Unit - 1

Saathi

Date __/__/__

- **Data** ⇒ Data is meaningful known fact that can be processed and stored as information.

- **Database** ⇒ Database is a collection of inter-related and organised data.

- **Database Management System** – It is a collection of inter-related data and a set of program to access, update and manage those data.

eg - · MS access
· dBase
· File Maker Pro
· FOX BASE
· ORACLE
· Postgre SQL
· Ingress
· Informix
· My SQL
· MS SQL Server
· IBM DB2

Primary goals of DBMS

- To provide a way to store and retrieve database information that is both convenient and efficient.

- To manage large and small bodies of information. It involves defining structure for storage of information and providing ~~mecha~~ mechanism for manupulation of information.

- It should ensure safety of information stored despite system crashes or attempts on unauthorised access.

- If data are to be shared among several users then the system should avoid possible anomalous result.

Disadvantages of File Processing System

1. Data Redundancy :- Same information may be duplicated in several file and this duplication data over several file is known as Data Redundancy.

ex:- Address and telephone no. of a particular customer may appear in a file that consist of saving account record and in a file consist of current account records

2. Data Inconsistency :- The various copies of same data may no longer agree that is various copies of the same data may contain different data.

3. Difficulty in accessing the data :-

In conventional file accessing system it is difficult to acces data in a specific manner. And it is required to create a application program to carry each new task.

4. Data Isolation :- Because data is scattered in various files and files may be in diff. format, writing new application program to retrive appropriate data is difficult.

5. Integrity constraints :- The data value stored in a database must satisfy certain types of consistency contraints.
eg:- balance of bank account may never fall below the prescribed amount.

6. Atomicity problem :- In many application it is critial that if a failure occur the data may be restored to the consistent state that exist prior to the failure. It is difficult to ensure atomicity in a conventional file processing system

A	B
1000	2000
1000 - 500 = 500	2000 + 500 = 2500

during many transfer process may some failure occur.

7. Concurrent Access Anomalies :-

For the sake of overall performance of the system and faster response, many system allocation multiple user to update the data simultaneously, in such an environment interaction of concurrent update is possible and may result in inconsistent data.

A
1000

clerk 1		clerk 2	
500		100	
↓		↓	
500	or	900	correct 400

8. Security Problem :-

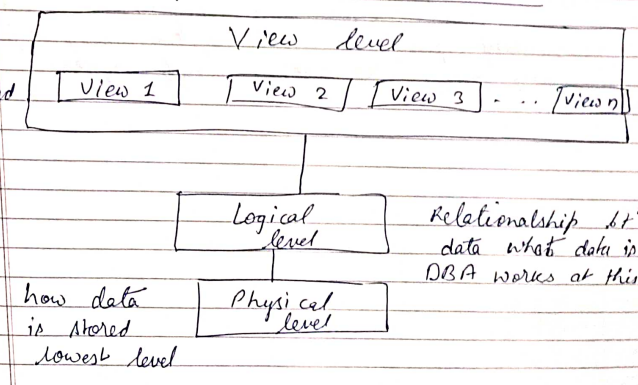
Not every user of database system should be able to access all the data.

Data Abstraction

When the DBMS hides the certain details of how data is stored and maintained it provides what is called as the abstract view of data. Abstract view. It is used for following :-

i) To provide Abstract view of data.
ii) To hide complexity from user.
iii) To simplify user interaction with DBMS.

Levels of Data Abstraction



Physical level :-

It describes how a record is stored. It is a lowest level of Abstraction efficient algorithm are defined to access data.

Logical level :-

It describe what data are stored in the database and the relationship among the data database administrator (DBA) work at this level.

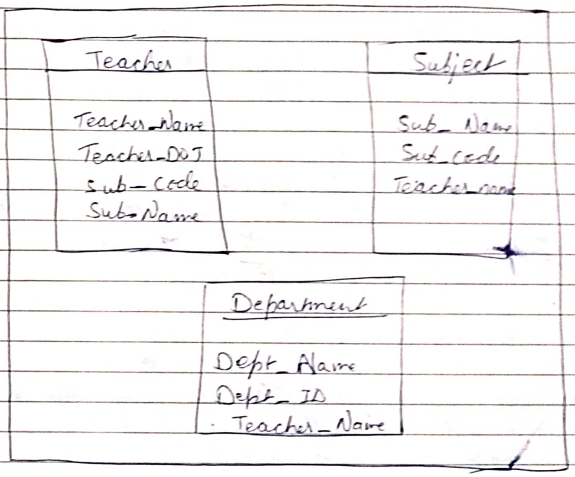
View level

Application program hide details of datatype. Views can also hide information for security purposes. The system may provide many views for the whole system.

Instance and Schema

Data base :- School

Tables :- Teacher, Subject, Department



Saathi

Date

Schema :- Description of table is schema
We cannot modify it.

Instances :- It is collection of information stored at particular moments.
The instances can be changed by certain operation like deletion, addition.
Let say a table teacher in our database whose name is " ". Suppose if the table has 50 records so the instance of database has 50 records for now & tomorrow we are going to add another 50 records. So tomorrow instances has 100 records. This is called as instances.

Schema :- It is the overall description of the database.
The basic structure of how the data will be stored in the database is called Schema.
There are of 3 types:-

i) Physical schema
ii) Logical schema
iii) View schema

i) Physical Schema - It describes the database design at physical schema.
It is hidden below the logical schema and can be changed easily without affecting the application program.

Page No.

