2/2/2021 Tuesday

01. Introduction to database Concept

Data = statistics → it is raw and unprocessed.
eg = name, class, marks etc.

information = when data is processed. "Record is also information."
eg, pass or fail Table etc.

Database : an organized collection of data and information or interlated data collected at one place.

DBMS = collection of interrelated data and set of program to access data.
eg: MySql, Oracle

Application = Company information, Account information, manufacturing etc.
Banking, finance transactions, Universities, telecommunication.

3/2/2021 wednesday

* Characteristics or Database system over File management system

* Avoid Data redundancy (same data multiple time available) and inconsistency (same data having ~~one~~ two different values i.e roll No.1 xyz
roll No.1 abc

* Acessing of data = easily acessable.

* Data isolation = data saved in standard format/ ~~structured~~ structure.

* Data integrity = Security (constraints, grant commond, data not null)

* Atomicity = It shows either all transactions executed successfully or not.
eg: ATM widrawal

* Concurrent - access to data =
multiple user accesses data same time.

* Security = Authentication (user name & password)

* View of data or part of data.

Data abstractions = the system hide certain details of how data are stored & maintain.

1) physical level = how data are actually stored.

2) logical level = what data are stored and what is relationship exist among that data.

3) View level = describe only part of the entire database.

Instances & Schemas

Collection of information stored in the database at a particular moment is called an instance.

Overall design of database is called schemas. (only attributes / title)

physical schemas describe the database designed at physical level.
logical -11- ——— 11 ——— 11 ——— logical level.
view level schemas having several schema.

Data model = way to describe the design of a database at the physical, logical and view mode. (both attributes and its values)

Categories of data model :

1) Relational Model = uses collection of table which contain both data and relationship among that data.

2) Entity - Relationship Model (ER) = It uses collection of basic objects (entities) and relationship among this object.
eg. Name, age etc.

3) Object Based data model = OOP

4) Semistructured Data model = Combination of programming and standard syntax.
eg: (XML) = Extensible Markup Language.

04/02/2021
Thursday

Architecture of Database

* naive users (existing system) (agents, tellers etc.) (eg. ATM, booking tickets)
* application programmers (write program)
* Sophisticated users
* database administrators → install, update and maintain database

* DDL interpreter → decide what data, datatype ect use
* DML queries → delete, insert data etc.
* DML compiler & organizer → Compile
* application program object code.

Connected

} query processor

* buffer manager = extra storage
* file -11 → allocation of file
* authorization & integrity manager → permission
* transaction manager → handling all transaction.

} storage manager

* data → info
* statistical data → calculated data
* indices → provide index to data
* data dictionary → database related words.

} disc storage

* Functional Components of database mainly divided into storage manager and query processor.

storage manager - store all data.
query processor - it helps database system simplify and facilitate access to data.

* Database users

- Application programmers - They are the developers who interact with the database by means of DML queries (written in C, C++, java etc).
- Sophisticated users - They are database developers who writes SQL queries for select / insert / delete / update data. doesn't use any application or program to request database.
- Specialized users - Same as Sophisticated users but the develops Complex programs.
- Database administrator - install, update and maintain database.
- Stand-alone users - personal use. having readymade database.
- Native users - use existing application
  eg: online library, ATM's, ticket booking etc.

* Query processor - it includes

- DDL interpreter - (Data Definition Language)
-interprets DDL statments & record the definition in data dictionary.

- DML compiler - translate DML statements in query language into an evaluation plan consisting of low level instructions.
  ↓
- Query evaluation engine - execute low-level instruction.

* Storage Manager -
  provide interface between low level data stored in database and application program and queries. it includes

- Authorization and integrity manager - Test satisfaction of integrity constraints and checks authority to access data.
- Transaction manager - ensures all transaction.
  Transaction = Collection of operation perform a single logical function in database.
- File manager - manage allocation of space on disk storage.
- Buffer manager - it enables the database to handle data size that are much ~~higher~~ larger than size of main memory.

* Functions of Database administrators (DBA)
- Installing & upgrading DBMS Servers.
- Design and implementation
- performance tuning
- Migrate database servers
- Backup and Recovery
- Security
- Documentation

* Types of DBA

- Administrative DBA : installing and maintaining DBMS servers.
- Development DBA : creating queries and procedures for requirement.
- Database Architecture : creating and maintaing the users, roles, access rights, tables, view, constraints, indexes.
- Data warehouse DBA : maintain Collection of data and procedures from various sources in data warehouse.
- Application DBA : bridge betⁿ application program & database.
- OLAP DBA = installing and maintaining database in OLAP system only.

* Data Independence : → data abstraction (hide info..)
It is the ability to modify schema at lower level without alteration at a higher level.

- goal → data independent from user.

- physical level - describe where and how data is stored
  - lowest / internal level
  - Control by DBA.
- Conceptual level - logical level
  - how data stored and structures use
  - Control by database designer.

- External level - view level / higher level
  represent data to the user
  control by interface designer.

5/02/2021
Friday

02. Entity Relationship Data Model

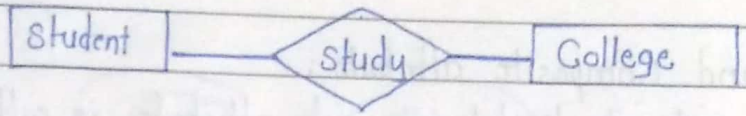
- ER Diagram is describing structure of database
- ER-Model is blueprint of database

Component of ER diagram -

1. Entity (symbol ▭)
2. Attribute
3. Relationship

1. Entity - object or component of data. Symbol of entity → Rectangle

eg :



- Weak Entity - An entity that cannot be uniquely identified by its own attributes.

eg - [illegible]

represented by - ▣

- Strong Entity - uniquely identified by its own attributes.

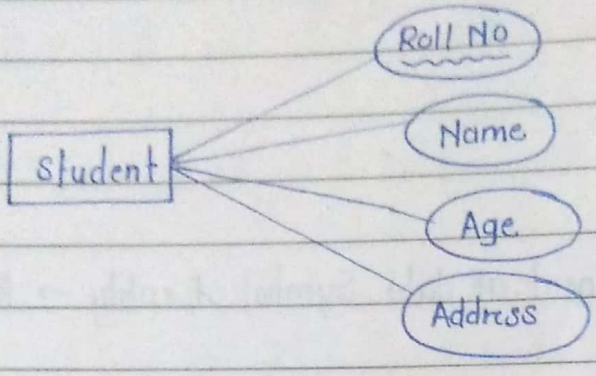
eg - [illegible]

represented by - ▭

2. Attribute - it describe the property of an entity.
   - Represented by - ⬭

Types of attribute -

- key attribute : key attribute can uniquely identify an entity from an entity set.

