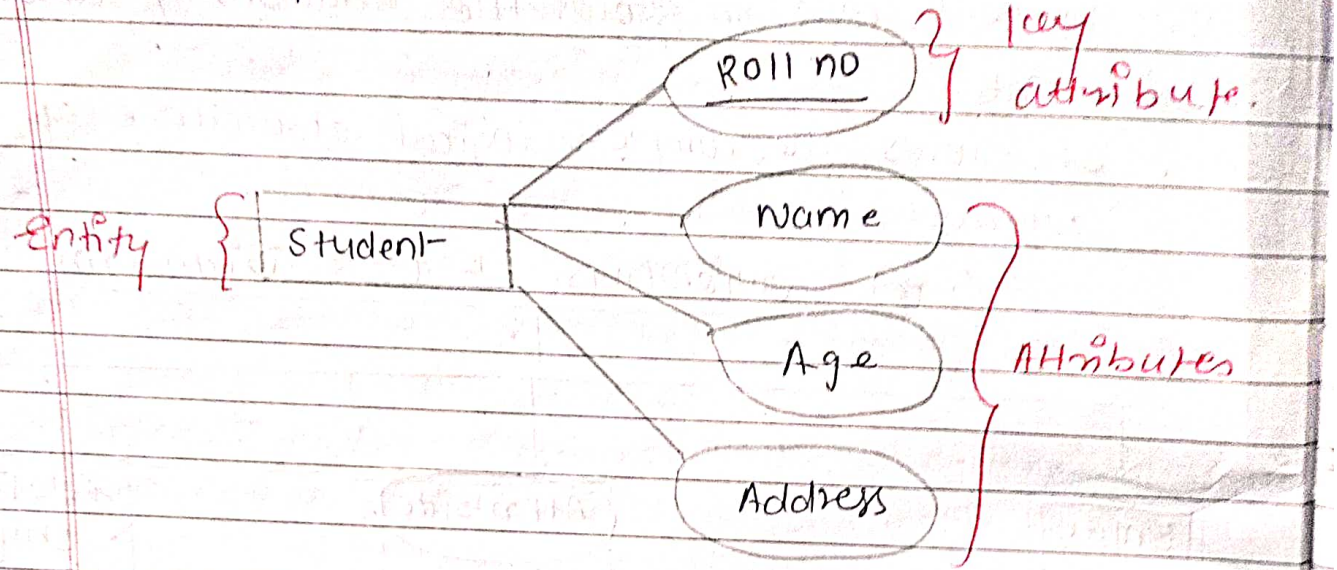
- An Entity is an object or component of data.
- An Entity is represented as 'rectangle' in an E.R diagram.

(i) Weak Entity :-

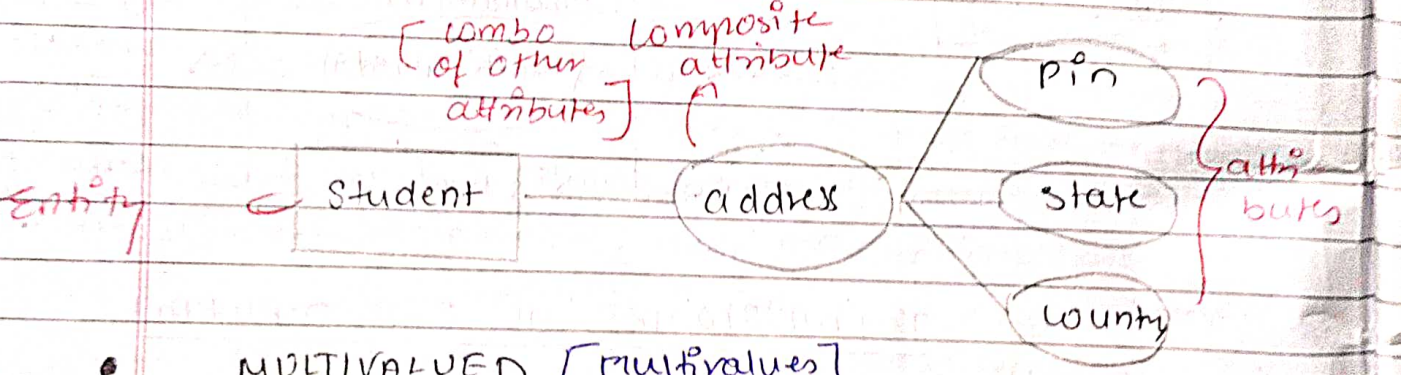
- An Entity that cannot be uniquely identified by its own attributes and relies on the relationship with other entity is called weak Entity.
- It is represent by 'double rectangle' in an E-R diagram. [depends on entity]

ATTRIBUTE

- **KEY**
  - A key attribute can uniquely identify an entity from an entity set.
  - Text of key attribute is 'underlined'.

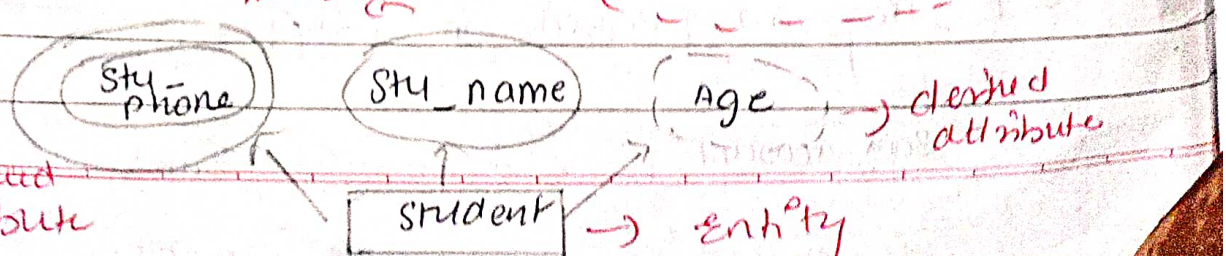


- **COMPOSITE**
  - An attribute that is a combination of other attributes is known as composite attribute.