Saathi

Date

ii) Logical Schema :- It describes the database design at the logical level.
Programmers construct application using logical schema.

iii) View Schema :- It defines the design of the database at the view level.
It is the highest level of schema which defines the views for the end user.

Example :-

Teacher table

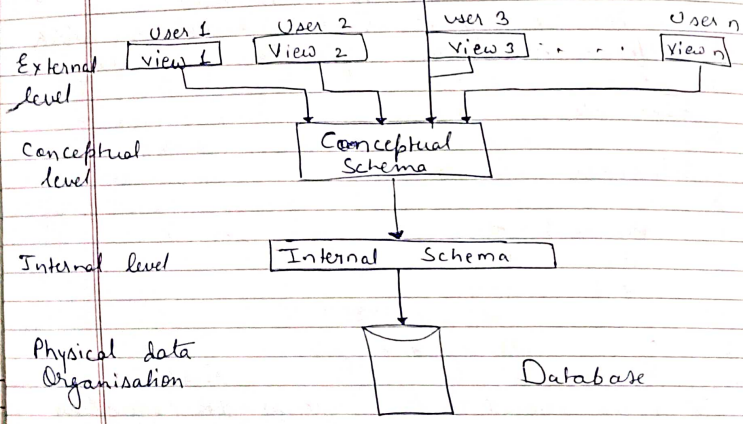
- name : string
- doj : data
- dob : date

Schema	Instance
• It is the overall description of the database	• It is the collection of information stored in a database at a particular moment.
• It is same for all database.	• Data in instances can be changed using addition, deletion, updation.
• Doesn't change frequently	• changes frequently.
• Defines the basic structure of the database i.e.	• It is the set of information stored

Page No.

how the data will be stored in the database. | at a particular time.

ANSI / SPARK 3 level Architecture of DBMS



In 1971 DBTG (Database Task group) realised the requirement for a 2-level approach i.e. views & schema and in afterward 1971 Henry Sparc realised the need for 3 level approach with the 3 levels of abstraction.

comprises of n external, conceptual and internal level.

The aim is to seperate each user view of the database from the way the database is physically represented.

- External level :- It is the view how the user views the database. The data of the database that is relevent to that user is described at this level. In the external view only that entities, attributes, relationships are included that the user wants. The different views may have different ways of representing the same data. Example - One user may view name in the form (first name, last name) while another may view as (last name, first name)

- Conceptual level :- It is the community view of the database and describes what data is stored in the database and represents the entities, their attributes and their relationships. It represents the semantic, security and integrity information about the data. This level contains the logical structure of the entire database.

- Internal level :- At the internal level the database is represented on the computer. It emphasises on the physical implimentation of the database to do storage space utilization and to achieve the optimal runtime performance and data encryption technique.

Saathi

Date / /

Data Independence :- It is the ability to modify a schema definition in one level without affecting a schema definition in higher level is called data independence.

There are two types of data independence :-

- Physical data Independence
- Logical data Independence

Physical data Independence - It is the ability to modify the physical schema without causing application programs to be rewritten. Modification at this level is usually to improve performance.

Logical data Independence - It is the ability to modify the conceptual schema without causing application program to be rewritten. It is usually done when logical structure of database is altered.

Different Types Of DBMS

1) **Naive user** - Parametric User

They are also known as Parametric end user. They donot have any knowledge of DBMS but still frequently use database application to get the desired result with the help of interface

Saathi

Date / /

provided by the DBMS application. Naive user mostly use the database to fill in or retrieve the information. They donot need to be aware of the presence of the database system as they can interact with database with the help of menu driven application interface.

Eg - Railway ticket booking

2) **Application Programmers** :- They are also known as backend developers, are computer professional users who are responsible for developing the application program or the user interface so that the other user can use these application to interact with database. They have deep knowledge of DBMS and know everything in detail. When needed application programmers do specified modification needed in the database structure for an application. They are efficient enough in designing or developing their database in any language they know.

3) **Sophisticated users** :- They are that type of database user who know DBMS and are familiar with the database. They can be business analysts, scientist, engineers etc. They can develop or access they their database application according to the

requirement with actually writing the code

iv) Specialized user :- They are also called sophisticated user but they are responsible for writing the special database application program. The specialized users are also responsible for the develop of the program according to the given requirement

Database Administrator And its Functions

Database adminstrator is a person having central control over data and program accessing that data. He coordinates all the activities of the database system. He has a good understanding in enterprise's information and needs.

Function Of DBA

i) Schema Definition :- The creation of original database schema.
This involves writing a set of informations in DDL compiled by DDL compiler into a set of table stored in a database dictionary.

ii) Storage structure and access method definition :- Writing a set of definition translated by a data storage and definition language compiler

iii) Schema and Physical Organization modification :

Writing a set of definitions used by the DDL compiler to generate modification to appropriate internal systems table. This is done rarely but sometime database schema or physical organisation must be modified.

iv) Granting user authority to access database :

Granting different types of authorization for data access to various user.