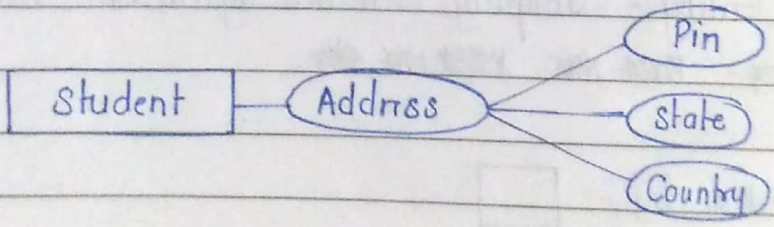


- Simple and Composite attribute :

which can't get divided into sub attribute is called SIMPLE ATTRIBUTE.

which can get divided into sub attribute is called COMPOSITE ATTRIBUTE.

e.g - Composite attribute → Address → (PIN, Road, state etc)



- Single valued and Multivalued attributes:

An attribute that can hold single value called as single ~~valueg~~. valued.

eg - student Roll.No.

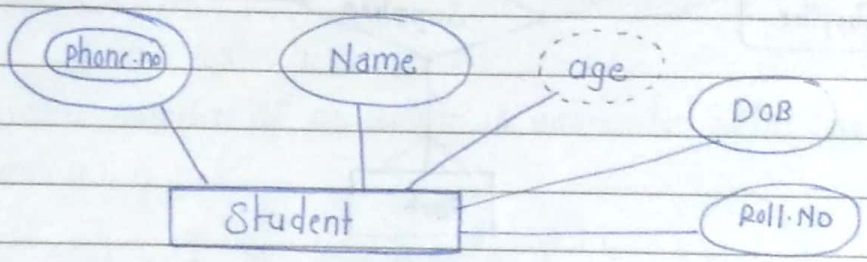
An attribute that can hold multiple values is known as multivalued attributes. represented with double oval.

eg - person can have more than one mobile number.

- Derived attribute :

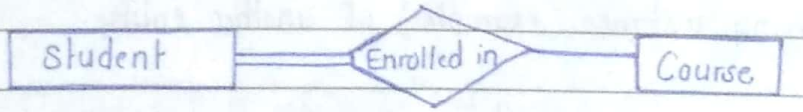
- derived attribute is one whose value is dynamic and derived from other attribute.
- Represented by dashed oval

eg - person age is change time to time and age is derived from (DOB)



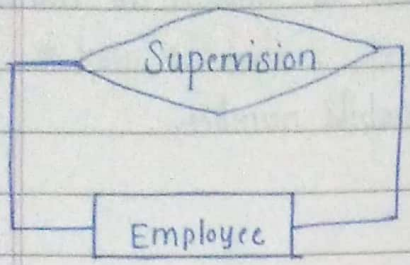
3. Relationship -

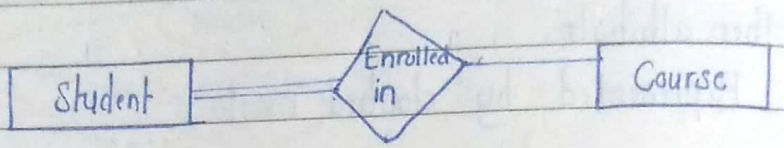
- Represented by diamond in shape. shows association among entities

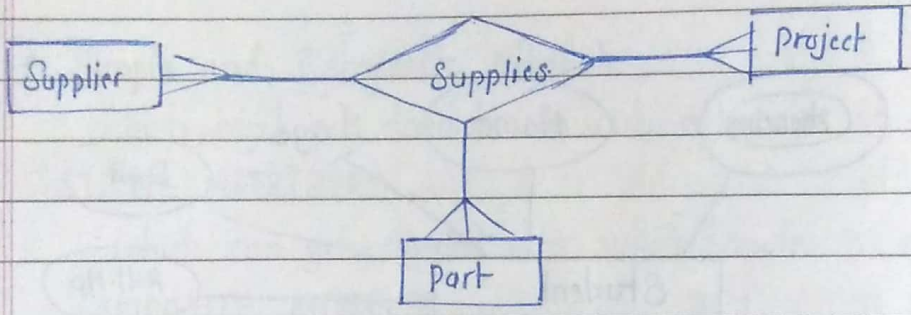


Types of Relationship -

- Unary relationship - Relationship with self
- Binary —11—— - —11—— 2 entities
- Ternary —11—— - —1<— more than 2 entities.







4. Link - Connection (line) betⁿ 2 or more entities.

5. Mapping Cardinality -

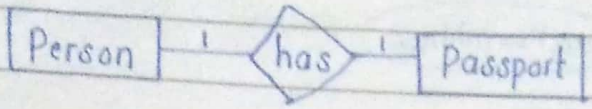
It defines how many instance of an entity is associated with a how many instance (example) of another entity.

Types :

- One to One -

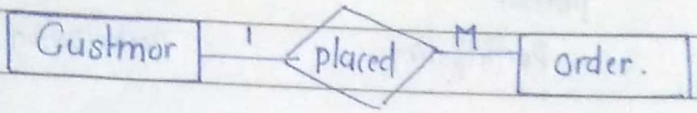
When a single instance of an entity is associated with a a single instance of another entity then it is called one to one mapping.

eg -



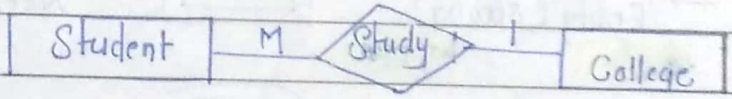
- One to many -

When a single instance of an entity is associated with more than one one instance of another entity.



- Many to one -

More than one instance of an entity is associated with one instance of another entity.