- **MULTIVALUED [multivalues]**
  - An attribute that can hold multiple values is known as Multivalued attribute.
  - It is represented as 'double Ovals'.

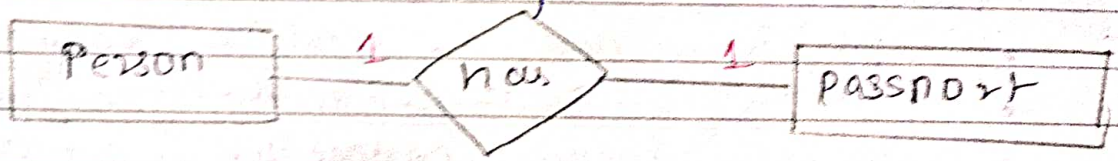
- **DERIVED (dependent on others) → only one value.**
  - A derived attribute is one whose value is dynamic and derived from another attribute.
  - It is represented as 'dashed oval'.



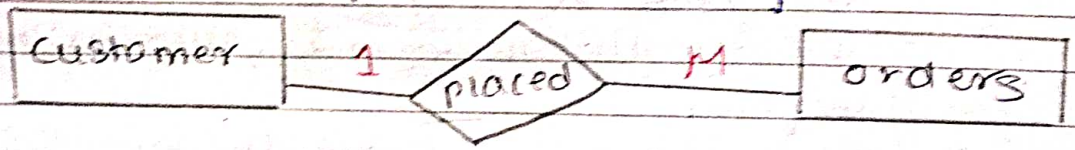
RELATIONSHIP

- It shows the relationship among entities.
- It is represented as diamond.

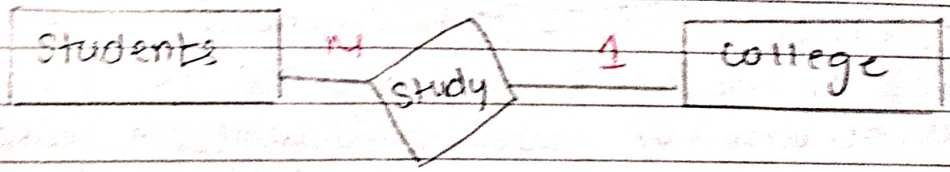
- **One-to-one**
  - when a single instance of an entity is associated with a single instance of another entity.



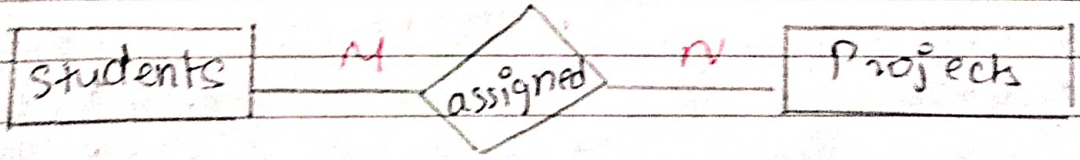
- **One-to-many**
  - when a single instance of an entity is associated with more than one instances of another entity.



- **Many-to-one**
  - when more than one instances of an entity is associated with a single instance of another entity.



- **Many-to-many**
  - when more than one instances of an entity is associated with more than one instances of another entity.



How examination module in college from student point of view

⊛ SQL [Structured Query language]

- Stands for Structured Query Language.
- It is a language to manipulate data with database.
- It is a non-case sensitive language.
- It is a non-procedural language.
- It is a Query language we cannot create frontend softwares with the help of SQL.

Non-Procedural language:- when language does not have strict pattern or constraints while writing the program is called as non-procedural language.

- The commands of SQL are broadly divided into 3 types:

DDL	DML	DCL
- Stands for 'Data Definition Language' - This category contains command associated with the structure of the table - Commands:- • CREATE (Used to create new db table) • ALTER • DROP	- Stands for 'Data Manipulation language'. - This category contains command associated with the values of the table - Commands:- • INSERT • UPDATE • DELETE • SELECT (To show records from the table)	- Stands for 'Data control language'. - This category contains commands associated with the administrative point of view - Commands:- • ROLLBACK [undo] • COMMIT [save] • SAVEPOINT [selective undo] • GRANT • REVOKE

only primary key is the constraint it can be used in only 1st column

04/07/23

(*) DATA TYPES IN SQL (ORACLE)

- **int** :- integer values [In oracle known as 'number']
- **varchar** :- variable characters [used to summon ents spaces if not occupied by actual data of a table.
- **char** :- character [char will occupy the space on the basis of initial data structure].
- **float** :- 2 parameter
  - size
  - no of decimal values.
- **date** :-

(*) CONSTRAINTS IN SQL (ORACLE)

(1) **Primary key**:-
- primary key doesn't allow 'duplicate' or 'blank spaces' values in a column.
- can be applied only on 1 column of a table. (mostly 1st column i.e unique)

(2) **not null**:-
- This constraint do not allow 'empty' values in a column but duplicate values are allowed.
- It can be applied on multiple column of a table.

(3) **unique**:-
- This constraint do not allow duplicat values in a column and only one 'empty' value is allowed.
- can be applied on multiple column of a table.