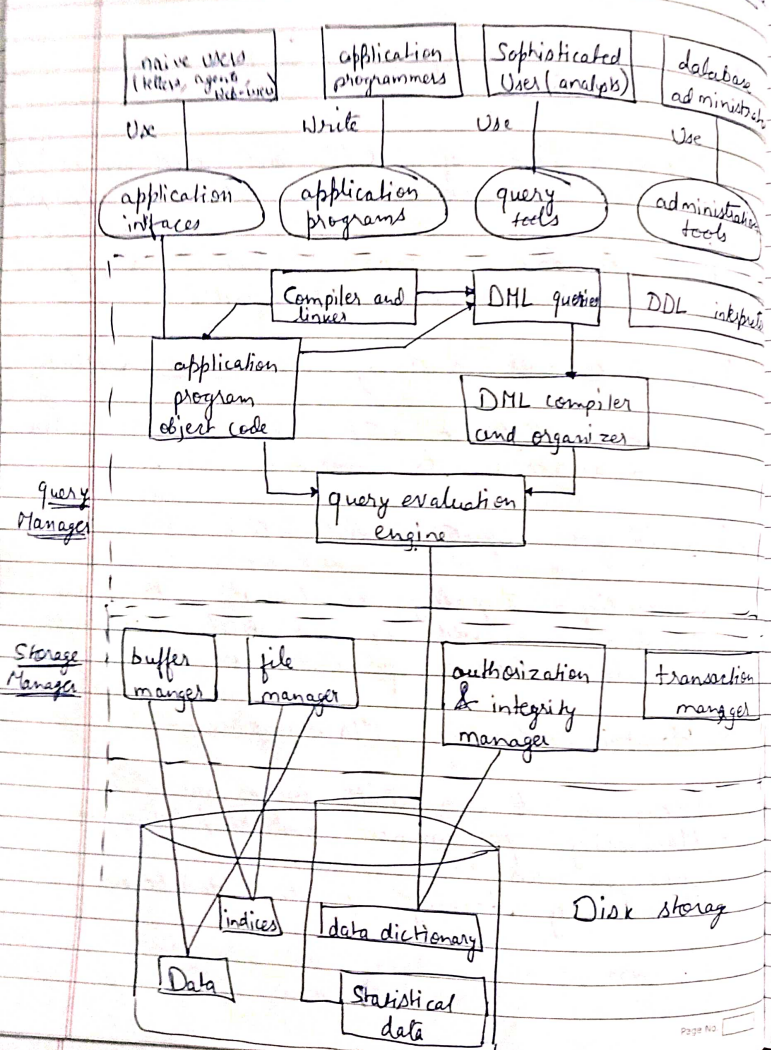
v) Specifing integrity constraints :

Generating integrating constraints. These are consulted by the database manages module whenever update occur

vi) Maintain. Routine Maintainance :

- Acting as liaison with users to
- Monitoring performance and responding the changes in requirement.
- Periodically backing up the database.

DBMS system structure and its components



Truncate is used to delete all rows at a time.

DDL - Data definition language.
Create, Drop, Alter, Truncate

DML - Data manipulation language.
Select, insert, update, delete.

Database system are partioned into modules for different functions. Functional components of database system are query processors. It interprets the request (queries) recieved from end user via an application program into instruction. It also executes the user request which is recieved from the DML compiler.

Query processor contains the following components

i) DML compiler - It processes the DML statements into low level instruction so that they can be executed.

ii) Query evaluation engine - It executes low level instruction generated by DML compiler.

iii) DDL compiler - It processes the DDL statements into a set of table containing meta data (data about data)

Storage Manager - It is a program that provides an interface between the data stored in the database and the queries recieved. It is also known as database control system. It maintains the consistency and integrity of the database by applying the constrains and executing the DCL statements. It is also

responsible for updating, storing, deleting, retrieving data in the database.
It contains following documents :-

i) Authorization manager - It ensures rules based access control i.e. checks whether the particular person is previldges to perform the requested operation or not.

ii) Integrity Manager - It checks the integrity contrains when the data is modified

iii) Transition manager - It control concurrent access by performing the operations in a scheduled way that it received the transition thus it ensures that the database re consistent state before and after the ~~transe~~ execution of the transaction.

iv) File manager - It manages the file space and the datastructure used to represent information in the database.

v) Buffer Manager - It is responsible for cache memory and the transfer of data between secondary storage and main memory.

Disk Storage :-

i) Data files - Stores data.
ii) Data dictionary - Structure of database object
iii) Indices - Provides faster retrieved of data items

Data Models in DBMS

1) Relational data model
2) Entity Relationship (ER) model
3) Object Based Data model
4) Semi-structured
5) Older models
   - Hierarchical
   - Network Model

- Data Models in DBMS is the concept of tools that are developed to summarize the description of the database.
- It defines how the logical structure of a database is modelled.
- A datamodel is collection of conceptual tools or describing, data relationship, data symantics, consistency constraints.
- It describe the design of data base at each level of data abstraction.
- It defines how data is connected to each other, how they are processed and store inside the system.

Saathi

Date __/__/____

1. ~~Relational Student~~

1. Relational data model :-

Table : Student (Entity) — Attributes

Student Id	Student Name	Student Age
1	Ram	23
2	Shyam	22
3	Mohen	20
4	Sohan	18

Records / Tuples

- Most widely used model by commercial data processing application
- Uses collection of tables for representing data and the relationship among those data.
- Data stored in table called relations.
- Each table is group of columns and row where column represent attributes of an entity and rows represents records (tuples)
- These model was initally described by Edgar F Codd in 1969.

Attribute or field - Each column in a relation is called an attribute. The values of attributes should be from the same domain.

Tuples or Records - Each row in the relation is called tuple. Tuple defines a collection of attributes values. Each row in a relation contains unique values.

Page No.