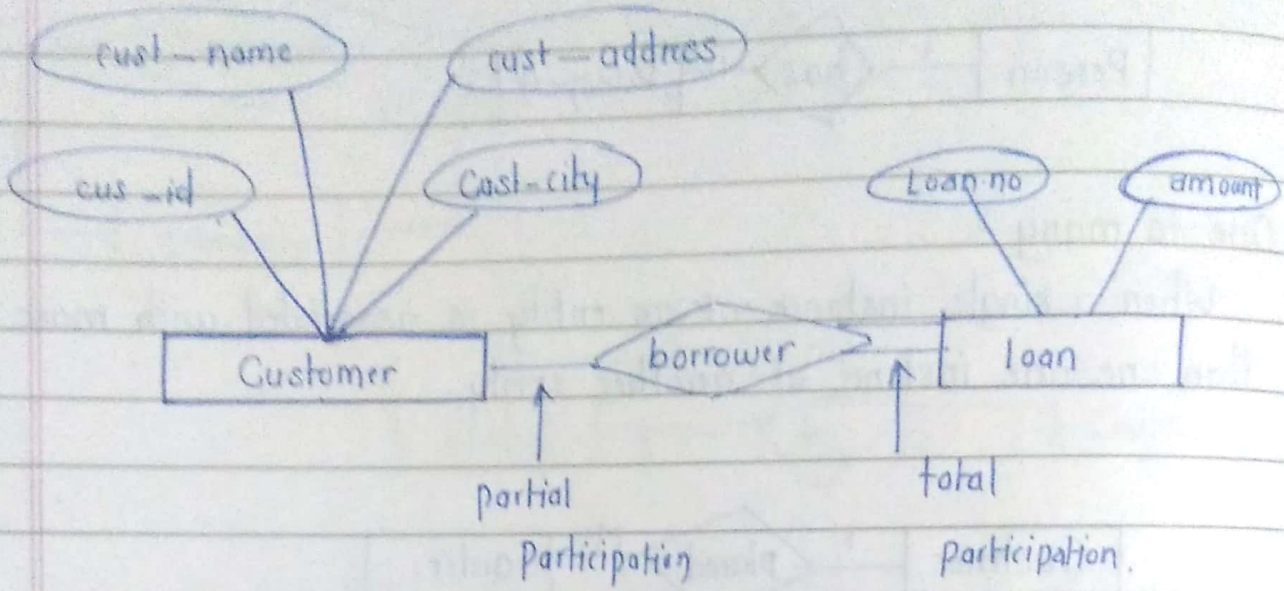
- Many to Many -

when more than one instance of an entity is associated with more than one instance of another entity.

* Total and partial participation -

Every entity in entity set participate in at least one relationship in the relationship set then it is called total participation. ═══

When some entities ~~many~~ may not participate in relationship that time it is called partial participation.



9/02/2021
Tuesday

ER Diagram

φ plotting ER diagram using case studies -

→ Enlity (strong)

→ Week

→ Attributes

→ Multivalued

→ Relation

→ week relation

— partial participation

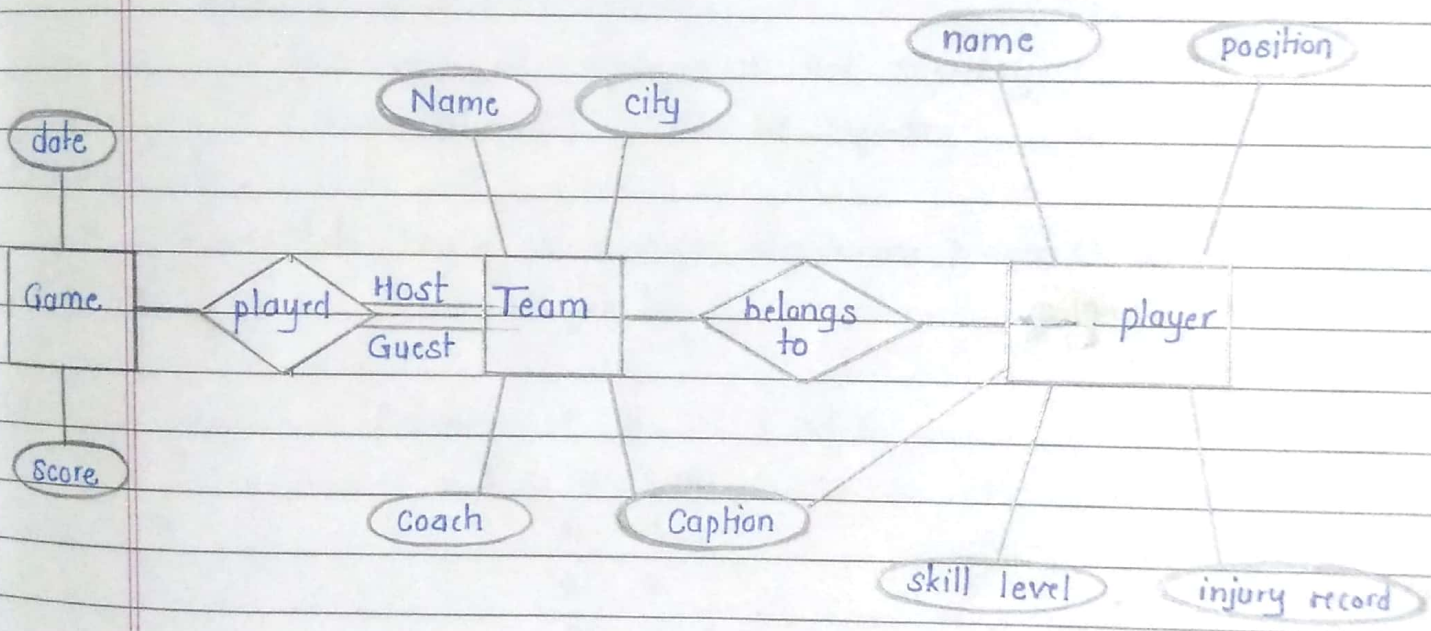
═ Total participation

National Hockey League -

Entity - Team
Attributes - Name, city, coach, caption, set of players

Entity - player
Attributes - name, position, skill level, set of injury records.

Entity - Game
Attributes - date, score



11/02/2021
Thursday

keys :-
- identifies unique rows by combination of one or more columns / attribute

* Types of keys -

- primary key - it is attribute or group of attribute that uniquely identify records.
  eg - Employee-Id

- Super key - set of one or more attribute that taken collectively to identify uniquely record / Tuple. (attribute can be one, two ....)
  eg - E-Id can be superkey
  but name of employee is not superkey
  (name, address) → can be superkey

- Candidate key - is a super key whose proper of subset is not a Super key

eg :- Relation (A , B , C) → attributes

1	1	1
2	1	2
3	2	1
4	2	2

Super key = A , AB , AC , ABC , BC

① ABC → SK
proper subset = AB , AC , BC , A , B , C
Super key ∴ ABC is not candidate key

② AC → sk
p.s = A , C → not candidate
sk

③ BC → sk
p.s → B , C
✓ Candidate key

④ A → A → no p.s is cardidate key ∴ A → Candidate key

minimal Superkey.

eg ABC BC

AB B → not S.k

✓ A ∴ BC is candiate key

* Every candidate key is super key but vice versa is not ~~of~~ true.