(4) default:- • It is used to defin default value of a column.
• can be applied on multiple column.
• During writing code, if more than one constraint is used then default has to be written in the last.

(5) check:- • It is used to allow range of values or specific values for a column.
eg: A column should contains values between 0-50 only.
eg: department column should contain IT, HR or marketing.

(6) foreign key:- (child Table)
• values of a table dependent on values of another table (parent table). This concept can be implemented with the help of foreign key. [only allowed with primary key]
• foreign key is represented with the help of 'references' keyword.
• It can be used on multiple table with the help of 'referenes' keyword.

Q Create a table college with following columns and conditions.
(i) Roll no:- It should contain 7 integer value. It should not allow duplicate or null values (primary key).
(ii) s_name:- should contain 15 varchar. This column should not contain empty values. (not null).
(iii) program:- should contain 10 char. should not contain empty value, values allowed in the column are: baf, bms, bscit, bcom
(check and not null)

(iv) marks :- should contain a integer with 2 decimal pts. values allowed for the column is between 1 - 100 (check).

(v) address :- should contain 15 varchar, should not contain empty values, default value is 'mumbai' (default)

(vi) D.O.A :- should contain value in date formate.

char, varchar, date ⟹ values enclosed always in single quotes.

Program:

```
create table college
(
roll_no int(7) primary key,
s_name varchar(15) not null,
program char(10) not null check(program in('baf', 'bms', 'bscit', 'bcom')),
marks float(8,2) check(marks between 1 and 100),
address varchar(15) not null default 'mumbai',
DOA date
);
```

not to add size in oracle during program practical.

in :- • act as 'OR'
• can't take range
• always in bracket

between :- • only with integer
• 'AND' is compulsory
• enclosed with starting & ending value.

Commands

11/02/23

- **DESC**
- To see the structure of the table, 'desc' command is used. (includes - name of column, - datatype of column, - size of the column)
- desc stands for 'describe'.

Eg command:- `desc college;`

name of the respective table.

- INSERT

- To insert the values in the table OR to insert new row in the table 'INSERT' command is used.

Ex command:- insert into college values (1, 'Ram', 'Bat');
↳ name of table (unconstant)

- SELECT

- To show the values from the table 'SELECT' command is used.

Ex command :- Select * from college;
↳ for showing pure table (not missing anything)

- COMMIT

- To save the inserted value 'COMMIT' command is used.

Ex command :- commit;

- TO INSERT VALUES IN SPECIFIC COLUMN

Ex command :- insert into college (roll_no, s_name)
values (4, 'Mohan');
tab name
[name of the rows into which want to insert; write as it is]
[values wished to insert]

- TO SHOW SPECIFIC COLUMN AS OUTPUT

Ex command :- select roll_no, program from college;
[column which you wish to see] [table name]

- TO ACCEPT VALUES FROM USER [bind operator]

Ex command :- '&' operator is used to accept values.
insert into college values (&roll, '&name', '&prog');

- It will ask for roll, name & prog.
- If your datatype is char, varchar, date then previous only mention them in single quotes otherwise during entering write it in single quotes.

- TO EXECUTE LAST RECENT COMMAND
  - '/' Operator is used to execute the recently previous command.

- ALTER COMMAND
  - TO modify the structure of the table 'ALTER' command is used.
  - We can do following things using alter comman
    - with the help of alter command we can add a new column in existing table
    - Delete the column from existing table
    - Change datatype and size of the column.
    - change name of the column
    - Change name of the table
  - Few subcommands help alter to work as follows:
    - add

Q Add a new column subject in table 'college' with 10 characters.

command: alter table college add subject char(10);

- drop

Q Remove program column from table college.

command: alter table college drop column program;

- rename
- modify

Q change the size of subject column to 20 varchar

command: alter table college modify subject varchar(20);

- change [can be used in place of modify]

Q change the name of s_name column to f_name.

command: alter table college change s_name f_name varchar(20);

- rename

Q Change the college table name to tolani_college

Comm: alter table college rename tolani_college;

OR

alter table college rename to tolani_college;

Q Insert 3 rows in table 'college' values are as follows:-

1st row: 1, Ram, BAF

2nd row: 2, Shyam, bcom

3rd row: 3, Rohan, bscit

Code:-

```
insert into college values (1, 'Ram', 'BAF');
insert into college values (2, 'Shyam', 'BCOM');
insert into college values (3, 'Rohan', 'BSCIT');
commit;
Select * from college:-
```

O/P:

roll_no	S_name	Program
1	Ram	BAF
2	Shyam	BCOM
3	Rohan	BSCIT

✍ To insert values in specific columns

```
insert into college (roll no, s_name) values
(4, 'Mohan');
```

O/P:

roll_no	S_name	Program
1	Ram	BAF
2	Shyam	BCOM
3	Rohan	BSCIT
4	Mohan	NULL

✍ To show specific column as output

```
select roll_no, program from college;
```

roll_no	Program
1	BAF
2	BCOM
3	BSCIT
4	NULL

14/07/2023

(*) TO MODIFY THE VALUES OF A TABLE

- To modify the value of a table update command is used.