Saathi

Date __/__/____

Model

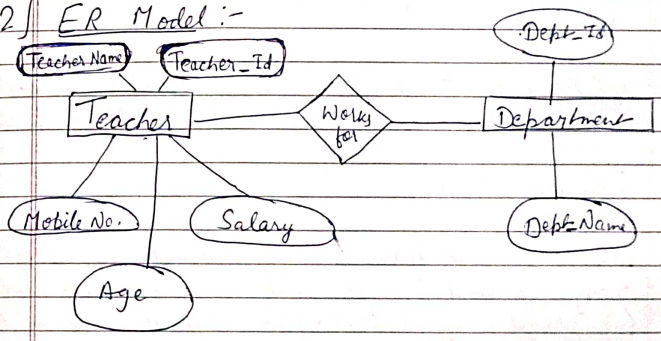
Advantage and Disadvantages of Relational

- Structural Independence
- Simplicity
- Ease of designing, implementation, maintainance and usage.
- Adhoc querry capability.

Disadvantages

- Hardware overheads.
- Ease of design can result in bad design.

2. ER Model :-



ER - Model

- ER-Model high level data model diagram.
- Describes structure of the database with the help of diagram which is known as ER-diagram.
- When ER-diagram is design or a database that can be implemented as a database.
- It is based on the notion of real world entity and relationships among them.

Page No.

Saathi

Date __/__/____

- ER diagram has the following 3 components:

i) Entities :- It is a real world thing or object. It can be a person, place or even a concept. Ex- Teacher, student, course, building, department etc are some of the entities of school management system.

ii) Attributes - An entity contain real world property called attributes. The entity teacher has properties like name, salary, age etc.

iii) Relationships :- It tells how 2 entities are related. Teacher works for a department.

Advantages:-

- Straight forward relational representation.
- Easy conversion for ER to other data model.
- Graphical representation for better understanding.

Disadvantages:-

- No industry standard for notation.
- Popular for high level design.

Page No.

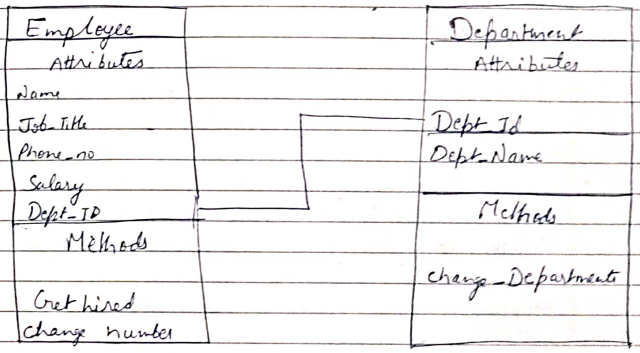
Saathi

Date __/__/____

Object - Based Data model

1) Object - oriented data model
2) Object - relational data model

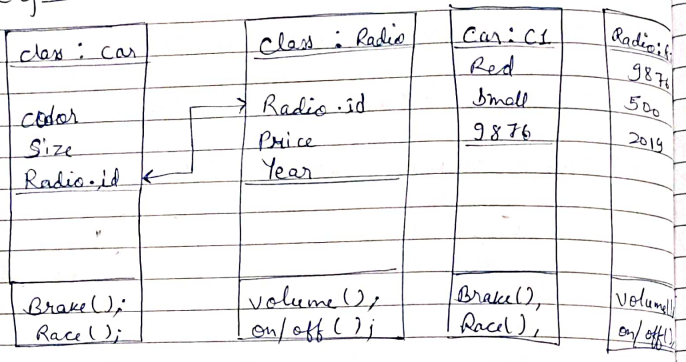
1) Object oriented data model :-



- An extension of ER model with notion of functions, encapsulations and object identity as well.
- Both the data and relationships are present in a single structure known as object.
- Two or more objects are connected through links, we use these links to relate one object to other object where employee and department are object.
- Two object are connected through a common attribute i.e. 'dept_Id' and the communication between these 2 will be done with the help of these common Id.

Page No.

b] Object-relational data model



- Combination of object oriented data model and relational data model.
- Build to fill gap between object oriented model and relational model.
- It has many advances features like complex data type that can be formed using existing data Item.
- Complex and difficult to handle.
- Here object are created like Car C_1 and Radio R_1, by using above class & object we can find the year of Radio having Id 9876

Advantages:- [Object Oriented data model]

- Capable of handling large variety of data types.
- Combining object oriented programming with data base technology
- Improved productivity
- Improved Data axis

Disad.

- No precise definition
- Difficult to maintain
- Not suited for all application

Semi structured data Model

Ex:

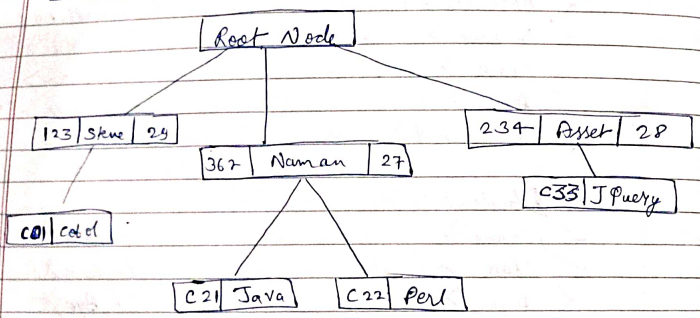
```
<Student 1>
<Roll no> . . . . . </Roll no>
<Name> . . . . . </Name>
<class> . . . . . </class>
<Age> . . . . . </Age>
</Student 1>

<Student 2>
<Name> . . . . . </Name>
<class> . . . . . </class>
<Age> . . . . . </Age>
</Student 2>
```

- An evolved form of Relational model.
- This model allows the data specification at place where individual data items of the same type may have diff. sets of attribute.
- In this model some entities may have missing attributes while others may have extra attribute.
- ~~This model helps in giving~~ flexibility to the attributes for ex
- This also give flexibility to the attribute. For ex if we are storing any value in

attribute then that value may be atomic values or collection of data.