Altemate key = primary key is single candidate key and other than that keys are called Altemate key.

eg:- if A is primary key the remaining ABC, AB, AC, BC are candidate key.

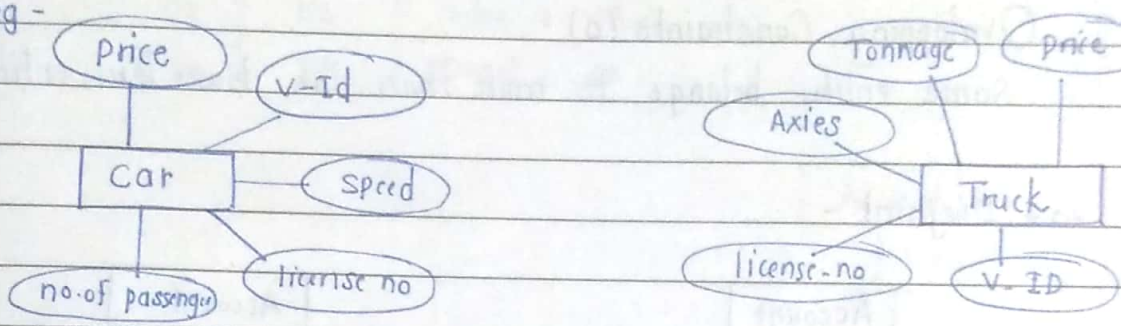
- Composite key = A key that consist of more than one attribute to uniquely identify record.
  eg = Address.

- Forrign key = Forrign keys are the columns of a table that points to the primary key of another table / cross reference.

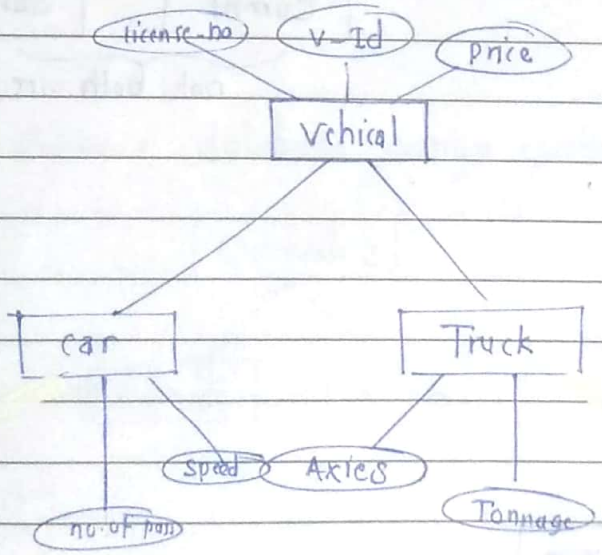
* Extended Entity Relationship (EER) Model

Generalization = extracting common properties from a set of entities and create a generalize entity from it.
It is bottom-up approch. in which two or more entities can generalised to a higher entity if they have same common attributes.

eg -



= Generalize car and Truck into vehical



Specilization = entity divided into sub-entities based on their characteristic.

- Top-down approch where higher level entity is specilized into two-or-more lower level entity.
- Into super classes and subclasses.

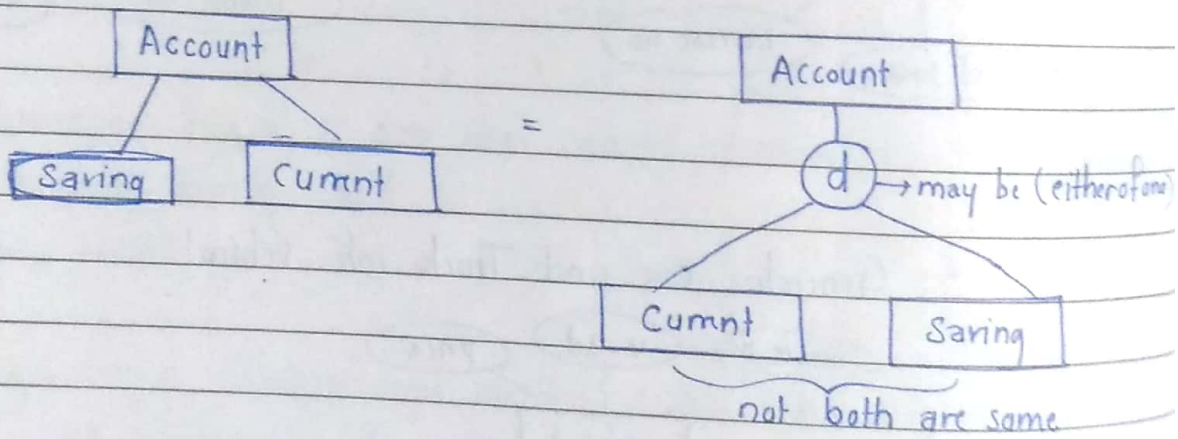
Subclass inherits attributes and relationship of super classes.

* Constraints

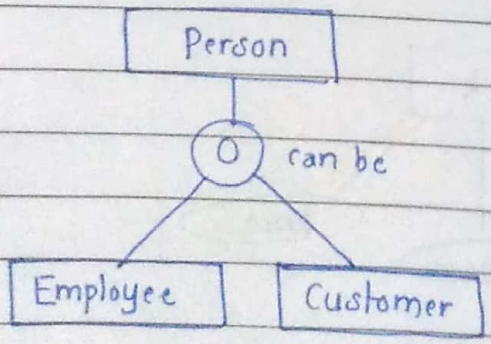
① Disjoint Constraints (d) =
An entity cannot belongs to more than one lower level entity set

② Overlapping Constraints (o) =
Same entity belongs to more than one lower-level entity set

eg = Disjoint -



eg = Overlaping -



- Customer can be employee of some company.
- Employee can be customer

Generalization type

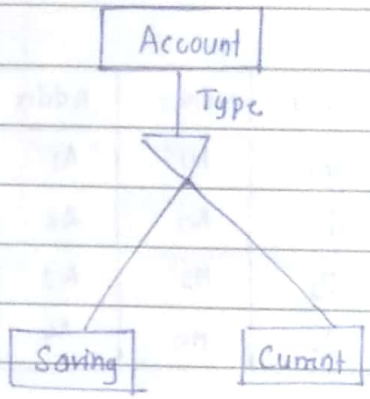
* Membership Constraints =

Attribute define ≠ Condition define =

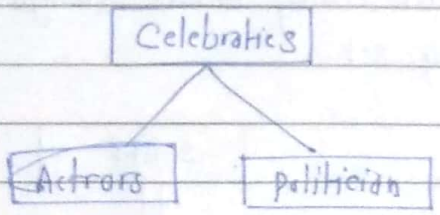
All the lower level entities are evaluated on the basis of same attribute =

eg -

Acc-no	Name	Branch	Type
a1	N1	Surat	SA
a2	N2	Cochin	CA
a3	N3	Cochin	SA
a4	N4	Bengal	CA