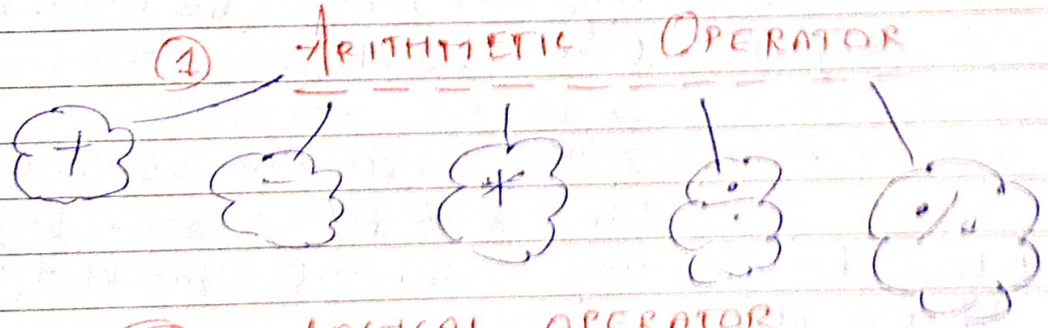
(*) TO GIVE THE CONDITION

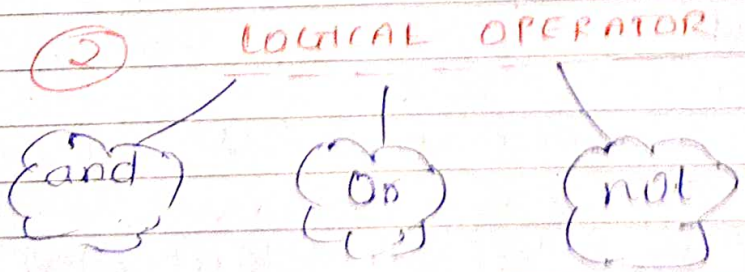
- TO give the condition in sql where clause can be used.

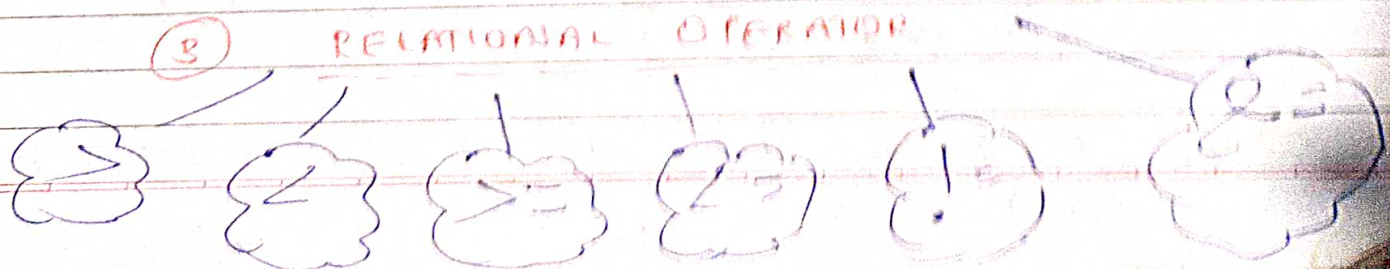
Q As a 'sub' ROll NO. 1 & 3 ⇒ I.P
ROll NO. 2 ⇒ C.A
ROll NO. 4 ⇒ I.O.T

Soln :- (i) update tolani_college set subjects = 'I.P' where roll no. in (1,3);
(ii) update tolani_college set subjects = 'C.A' where roll no = 3;
(iii) update tolani_college set subject = 'IOT' where roll no = 4

OPERATORS IN SQL

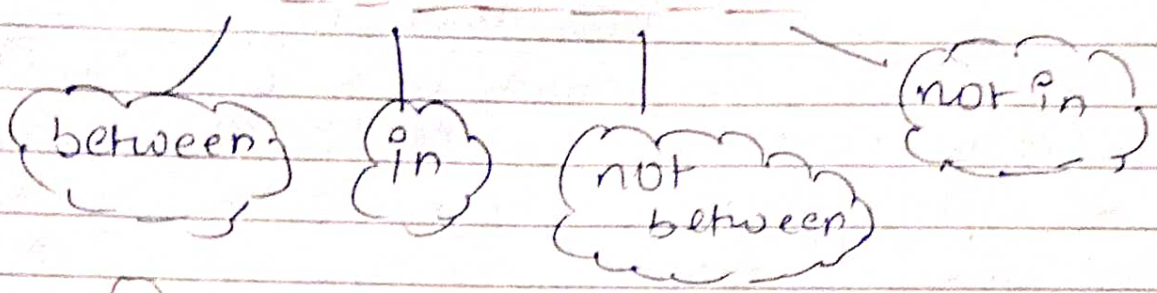






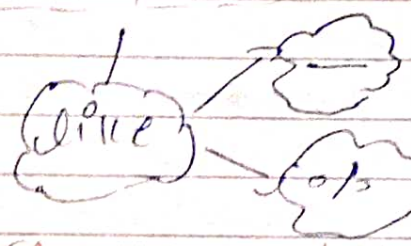
EXCLUSIVE SQL OPERATORS

(4) RANGE OPERATOR



(5) MATCH PATTERN

imp



underscore : one character

percent : '0' or more character

when Que includes 'Except' word then use not in.