- XML is used for representing semi-structure data
- Here in Student 2 Roll no. is missing but in relational model, it is not possible that Student 1 and Student 2 have different attributes.

Hierarchical Data Model



DBMS model

- Data is organized into a tree like structure with each record is having one parent record and many children.

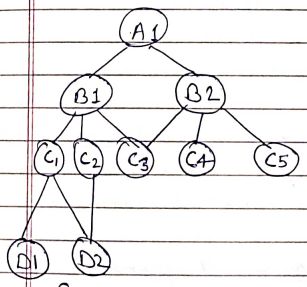
Drawbacks:-

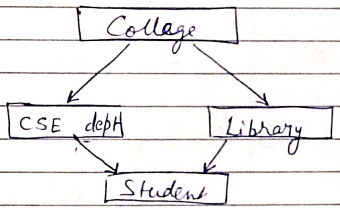
- It can have only 1 to many relationship between nodes.
- This model used rarely now.
- In this diagram we have few student & few course A course can assign into single student only.

however student take any no. of courses

Advantages:-	Disadvantages:-
1. Simplicity	1. Implementation Complexity
2. Data Integrity	2. Database management problem.
3. Data Security	3. Lack of structural independence.
4. Efficiency	4. Programming Complexity
	5. Implementation limitation

Network Model





Extension of Hierarchical Model

- Most popular model before relational model.
- Same as hierarchical model except that it has graph like structure rather than tree based structure and are allowed to have more than one parent node.

- Supports many to many relationships.
- Here no students have 2 parent node i.e. cse department & Library.

Advantages:-

- Simplicity
- Ability to handle more relationship types.
- Ease of data access
- Data Integrity
- Data Independence
- Database standard.

Disadvantages:-

- System complexity
- Lack of structural independence.

ER Model in detail

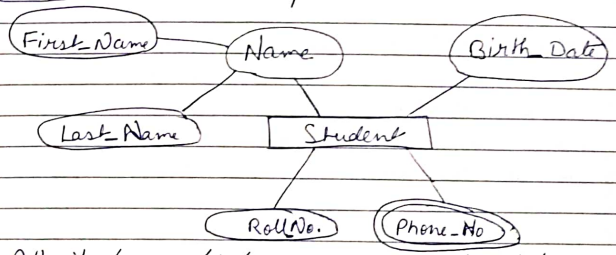
1) Entity set:- It is a set of entities of the same type that share same properties or attributes. Ex- Student set, Teacher set.
   - Entity set need not be disjoint.
   - Entities set should be Capital letter.

2) Attribute :- It should be noun and 1st letter of each word should be capital. Should follow standard format.
   Ex:
   Student_GPA not GPA_of_student

Types:-

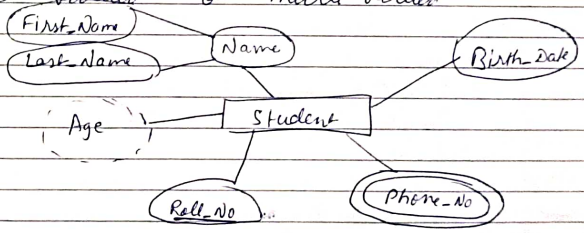
- Simple and composite.
- Singled Valued or multi-valued
- stored or derived
- Null or prime
- Key attributes

1 => Simple and composite :-



Attributes which are not divided into sub parts are called simple attributes.
Ex:- Costumer_Id.
Composite attributes are made up of more than 1 simple attribute. Ex:- Student complete name. First_Name and Last_Name.

- Singled valued or multi valued



2 → Single valued attribute contain single value. Ex:- Roll_No, Aadhar_No
Multivalued may contain more than one value and is represented by double ellipse. Ex - Phone_No

3 → Stored or derived :-

Stored attributes are physically stored in data base. Ex - Name, Roll_No.
Derived attributes are attribute that doesn't exist in the physical values but their values are derived from other attribute present in database. It is represented by dashed ellipse. Ex:- Age can be derived from dateofBirth.

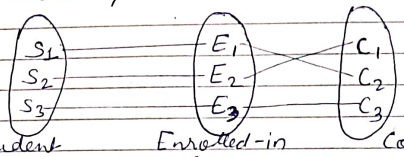
4 → NULL or Prime :- An attribute takes NULL values when an entity doesn't have value for it. It may indicate not applicable. Ex:- One may have no middle name.

5 → Prime attribute :- An attribute whose value are used to identify an entity of within an entity set is known as prime attribute. It is similar to key attribute. Ex - Registration_No, Roll_No

5 → Key attributes → Attributes which uniquely identifies each entity in the entity set is called key attribute. It represents primary key. It is represented by ellipse with underline.

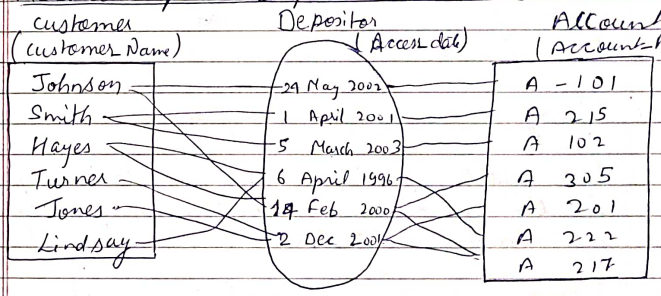
3) Relationship Set :-

Set of relationship of similar type is called relationship set.
Relationship set Enrolls (E_1, E_2, E_3)



Student Entity set | Enrolled-in Relationship set | Course Entity set

4) Relationship descriptive attributes



- Like entities, a relationship to have attributes, this attributes are called descriptive attributes.
- Eg - the depositure relationship set betn entity set customers & account, may have the attribute access date.