User define = database assign entities to given entity set.
(insertion operation level)



overlapping

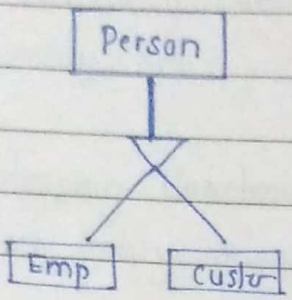
Predicate defined subclasses =

predicate on the attribute of super class / condition - - ie working hours, manage etc.

Completeness Constraints - (at least one)

1) partial - Some higher level entities may not belongs to any lower level entity.
(Through single line)

eg =



(Some persons are visitors so they can't be employee or customer)

P-id	Name	Addre	phon-
P1	N1	A1	P1
P2	N2	A2	P2
P3	N3	A3	P3
P4	N4	A4	P4

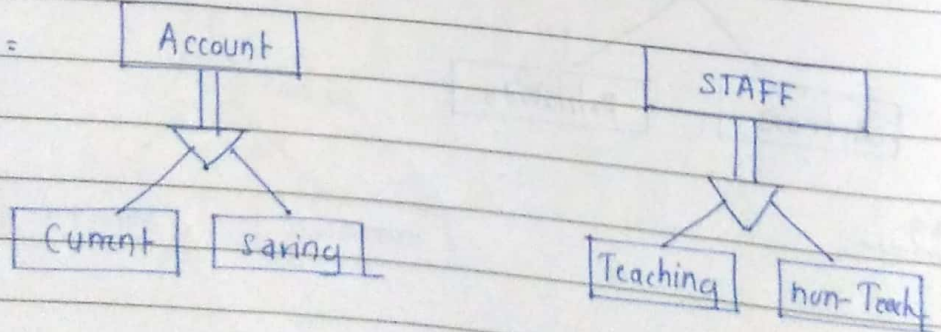
e-Id	DoB	Dept
P1	P1	X1
P2	P2	X2

C-ID	Purs..	Pate
P3	P1	P1

∴ P4 is visitor

2) Total - Each higher level entity must belongs to lower level level entity set.

eg =



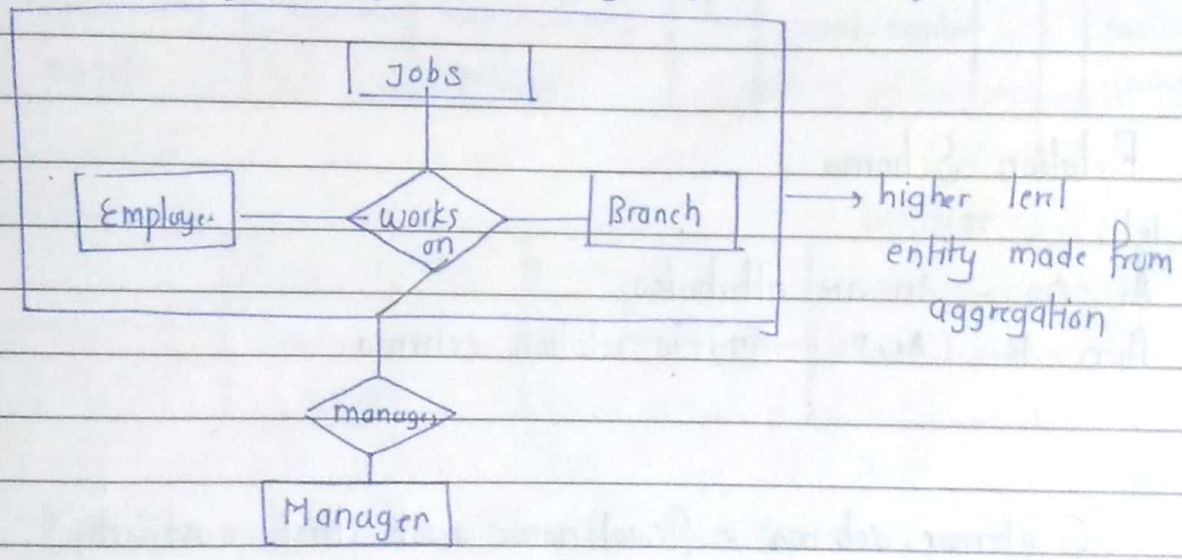
In ER Diagram → we cannot connect two relationship → in EER It is connect using aggregation

* Aggregation = / abstraction =

- In aggregation relationship between two entities is treated as single entity.

- relationship with its corresponding entities is aggrigated into higher level entity.

eg : Manager mages {Employee, job, branch}



17/02/2021
Wednesday

03. Relational Model & Relational Algebra

* Domain = The set of allowed values for each attributes is called domain of that attribute.

- Attribute values are required to be Atomic. i.e indivisible but eg - attribute can be an account number but cannot be set of account number.

- Domain is said to be atomic if all its members are atomic.
- Null value is member of every domain.

* Relation Schema

let,
$A_1, A_2, \ldots A_n$ are attributes
then $R = (A_1, A_2 \ldots A_n)$ is relation schema.

eg
Customer-schema = (cust-name, cust-street, cust-city)

r(R) = relation r on relation schema R.
eg =
customer (Customer-schema)