Q. Rohan should get 60 marks.

⇒ update tulani college set marks = 60 where f_name = 'Rohan';

Q. 65 marks should be given to the student who belong to WP & JOT subjects.

⇒ update tulani college set marks = 65 where subject in ('WP', 'JOT');

Q. Inc the marks with 5 for all the students who is having IP as a subject.

⇒ update tulani college set marks = marks + 5 where subject = 'IP';

Q. Change the sub name to WP whose marks are between 75 to 85

⇒ update tulani_college set subject = 'WP' where marks between 75 and 85;

Q. Assign marks 56 to the students with the roll no 3 & 4.

update tolani_college set marks = 55 where roll_no not in (3,4);

22/07/23

4. increase the marks by 5 of all the students whose marks is less than 70

⇒ update demo set marks = marks + 5 where marks < 70;

5. Show all the roll no & marks of all the students who belongs to DBMS or DS subjects.

⇒ select Roll, marks from demo where subject in ('DBMS', 'DS');

6. Show the name of the student who doesn't belongs to DS subjects

⇒ select name from demo where subject != 'DS';

7. Show all the records from demo table whose name ends with 'z'

⇒ select * from demo where name like '%z';

8. Show all the records from demo, where first & last character is 'a'.

⇒ select * from demo where name like 'a%a';

9. Show all the records from demo table whose second last character is 'b'

⇒ select * from demo where name like '%b_';

10. Shows all the records from demo where name is containing 'p'

⇒ select * from demo where name like '%p%';

11. Show all the records from demo where second & secondlast character is 't' and 'p' is also appearing in the name.

⇒ select * from demo where name like '_t%p%t_';

- To remove record from the table 'delete' command is used.
- To undo the record 'ROLL BACK' command is used.

Q Remove all the records from the table whose marks is more than equals to 90.

⇒ delete from demo where marks >= 90;

Q delete all records from demo table where marks is more than equals to 85 and they belongs to DBMS or DS subjects.

⇒ delete from demo where marks >= 85 and subject in ('DBMS', 'DS');

- To sort the data in SQL 'order by' clause can be used.
  - It uses 2 keywords to arrange the data in ascending / descending order.
    'asc' → ascending order [By Default]
    'desc' → descending order.

Q show all the records from demo table in asc. order of marks column.

⇒ select * from demo order by marks asc; (OR)
select * from demo order by marks;

Q show all the records from demo table in desc. order of name column.

⇒ select * from demo order by name desc;

Q show all the records from demo table who belongs to DBMS or CN subjects in asc order of marks.

⇒ select * from demo where subject in ('DBMS', 'CN') order by marks.

• MULTIPLE SORTING

Q) Show all the records from demo table in asc order of subjects within the same subject marks should appear in desc order.

⇒ Select * from demo order by subject, marks desc;

Q) Show all the records from demo in desc order of subject column within the same subject name should also appear in asc order.

⇒ select * from demo order by subject desc, name desc;

• FUNCTIONS IN SQL

(1) aggregate function:

- max, min, count, sum, average (avg)

⇒ Select max(marks) "maximum marks" from demo;
⇒ Select min(marks) "minimum marks" from demo;
⇒ select sum(marks) "Sum of the marks" from demo;
⇒ Select avg(marks) "Average marks" from demo;

'dual' is a temproary table in oracle used to execute temproary queries

- **column alias**:- Some other heading as output.
  - If space is there between the heading then "double Quotes" is required. Ex:- "highest marks".
  - If space is not there between the heading then "double Quotes" not required. Ex:- Average.

(2) math function:

- round:- Takes 2 parameter.
  - 1st no:- 'number'
  - 2nd no:- 'position'.

⇒ select round(avg(marks, 1) "average marks" from demo;
⇒ Select round (234.4536, 3) from dual;

25/07/23

- TO SHOW THE DATA ON THE BASIS OF GROUP 'Group by' clause is used.

[imp pt.] 2 prerequisite for Group By clause:

(i) Atleast 1 column should contain duplicate value
(ii) one of the aggregate function must be there.

Q) show subject wise highest marks.

⇒ select subject, max(marks) "highest marks" from demo group by subject;

Q) show subject wise no. of students

⇒ select subject, count (roll) "no. of students" from demo group by subject;

- TO provide condition with groupby clause 'having' clause can be used. [Exclusive for Group by clause].

Q) show subject wise average marks where average marks is more than 60

⇒ select subject, avg(marks) "average marks" from demo group by subject having avg(marks) > 60;

- TO generate autolist in SQL 'sequence' command is used.
- Following are the attributes of sequence:-

(i) start with - This attribute can change by default starting position of any sequence.
(ii) increment by - This attribute is used to change the by default value of incrementing.
(iii) maxvalue - This attribute is used to show the highest or maximum value or limitation of any sequence.
(iv) cycle - This attribute is used to repeat the cycle from minimum value to maximum value.

(v) nocycle - This attribute is used for no repetition of any cycle.

- Normal program for Sequence:-

① Create sequence S1;
insert into demo values (S1.nextval, 'hgf', 67, 'CN');

- incrementing & cycle program:-

⇒ create sequence S2 start with 12 increment by 2 maxvalue 20 cycle;

Q Write a SQL command to create a table 'college' with following columns & conditions. [SQL package]

1st:- Std_ID:- contain 5 int values & should not duplicate and empty values.

2nd:- S_name:- should contain 10 varchar & column should not be empty.

3rd:- program:- contain 10 char & column should only contain values as BSCIT, BCOM, BAF.

4th:- Subject:- contains 10. char, should not contain empty values.

5th:- marks:- contains 7 int with 2 decimal pts. Values allowed for the column is between 0 - 100

Create an appropriate sequence to insert student_id when the no. starts with '1' & stops on '25'.

Structure:-

```
Create table college
(
Std_ID int(5) primary key,
S_name varchar(10) not null,
program char(10) check (program in ('BSCIT',
'BCOM', 'BAF'))
subject char(10) not null,
marks decimal(7,2) check (marks between
0 and 100)
);
```

Column allowing no in int then use number

Sequence :-

⇒ Create sequence s1 start with 1 maxvalue 25;

Use the following table for next que.