Degree Of Relationship

- Unary
- Binary
- Ternary
- N-ary

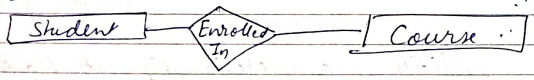
The no. of different entity sets participating in a relationship is called as degree of a relationship set.

i) Unary Relationship :- degree = 0
Here, only one entity participate in a relationship. eg :-

One person is married to only one person.

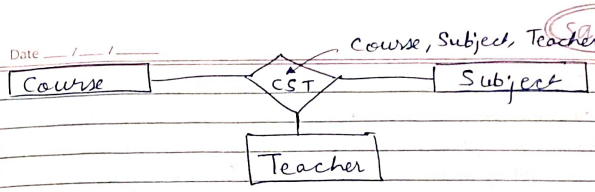
ii) Binary Relationship (degree = 2)

Two entity participate in a relationship and is the most common relationship degree.
eg :- Student is enrolled in course.



iii) Ternary Relationship (degree = 3)

Three entities participate in the relationship
eg :- The university might need to record which teacher taught which subject in which courses.



iv) N-ary (degree = N)
N-Entity set participate in a relation.

Mapping Cardinations

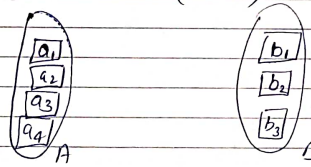
Cardinalities define the no. of entities of an entity set participate in a relationship set.

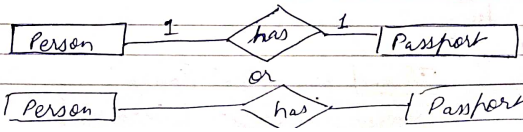
Most useful in describing binary relationship

Different types Of Cardinalities

i) One to One (1-1)
ii) One to Many (1-M)
iii) Many to One (M-1)
iv) Many to Many (M-M)

i) One to One (1-1)





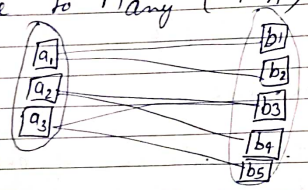
or

Saathi

Date __/__/____

- One entity from entity set A can be associated with atmost one entity of set B and vice-versa.
- Eg - A person has only one passport and a passport is given to one person.

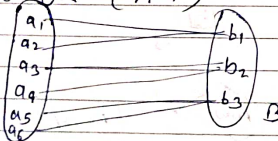
(2) One to Many (1-M)

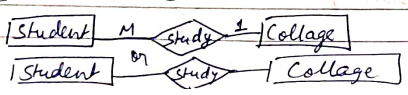


One Entity from entity set A can be associated with more than one entities of entity set B. However, an entity set from entity set B can be associated with atmost one entity.

Eg - Customers can place many order but order can't be placed by many customers.

3) Many to One (M-1)





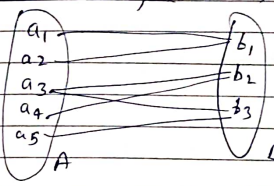
Page No.

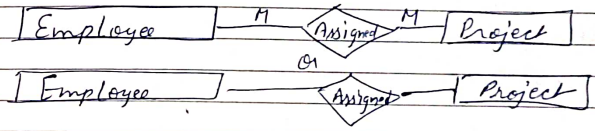
Saathi

Date __/__/____

More than one entities from entity set A can be ~~avoid~~ associated with almost one entity of entity B. However, an entity from entity set B can be associated with more than one entity from entity A.
eg:- Many student can study in a single collage but a student cannot study in many collage at the same time.

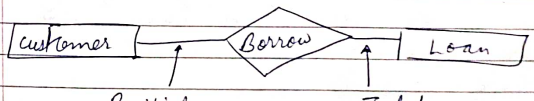
(4) Many to Many (M-M)





One entity from A can be associated with more than one entity of A and vice-versa.
eg - A employee can assign to many project and project can assign to many employee.

□ Participation Constraints



Paritial Participation

Total participation

Page No.

Total Participation

Each entity is involve and represented by double line.

Eg:-

Participation of loan in Borrower is total every loan must have a customer associated to it by a borrower.

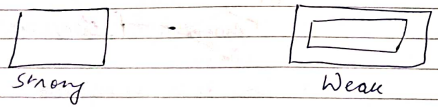
Partial Participation

Not all entity is involve in the relationship it is represented by single line.

eg:-

Participation of customer in borrower. A customer may have no loan.

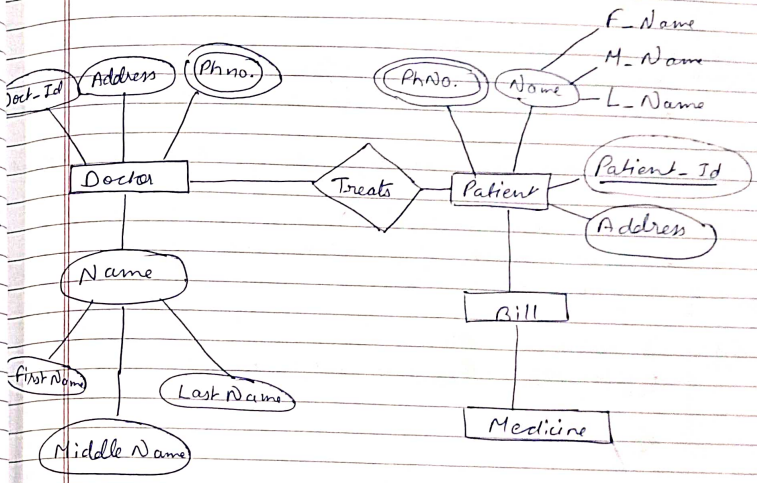
- Strong and weak entity set



Strong entity always have a primary key its existence is not dependent on any other set a set of strong entity.