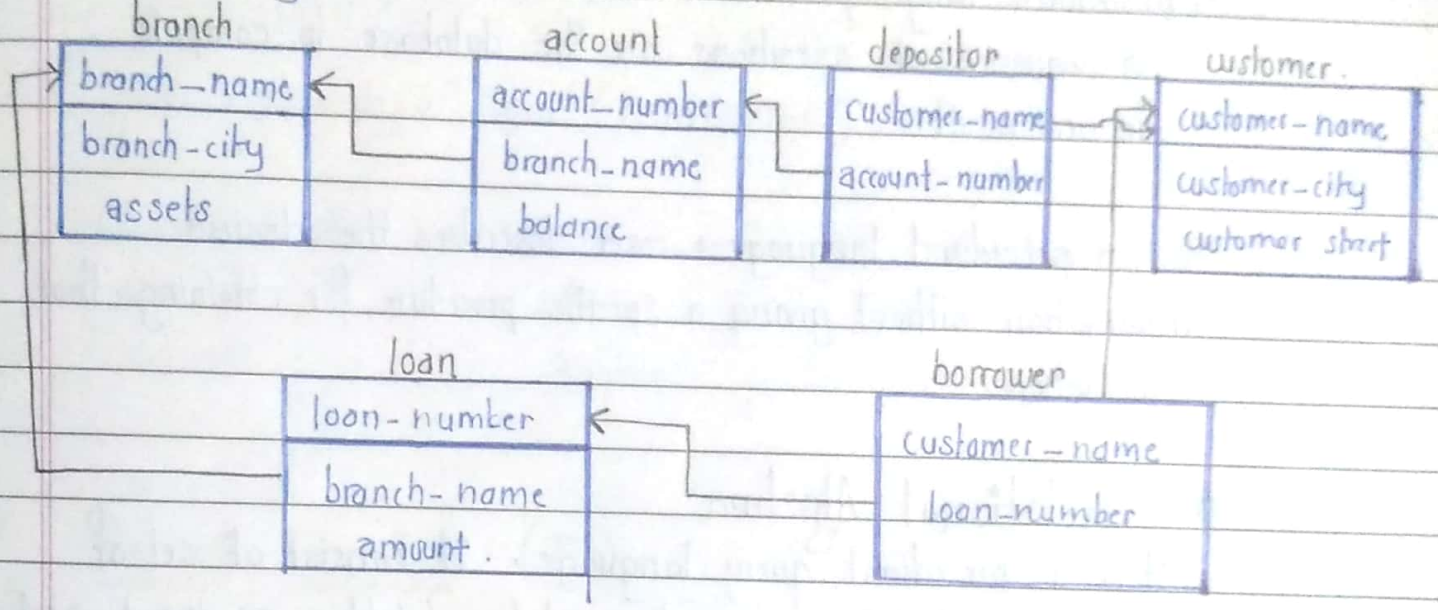
* Relation Instance = means current values of tuple/record are specified by a table.

* Relations or order of tuples are Unordered / irrelevant (not related).

* Database = Consist of multiple relations.

* Referenced relation and Referencing relation

Schema diagram =



* branch name is foreign key for loan & account and primary key for branch table

* branch is referenced relation and loan & account is referencing relation.

2021
rsday

DATE

* Relational Query Languages

- Query language is a language in which user requests information from the database.

- Query language categorized into -

1] procedural language = user instructs the system to perform a sequence of operations on the database to compute desired result.

2] non-procedural language = user describes the desired information without giving a specific procedure for obtaining that information.

* Relational Algebra

- It is a procedural query language. It consist of set of operations that take one or two relations/tables as input and produced new relation as their output.

* Relational operator-

Unary = select, project, rname.
binary = union, set difference, Cartesian product

1. Select Operation - (σ)

- It selects tuples/records that satify a given predicate.
- Selection predicates are =
  $=, <, \leq, >, \geq$

eg= Table - CUSTOMER

Customer-Id	Customer-name	Customer-city
C10100	steve	Agra
C10111	Raghu	Agra
C10115	chaitanya	Noida
C10117	Ajeet	Delhi
C10118	Carl	Delhi

Query : σ Customer-city = "Agra" (CUSTOMER)

Output

Customer-Id	Customer-name	Customer-city
C10100	steve	Agra
C10111	Raghu	Agra

2. Project Operation - (Π)

- Selects columns from the relations.

eg - from same table - CUSTOMER.

Query - Π Customer-name, Customer-city (CUSTOMER)

Output -

Customer-name	Customer-city
steve	Agra
Raghu	Agra
chaitnaya	Noida
Ajeet	Delhi
Carl	Delhi

3. Rename operation = (ρ)

- Used for renaming attributes of a relation.

eg - Table = CUSTOMER -

We are fetching customer names and renaming the resulted relation to CUST-NAMES

~~Customer-id~~

Query: ρ (CUST-NAMES , ΠT((ustomer-Name)(CUSTOMER))

Output

CUSTS-NAMES
Steve
Raghu
chaitanya
Ajeet
Carl

4. Union operation (U) =

- it includes all tuples that are in table A and in B.
- it eliminates duplicates tuple.
  Set A UNION set B = A ∪ B

- For union to be valid following condition must be hold=
- A and B must be same no. of attribute
- Attributes domain needs to be compatible
  i.e same no. of ~~datatyp~~ attributes and corresponding attributes having same datatype.
- Duplicate tuples should be automatically removed.

Table ① - STUDENT

Student-Name	Student-id	Student-Age
Aditya	S901	19
steve	S911	18
paul	S921	19
lucy	S931	17
Carl	S941	16
Rick	S951	18

Table ② - COURSE

Course-id	Student-name	Student-id
C101	Aditya	S901
C104	Aditya	S901
C106	steve	S911
C109	paul	S921
C115	lucy	S931

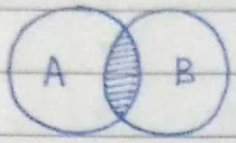
Query: Π Student-Name (COURSE) $\cup$ Π Student-Name (STUDENT)

Output

Student-Name
Aditya
steve
paul
lucy
Carl
Rick

5. Intersection Operation : (∩)

A ∩ B , Defines a relation consisting of a set of all tuple that are in both A and B. and A and B must be union compatible.



only matching rows.

Query : ΠStudent-Name (COURSE) ∩ ΠStudent-Name (STUDENT)

Output :

Student-Name
Adilya
steve
Paul
Lucy

6. Set Difference Operation = (-)

- The result of (A-B) , is a relation which includes all tuples that are in A but not in B.

- The attribute name of A must match with B.

Query : ΠStudent-Name (STUDENT) - ΠStudent-Name (COURSE)

Output :

Student-Name
Carl
Rick

Query: Π Student-Name (COURSE) − Π Student-Name(Student)

output:

Student_Name
NULL

7. Cartesian Product (X) = Cross product / Cross join.

- Used to merge columns from two relations / tables.

eg

Table ① = R

Col-A	Col-B
AA	100
BB	200
CC	300

R×S = 3×3 = 9

Table ② = S

Col-X	Col-Y
XX	99
YY	11
ZZ	101

Query: Find R×S

output

Col-A	Col-B	Col-X	Col-Y
AA	100	XX	99
AA	100	YY	11
AA	100	ZZ	101
BB	200	XX	99
BB	200	YY	11
BB	200	ZZ	101
CC	300	XX	99
CC	300	YY	11
CC	300	ZZ	101

8. Natural join (⋈)

- only performed if there is a common attribute betⁿ relations
- name and type of attribute must be same.

eg =

C

Num	Square
2	4
3	9

D

Num	Cube
2	8
3	27

C ⋈ D

Num	Square	Cube
2	4	8
3	9	27

9. Division Operation (÷) = A/B or A÷B

- Usefull for expressing queries that include a "for all" or "for every" phrase.