Std_ID	s_name	program	Subject	Marks
1	Sonam	BAF	AIC	78
2	Shilpa	BIOM	AIC	90
3	Shivam	BSCIT	CCT	98
4	Amrita	BSCIT	CCT	78
5	Amit	BIOM	TAX	65
6	Roshan	BAF	TAX	43
7	Ramesh	BSCIT	CCT	34

Accepting values from user:-

⇒ insert into college values (&ID, '&name', '&prog', '&Sub', &Marks);

(4) Write SQL command for following:-

Q Add a new column result in college table with 10 varchar.

⇒ alter table college add result varchar(10);

Q Change the name of subject column to 'course'.

⇒ alter table college rename column subject to course;

Q Students who have marks more than 80 should get the result as 'PASS' other should get 'FAIL'.

⇒ (i) update college set result = 'PASS' where marks > 80;

(ii) update college set result = 'FAIL' where marks ≤ 80;

Q Increase marks by 5 for the students who belongs to accounting subject.

⇒ update college set marks = marks + 5 where course = 'AIC';

Write SQL commands for the following:-

Q Show all the records from college whose student name starts with 's' and ends with 'a'. (24)

=> Select * from college where s_name like 's%a';

Q Show s_name and program name who belongs to bscit or bcom program.

=> Select s_name, program from college where program in ('BSCIT', 'BCOM');

Q remove all the records from the table whose marks is less than 40.

=> delete from college where marks < 40;

Q Show all the student's ID whose name starts with 'a' & second last character is 'i'.

=> select std_ID from college. where s_name like 'a%.i_';

Write SQL commands for the following:-

Q show program wise no. of students from clg table.

=> Select program, count(std_ID) from college group by program;

Q show total marks subject wise.

=> Select course, sum(marks) from college group by course;

Q show program wise highest marks in asc. orde

=> Select program, max(marks) from college group by program order by max(marks);

Q show subject wise avg. marks where avg. marks is more than 65 in asc. order.

=> Select course, avg(marks) from ~~tolani~~ college group by course having avg(marks) > 65 order by avg(marks);

Q show no. of students in clg table.

=> Select count(std_id) from ~~tolani~~ college;

01/08/23

SUB QUERY

- A Query inside another Query is known as Sub Query.
- The Query given inside bracket is known as 'Inner Query'
- The Query given outside bracket is known as 'Outer Query'
- Output of Inner Query become condition for Outer Query & final Output always comes from Outer Query.
- Sub Query is required when the single output should be merged with group function.

Q Show the name of the students whose marks are more than average marks of the table.

⇒ select name from demo where marks>(select avg(marks) from demo);

Q Show the name of the students whose marks is Equal to the minimum marks of table.

⇒ select name from demo where marks=(select min(marks) from demo);

JOIN IN SQL

- When the record has to be fetched from multiple table and need to be represent as one output it can be done with the help of join.

Types of Join

- Inner join
- Outer join
  - left outer join
  - Right outer join
  - full outer join

INNER JOIN

- used to fetch data from multiple tables.
- represented by "=" sign one common column should be there in all the tables used in inner join.

Table Alias:- To give the shortform name to table first give space & write the new name. Ex: lib 'l', demo 'd'.
table alias.

Q: Show students roll no & name along with the book name using inner join.

⇒ select l.roll, name, book_name from demod, lib l where d.roll = l.roll;

Q: Show the students roll no, subject name, marks & lib fees for all the students.

⇒ select d.roll, subject, marks, lib_fees from demod lib l where d.roll = l.roll;

LEFT OUTER JOIN

left Outer join returns all rows from the left (first) table specified in the 'on' condition and only those rows from the right (second) table where the join condition is met will come as output.

Q: ⇒ select demo.roll, name, book_name from demo left outer join lib on demo.roll = lib.roll;

RIGHT OUTER JOIN

Right Outer join returns all the rows from R.H.S

Table specified in the 'On' condition and only those rows from the outer table where the join condition met will come as O/P.

~~Eg~~

Q:- ⇒ Select demo.roll, name, book name from demo right outer join lib on demo.roll = lib.roll;

FULL OUTER JOIN

Full outer join returns all the rows from L.H.S & R.H.S Table it keep the places empty where the join condition is not met.

Q:- ⇒ Select demo.roll, name, book_name from demo full outer join lib on demo.roll = lib.roll;

SET OPERATORS IN SQL

These are special types of operators which can be used to combine the queries in the some result. Set Operators are :-
Union, Union all, intersect, minus.

- Prerequisites to use set Operators :-
(i) The no. & order of columns must be the same.
(ii) Data type must be compatible.

(a) UNION

⇒ It is used to combine the result of two select statements.
⇒ duplicate rows will be eliminated from the result obtained using union Operator.

(b) Union all

⇒ It combines all the records from both the queries.

⇒ duplicate rows will not be eliminated from the result obtained using union all operation.

(c) intersect

⇒ used to combine two select statements but it returns the record which are common from both tables.

(d) minus

⇒ It displays the rows which are present in the first query & absent in the second query with no duplicates.

04/08/23

when you want to create a new table identical to existing table [identical columns, datatype & sequence]. the following code needs to be followed.

⇒ Create table admin as select * from faculty where 1>2; [the condition should always be wrong; same can be applied]

Faculty Table

Emp_ID	Ename	Address	Salary
1	Amit	Mumbai	20000
2	Navin	gujarat	23000
3	Saima	mumbai	21000
4	aftab	nagpur	30000

Admin Table

Emp_ID	Ename	Address	Salary
1	mukesh	nagpur	10000
4	aftab	nagpur	30000
6	ramesh	pune	22000

Program for Union

- Select * from faculty union set * from admin;

[it will eliminate duplicates ex:- aftab]

EmpID	Ename	ADDRESS	SALARY
1	amit	mumbai	20,000
1	mukesh	nagpur	18,000
2	navin	gujarat	23,000
3	saima	mumbai	21,000
4	aftab	nagpur	30,000
6	ramesh	pune	22,000

Program for union all Operator

- Select * from faculty union all select * from admin;

[will not eliminate duplicate].