Weak entity doesn't have a sufficient attributes to form a primary key. It is dependent on strong to ensure its existence set of weak entity is known as weak entity.

ER diagram for Hospital Management System



Saathi

Date __/__/____

Que. Draw an ER diagram for an university database application.

(a) A university have many departments, each department have multiple instructure one among them is head of department

(b) An instructure belongs only one departmen

(c) Each department offers multiple courses, each of which taught by single instructors

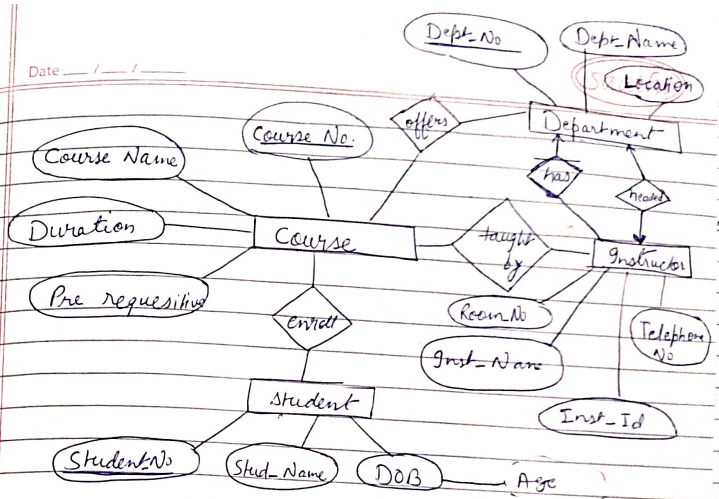
(d) A student may involve in many courses offers by different departmen

offers, enrolls / course \ student, taught by / course \ instructio

, headed by / dept \ instruction, has / instruction \ dept

Page No.

Date __/__/____



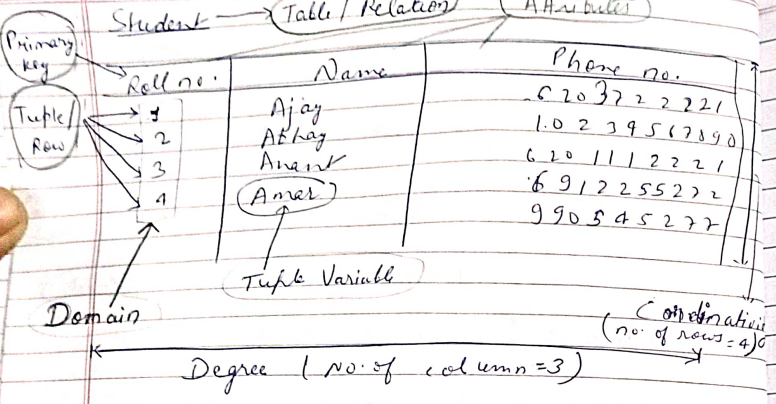
University Database Application

Page No.

Unit-2

Introduction To Relational Model

Student — (Table / Relation) (Attributes)



- The relational model is the theoritical basis of relational databases.
- The relational model is based on the concept of relations
- Relation is a mathematical concept based on the ideas of sets.
- After designing the conceptual model of database using the ER-diagram, we need a conceptual model which can implement using any RDBMS languages like Oracle, SQL, MySQL etc.
- RDBMS is a database management system that is based on the relational model as introduced by the E.F. CODD.
- Currently popular RDBMS include
  1) DB2 and informix Dynamic Server - from IBM
  2) Oracle & RDB - from oracle
  3) SQL server & MS Access - from Microsoft

- Domain :- A domain is the set of allowable values for one or more attribute, for ex - Phone no. should contain only number
- Relational Schema - It represents the name of the relation with its attributes.

Keys And Its Types in DBMS

Key - An attribute or set of attribute that uniquely identifies any record (or tuple) from the table.

Types

1) Super Key :-

Employee

Emp_Id	Name	Aadhar_No.	Email_Id	Dept_Id
1	Dev	123456789101	xyz@gmail.com	1
2	Shyam	555566667779	Shyam@gmail.com	2
3	Ram	999295679101	Ram@gmail.com	2
4	Allu	011122213315	Allu@gmail.com	3

1. { Emp_Id }
2. { Aadhar_No }
3. { Email_Id }
4. { Email_Id , Aadhar_No }
5. { Email_Id , Aadhar_No, Emp_No }
6. { Emp Id , Email_Id }

Date / /

Saathi

possible

- Combination of all attributes that can uniquely identify the rows (or tuple) in the given relation.
- Super key is a super set of candidate key
- A table may have many super keys.
- A super key may have additional attributes that are not needed for unique identity.

Emp_Id	Name	Aadhar_No	Email_Id	Dep_Id
1	Dev	123456789101	- -	1
2	Shyam	000011112112	- -	2
3	Ram	533945671911	- -	2
4	Pllu	- - -	- -	3

2] Candidate key

- Minimal Super key
- Super key with no redundant attributes

{Emp_Id}
{Aadhar_No}
{Email_Id}

- It is an attribute or set of attribute which can uniquely identify
- Candidate key is a minimal super key or super key with no redundant attributes
- It is called a minimal super key because we select a candidate key from a set of super key such that selected candidate key is the minimum attribute required to uniquely identify the table.