(A ÷ B) - applicable if and only if =
Attribute of B is proper subset of Attribute of A.

- The relation returned by division operation will have attributes = (All attributes of A - All attributes of B)

- The relation returned by division operator will return those tuples from relation A which are associated to every B's tuple

eg

R

A	B
1	1
②	①
②	②
3	4
4	1
4	4
⑤	①
⑤	②
5	3

S

B
1
2

R ÷ S

A
2
5

23/02/2021
Tuesday

Q.1 STUDENT (Ssn, Name, Subject, DOB)
COURSE (Course-id, Name, Dept)
ENROLL (Ssn, Course-id, Semester, Grade)
Book-issued (Course-id, Semister, ISBN)
TEXT (ISBN, Title, Publisher, Author)

Query ① Write a query to select all courses available in institute.

$\Pi_{Course\text{-}id, CName, Dept}(COURSE)$

② Find all student detailed registerd for course-id 10.

$\Pi_{Ssn, Name}(\sigma_{Course\text{-}id = 10}(ENROLL \bowtie STUDENT))$

③ Find various book titles and authors for semester higher than 3.

$\Pi_{ISBN, Title, Author}(\sigma_{semister > 3}(TEXT \bowtie Book\text{-}issued))$

④ Find all students belonging to IT Department

a) To Find Course-id of 'IT' Department

$T_1 \leftarrow \Pi_{\text{course-id}}(\sigma_{\text{Dept = IT}}(\text{COURSE}))$

b) To Find all students enrolled for above course-id

$T_2 \leftarrow \Pi_{\text{SSN}}(\text{ENROLL} \bowtie T_1)$

c) To Find student details having above ssn.

$\Pi_{\text{SSN, Name, DOB}}(\text{STUDENT} \bowtie T_2)$

Q2. Dealer (Dealer-no, DealerName, address)
Part (Part-No, Part-name, color)
Assignmedto (Dealer-no, part-no, cost)

1. Name of all dealers who supply red part.

$\Pi_{\text{DealerName}}(\sigma_{\text{color="red"}}(\text{Dealer} \bowtie \text{part}))$

2. —11— Supply red and yellow part.

$\Pi_{\text{DealerName}}(\sigma_{\text{color="red" OR color="Yellow"}}(\text{Dealer} \bowtie \text{part}))$

3. Name of dealers who supply all the parts.

$\Pi_{\text{DealerName}}(\text{Dealer} \bowtie \text{part})$

4. list of all dealer name.

$\Pi_{\text{Dealername}}(\text{Dealer})$

Q3. employee (person-name, street, city)
works (person-name, company-name, salary)
company (company-name, city)

Find the names of all employee who live in city "Miami"
Π person-name (σ city = "Miami (employee))

Find the names of all employee whose salary is greater than $100000
Π person-name (σ salary > 100000 (employee $\bowtie$ works))

Find the name of all employees who live in "Miami" and whose salary is greater than $100,000
Π person-name (σ city = "Miami (employee)) $\cap$ Π person-name (σ salary >100000 (employee $\bowtie$ works))
= Π person-name (σ city = "Miami" $\cap$ salary >100000 (employee $\bowtie$ works))

Q4. branch (branch-name, branch-city, assets)
customer (customer-name, customer-street, customer-id)
loan (loan-number, branch-name, amount)
borrower (customer-name, loan-number)
account (account-number, branch-name, balance)
depositor (customer-name, account-number)

Find name of all branches located in "chicago"
Π branch-name (σ branch-city = "chicago" (branch))

Find name of all borrowers who have a loan in branch "Down Town".
Π customer-name (σ branch-name = "Down Town" (borrower $\bowtie$ loan))

25/02/2021
Thursday

* Mapping of EER diagram. *

- ER-to Relational mapping algorithm

1] mapping regular entity
2] mapping of weak entity
3] mapping of binary 1:1 relation types
4] mapping of binary 1:N relationship type
5] mapping of binary M:N relationship type
6] mapping of N-any relationship + multivalued attributes.
7] mapping of N-any relationship type.
8] option for mapping specialization or generalization
9] mapping of union type.

2/03/2021
Tuesday

04. Structured Query Language (SQL)

Rename of IBM sequel language → SQL

Data Definition Language =
It allows the specification of information about relations, including :

- schema for each relation.
- domain of values associated with each attribute.
- Integrity constraints.

DDL Commands in SQL

1. CREATE -
   Used to define the database structure schema.
   Syntax :

```
CREATE TABLE TABLENAME (COLUMN_NAME DATATYPES [,... ]);
```

eg -

```
Create database university ;
Create table students (Roll_no int, Sname varchar(20));
```

2. DROP -
   Drops commands remove tables and database from RDBMS
   Syntax :

```
DROP TABLE ;
```

eg -

```
Drop database university ;
Drop table student ;
```

3. ALTER -

Alters command allows you to alter the structure of the database.

Syntax :

- To add a new column in the table

```
ALTER TABLE table-name ADD column-name COLUMN-definition;
```

- To modify an existing column in the table :

```
ALTER TABLE MODIFY( COLUMN DEFINTION...);
```

eg =

```
Alter table gurugg add subject varchar(10);
```

4. TRUNCATE -

Used to delete all rows from the table and free the space containing the table.

Syntax :

```
TRUNCATE TABLE table-name;
```

eg =

```
TRUNCATE table student;
```

5) RENAME -

Used to set new name for any existing table.

Syntax :

RENAME TABLE old-table-name to new-table-name

eg -

RENAME TABLE student to student-info;

13/2021 Wednesday

* DML (Data Manipulation Language)

- This language allows us to modify database instances by inserting, modifying and deleting its data.

DML Commands in SQL

1] INSERT - Insert data into the 'Row'

Syntax :

INSERT INTO TABLE-NAME (COL1, COL2, COL3 ... COLN) VALUES (value1, value2, value3, ... valueN);

OR

INSERT INTO TABLE-NAME (value1, value2, value3, ... valueN);

eg = INSERT INTO students (RollNo, FirstName, LastName) VALUES ('60', 'TOM', 'Erichsen');

2] UPDATE - update or modify the value of a column in the table.