Program for intersect Operator

- select * from faculty intersect select * from admin;

Emp-ID	Ename	Address	salary
4	aftab	nagpur	30,000

Program for minus Operator

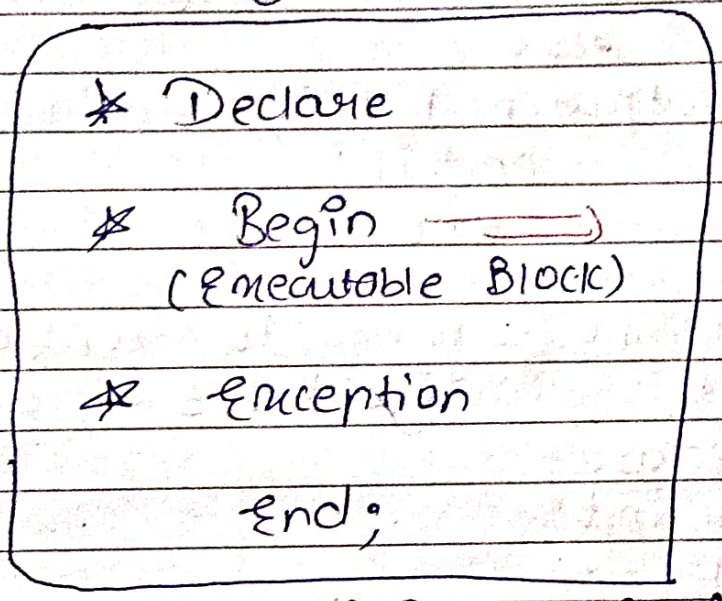
- Select * from faculty minus select * from admin;

- Select * from admin minus select * from faculty;

PLSQL

- PL ⇒ Procedural language
- SQL ⇒ Structured Query language.
- It is a case-sensitive language.

3 Main Blocks.



Main block.

[Collection of classes & functions]

* To show the O/P in PLSQL following package and function can be used.

dbms_output.put_line()

put_line(): to print the output

* Each day for PLSQL code, activate dbms output package with code:

set serveroutput on

/ is used to execute.

Q. Write a PLSQL program to define an integer variable with a value & show the square of given nos.

⇒

```
declare
n int;
begin
n:= &sum;
dbms_output.put_line(n*n);
end;
```

Q Write a PLSQL program to accept a no. from user & check it is even or odd.

```
declare
x int;
begin
x:= &value;
if(x mod 2 = 0) then
dbms_output.put_line('even no');
else
dbms_output.put_line('odd no');
end if;
end;
```

Q Write a PLSQL program to accept a no. from user and check it is divisible by 3 & 5 or not.

```
declare
x int;
begin
x:= &value;
if(x mod 3 = 0 and x mod 5 = 0) then
dbms_output.put_line('divisible');
else
dbms_output.put_line('not divisible');
end if;
end;
```

Q Write a PLSQL program to accept no. from user if the no. is between 80-100 => 'O' grade, 60-79 => 'A' grade, 40-59 => 'B' grade. Others are fail.

```
declare
x int;
if(x>=80 and x<=100) then
dbms_output.put_line('O grade');
elsif (x>=60 and x<=79) then
```

```
dbms_output.put_line ('A grade');
elsif (n>=40 and n<=59) then
dbms_output.put_line ('B grade');
    else
dbms_output.put_line ('fail');
    end if;
    end;
```

f) Write a PL/SQL program to accept salary from user & store HRA [Housing rent allowance] value on the basis of following condition :-

(i) more than 150000 HRA value => 15% of sal.

(ii) sal between 100000 – 150000 HRA value => 12% of sal.

(iii) Otherwise 5% of sal.

=>
```
declare
n int;
begin
n:=&sal;
if (n>150000) then
dbms_output.put_line(n*0.15);
elsif (n>100000 and n<150000) then
dbms_output.put_line(n*0.12);
    else
dbms_output.put_line (n*0.05);
    endif;
    end;
/
```

08/01/23

LOOPS IN PL/SQL

- Syntax of Basic loop in PL/SQL

```
LOOP
statement to be executed
[exit when n>=10;]
end LOOP;
```

Ex:-

```
declare
 x int;
 begin
 x:=5
  loop
dbms_output.put_line(x+x);
   x:=x+2;
    exit when x>=10;
      end loop;
       end;
```

/

o/p: 25
49
$1

Q. Write a PLSQL program to accept a no. from user and show its table using basic loop.

=>

```
declare
 x int;
 y int;
  begin
   x:=4;
   y:=1;
    loop
dbms_output.put_line (x*y);
   y:=y+1;
    exit when y>10;
      end loop;
       end;
```

/

CURSOR IN PLSQL

(*) To Fetch the records from SQL Table to PLSQL 2 commands are Available!

1) Select into [It can fetch only one record at a time]
2) Cursor [multiple records can be fetched together]

(*) Following 2 pre-defined attributes assign the data type of a column to a variable.

1) %type = it can fetch data type of 1 column at a time.

2) %rowtype = It can fetch data type of all the columns to a variable.

Q Write a PL/SQL command to show all the details of employee ID accepted by user

⇒
```
declare
eid faculty.emp_id%type;
n faculty.ename%type;
addr faculty.address%type;
sal faculty.salary%type;
begin
select * into eid, n, addr, sal from faculty where
                emp_id = &eid;
dbms_output.put_line(eid||' '||n||' '||addr||
                sal);
end;
/
```

Note:- || (pipe) ⇒ Concatenation Operator.

11/08/23 * To fetch Multiple Records from the table we need cursor in PL/SQL which can be defined in following steps

Step 1:- Declaration of cursor.
Step 2:- Open the cursor
Step 3:- Fetch the Reword in the cursor
Step 4:- Close the cursor.

Q Write a PL/SQL program to create a cursor C1 who fetch all the records from faculty where salary is more than 22,000. Show the employee

name, their salary and bonus value (5% of sal).

⇒
```
declare
var1 faculty %rowtype;
cursor c1 is select * from faculty where
  salary > 22000;
    begin
      open c1;
      loop
    fetch c1 into var1;
  exit when c1%notfound;
dbms_output.put_line (var1.enomell' '||
var1.salary ||' '|| var1.salary + 0.05);
    end loop;
      close c1;
        end;
        /
```

⇒

navin	23,000	1150
aftab	30,000	1500
aftab	30,000	1500

ATTRIBUTES

1) %notfound = If the data is not found in the cursor it will exist from the specific block.
2) %found = It stay in the specific block when the record is found in the cursor.
3) %rowcount = It exist from the specific block when it reach to the total number of record given the condition.
4) %isopen = It will check ensure whether the cursor is open or not.

Q. Write a PLSQL command to show all the even numbers between two nos accepted from user.

⇒
```
declare
x int;
y int;
begin
x := &num1;
y := &num2;
for a in x..y loop // for a in x..y by 2 (to increment the value)
if (a mod 2 = 0) then
dbms_output.put_line(a);
end if;
end loop;
end;
/
```
for loop

17/08/23

PROCEDURE IN PLSQL:

Procedure is a block of code which compile ones, It can be executed multiple times on the basis of user's requirement.

SYNTAX:

```
create or replace procedure procedure_name()
                                is block
end procedure_name
```

Q. Write a PLSQL Program to create a procedure P1 which should accept an int. value from the parameter and show the square of given no. whenever procedure execute

⇒
```
create or replace procedure p1(x int)
is
begin
```

```
dbms_output.put_line(x*x);
        end;
        /
⇒)  exec P1(9);
```

Note:- i) exec:- command is used to execute o/p in procedure.
ii) show errors:- This command is used to show errors in a particular code.

Q) Write a PLSQL code to create a procedure result which are roll no, & 3 subject marks from the parameter from student table show the marks along with the result on the basis of following condition
1) (IF the student is passing (≥40) in all the subjects, the result should be pass, otherwise the result should be fail).

```
⇒)  create or replace procedure result,
                    (r demo.roll%type)
        is
    m1 demo.DBMS%type;
    m2 demo.DS%type;
    m3 demo.AM%type;
      begin
select DBMS, DS, AM, INTO m1, m2, m3 from
demo where roll = r;
if(m1 >= 40 and m2>=40 and m3>=40) then
then dbms_output.put_line('pass'||' '||'total
marks' || (m1+m2+m3));
else
```

```
dbms_output.put_line('fail'||' '||'total marks'
        ||(m1+m2+m3));
    end if;
    end;
    /
```

HW

Q Write a PLSQL program to create a procedure subject_result which accept subject name from the parameter and show all the students name and their marks who belong to that specific subject.

25/08/23

SWITCH CASE

Q Write a program to accept a number from user and show the name of week day on the basis of given number.

=>

```
declare
n int;
begin
n:=&a;
case n
when 1 then dbms_output.put_line('Monday');
when 2 then dbms_output.put_line('Tuesday');
when 3 then dbms_output.put_line('Wednesday');
else
dbms_output.put_line('Invalid day');
end case;
end;
/
```

---

* CREATING A PROCEDURE

⇒ create or replace procedure p2(n int)
is
y int;
begin
y := n+3;
dbms_output.put_line(y);
end;
/

* EXECUTABLE THE OUTPUT

⇒ exec p2(3);

/

* PROGRAM

⇒
declare
n int;
begin
n := &n;
case n
when 1 then p2(n);
else
dbms_output.put_line('invalid');
end case;
end;
/

30/08/23

* PACKAGE

⇒ collection of procedures & functions are called as package.

⇒ package can be declared in 2 steps
i) Declaration & ii) Package Body.

Q Write a PLSQL language Program to create a package pack1 with following procedures:-

1) name of the procedure is P1 should contain 1 integer parameter & show the square value of the given parameter.

2) Procedure name p2 one integer parameter & show the factorial of given no.

⇒

```
set serveroutput on
create package pac1 is
    procedure p1 (n int);
    procedure p2 (n int);
    end pac1;
    /
create or replace body pac1 is procedure p1(n int)
    is
    begin
dbms_output.put_line (n*n);
    end p1;
procedure p2 (n int)
    is
    y int;
    begin
    y:=1;
    for z in 1..n loop
        y:= y*z;
        end loop;
    dbms_output.put_line(y);
        end p2;
        end pac1;
        /
```

→ exec pac1.p1(4);
16

→ exec pac1.p2(4);
24

Q. Create a package pac2 with following procedure:- procedure p1 should accept 2 integer & show all even no between them procedure p2 should accept 2 integer & show all prime no. between them.

⇒

```
Set ServerOutput on
create package pac2 is
  procedure p1(x int, y int);
  procedure p2(x int, y int);
  end pac2;
  /
```