Syntax :

UPDATE table-name SET [column_name1 = value1 .... column-nameN = valueN][WHERE CONDITION]

eg = UPDATE students SET FirstName = 'Jhon', LastName = 'Wick' WHERE studID = 3;

3] DELETE - Used to remove one or rows from a table.

Syntax :

DELETE FROM table-name[WHERE condition];

eg: DELETE FROM students WHERE First-Name = 'Jhon';

* DCL (Data Control Language)

- This language give "rights & permissions".

1] Grant =

use to give user access privileges to a database.

Syntax :

GRANT SELECT, UPDATE ON MY-TABLE TO SOME-USER, ANOTHER-USER;

eg: GRANT SELECT ON USER TO SAM;

2] Revoke =

It is useful to back permissions from the user.

Syntax :

REVOKE privilege-name ON table-name FROM [user-name | PUBLIC | role-name]

eg - REVOKE SELECT, UPDATE ON student FROM BCA, MCA;

* TCL (Transaction Control Language)
deals with transactions within the database.

1] Commit - used to save all the transactions to the database.

Syntax :

Commit;

eg - DELETE FROM Student WHERE RollNo = 25;
COMMIT;

2] RollBack - allows us to undo transactions that have not already been saved to the database.

Syntax:

ROLLBACK;

eg - DELETE FROM students WHERE RollNo = 25;
ROLLBACK;

3] SAVEPOINT - helps us to sets a savepoint within a transaction.

Syntax :

SAVEPOINT SAVEPOINT_NAME;

eg - SAVEPOINT RollNo;

4/03/2021
Thursday

- SELECT * FROM TABLE-NAME ;
  - It will select all columns in the table.

- SELECT column1 , column2 , ... FROM TABLE-NAME ;
  - It will select columns mention in query

- SELECT DISTINCT column1 , column2 , ... FROM TABLE-NAME ;
  - select only distinct values from given columns.

- SELECT column1, column2, ... FROM TABLE-NAME WHERE condition ;
  - extract only those records that fulfill a specified condition.

- SELECT column1 , column2 , ... FROM TABLE-NAME WHERE condition1 AND condition2 AND ... ;
  - display record if all conditions are true.

- SELECT column1 , column2 ... FROM TABLE-NAME WHERE condition1 OR condition2 OR ... ;
  - display record if any of condition is true

- SELECT column1 , column2 ... FROM TABLE-NAME WHERE NOT condition ;
  - select all values / record except the record which satisfy condition.

- SELECT column1 , column2 ... FROM TABLE-NAME ORDER BY column1 , column2 , ... ASC / DESC ;
  - Sort result set in ascending or descending order.
  - bydefault → ascending.

- INSERT INTO table-name VALUES (value1, value2 ...);
  - adding values in all table (make sure order of value is same as that of columns name)

- INSERT INTO table-name (Column1, column2, ...) VALUES (value1, value2, value3 ...);
  - Insert values in specified columns.

- SELECT column-names FROM table-name WHERE column-name IS NULL;
  SELECT columns which satisfy given condition.

- SELECT column-names FROM table-name WHERE column-name IS NOT NULL;
  —— 11 —— 11 ——

- UPDATE table-name SET column1 = value1, column2 = value2... WHERE condition;
  - modify the existing record of column which satisfies condition

- DELETE FROM table-name WHERE condition;
  - delete record of column which satisfies condition.

- SELECT TOP number|persent column-name(s) FROM Table-name WHERE condition;
  select the records of given number written in query.

- SELECT column-name(s) FROM table-name WHERE condition LIMIT number;
  = select the record of given number which ~~follows~~ satisfy condition.

- SELECT column-name(s) FROM table-name WHERE ROWNUM <= number;
  - select upto Rownum <= number.

- SELECT MIN(column-name) FROM table-name WHERE Condition;
  - return smallest value of selected column.

- SELECT MAX(column-name) FROM table-name WHERE Condition;
  - return largest value of selected column.

- SELECT COUNT(column-name) FROM table-name WHERE Condition;
  - Count no. of rows satisfies given condition.

- SELECT AVG(column-name) FROM table-name WHERE condition;
  - Returns avg value of numeric column.

- SELECT SUM(column-name) FROM table-name WHERE Condition;
  - Returns sum of numeric columns.

- SELECT column1, column2, ... FROM table-name WHERE ColumnN LIKE pattern;
  - search specified pattern in a column.

pattern ⟶ a% ⟶ name starting with a
%a ⟶ name ending with a
%or% ⟶ or in any position
_r% ⟶ r in second position.
a__% ⟶ starts with 'a' and atleast 3 character in
a%o ⟶ starts with a end with o
NOT LIKE 'a%' ⟶ not starts with a

_ → single character.
% → 0 1 or multiple

SQL Group by statement

* GROUP BY
- groups rows that have the same values
- it is also used with aggregate functions (COUNT, MAX, MIN, SUM, AVG) to group the result.

• SELECT column_name(s) FROM table_name WHERE condition GROUP BY column_name(s) ORDER BY column_name(s);

* Having Clause
- used instead of WHERE keyword bcz WHERE ~~col' col~~ could not be used with aggregate functions.

• SELECT column_name(s) FROM table_name GROUP BY column_name(s) HAVING condition;

* ANY -
It will return true value if any of the subquery value meet the condition.

• SELECT column_name(s) FROM table_name WHERE column_name operator ANY (SELECT column_name FROM table_name WHERE condition);

* ALL -
Returns true if all the subquery values meet the condition.

• SELECT column_name(s) FROM table_name WHERE column_name operator ALL (SELECT column_name FROM table_name WHERE condition);

* UNION Operator

- Used to combine the result-set of two or more SELECT statements.
- each SELECT statement within UNION have same no. of columns and columns must have same datatype and order.

- SELECT column-name(s) FROM table1
  UNION
  SELECT column-name(s) FROM table2 ;

} by default it only select distinct values.

UNION ALL → To select all values (including duplicate)

- SELECT column-name(s) FROM table1
  UNION ALL
  SELECT column-name(s) FROM table2;

* IN Operator -

- allows us to specify multiple values in a WHERE clause.

- SELECT column-name(s) FROM table-name
  WHERE column-name IN (value1, value2, ...);

OR

- SELECT column-name(s) FROM table-name
  WHERE column-name IN (SELECT STATEMENT);

* BETWEEN Operator -

Selects values within a given range. (Including begin & end value)

- SELECT column-name(s) FROM table-name WHERE column-name BETWEEN value1 AND value2;