Page No.

Date / /

Saathi

- Candidate keys are defined as distinct set of attributes from which primary key can be selected.
- Candidate keys are not allowed to have null keys.
- Example of candidate key - {Emp Id}, {Aadhar_NO}, {Email_Id}

3] Primary key :-

- Primary key is of the candidate key chosen by the database designer to uniquely identify the tuples in the relation.
- The value of primary key can never be null.
- The value of primary key must always be unique.
- The value of this key can never be changed i.e. no updation is possible.
- The value of this key must be assigned when inserting a record.
- A relation is allowed to have only one primary key.
- Eg - {Aadhar_No}, {Emp-Id}

4] Alternate keys :-

- Out of all candidate key only one gets selected as primary key, remaining keys are known as alternate keys.
- Eg - {Aadharno}, {Email_Id}

5] Foreign keys :-

Page No.

Employee Table (Referencing relation)

Foreign key

Emp_Id	Name	Aadhar_No	Email_Id	Dept_Id
01	Ram	275277	ram@gm	1
02	Shyam	110022	sh@.	2
03	Alka	120352	Alka@	3
04	Gola	757522	Gola@	4

Department Table (Referenced relation)

Primary key

Dept_Id	Dept_Name
1	Sales
2	Marketing
3	HR

- A foreign key is a key used to link two tables together, it is an attribute or set of attribute in one table that refers to the primary key to the another table.
- The purpose of the foreign key is to referential integrity of the data. Parent
- Department table is master / primary table
- Employee table is foreign table / child table.
- Foreign key can take only those values which are present in the primary key of the referenced relation.

- Foreign key may have a name other than that of a primary key.
- Foreign key can take null values
- There is no restriction on foreign key to be unique.
- In fact foreign key is not unique for most of time.

6) Composite key :-

Cust_Id	Order_Id	Product_code	Product_Count
C01	001	P111	5
C02	012	P111	8
C02	012	P222	6
C01	001	P333	9

→ A key that has more than one attributes is known as composite key, also called Compound key.

→ Any of the above attribute cannot be taken as primary key, so combination of 2 attribute can uniquely identify this relation.

→ Eg :- {Cust_Id, product_code}

Saathi

Date __/__/____

Integrity Constraints Over Relations

1. Domain constraint
2. Entity integrity constraints
3. Referential integrity constraints
4. Key ~~consto~~ constraints

- Integrity Constraints are used to ensure accuracy and consiskency of data in a relational database.
- Integrity constraints are set of rules that the database is not permitted by to violate

TYPES

1. **Domain Constraint** :- Defines the domain or the valid set of values for an attribute. The datatypes of domains are string, time, date, currency, character, integer etc. Eg- Age of student contain Integer only.

2. **Entity integrity constraint** - States that primary key is not null. It ~~state~~ is because primary key value is used to identify the individual ~~rows~~ rows in relation and if the primary key has null value then we cannot identify those rows.

Saathi

Date __/__/____

3) **Referential integrity constraints** :-

Emp (Table 1) Related / Referencing Table

Foreign Key

Emp_Id	Emp_Name	Age	Dept_No
111	Mohan	21	1
112	Sohan	22	2
113	Rohan	33	3
114	Allu	47	5

Not allowed as Dept_No 5 is not defined as primary key in department table.

Primary key

Table 2

Dept_No.	Location
1	Mumbai
2	Delhi
3	Kolkata
~~5~~	~~Durg~~

- Referential integrity constraints specify between 2 tables.
- It is enforces when a foreign key references the primary key of a table.
- If a foreign key in table 1 refers to the primary key of table 2 either every value of foreign key in table 1 must be available in primary key value of table 2 or it must be null.
- Some more rules are we cannot delete a record from primary table if the matching record exist in a related table.
- You cannot change primary key value in the primary table if ~~that~~ ~~related~~ that record has related record.

- You cannot insert a value is a foreign key field of the related table that doesn't exist in the primary key of the primary table.
- However you can enter a null value in a foreign key specified that the records are unrelated.

→ Key Constraints

Stud_Id	Name	Age	Sem
111	Mohan	21	1
112	Rohan	23	2
113	Sohan	27	3
111	Ram	29	5

Not allowed because all rows must be unique

Key Constraints must contains all the values of primary key must be unique The value of Primary key must not be null

Integrity Contraints in SQL

i) UNIQUE
ii) NOT NULL
iii) PRIMARY KEY
iv) CHECK
v) DEFAULT
vi) FOREIGN KEY

DDL, create statement

Create table Student (rollno number(30),
Sname varchar2(20), Sem number(1),
branch varchar2(20), marks number(2),
Pno number(10));

Data Definition Language (DDL)
It is used for creating and modifying the database objects such as table indices, views and users.
DDL commands:-
- Used to define the structure and schema of the database.
- All DDL commands permanently saves all the changes in the database.
- Following are DDL commands

1) CREATE - To create new table / database.
2) ALTER - To alter / modify structure of table
3) DROP - To delete a table from database.
4) TRUNCATE :- To delete all records from table.
5) RENAME :- To rename a table.

Saathi

Date __/__/____

Relation Algebra

It is a procedural query language which make takes a relation as an input & generates a relation as an output.

It is a language for expressing relational database queries. It uses operators to perform queries, an operator can be either unary or binary.

Types of operators in relational algebra

1. BASIC / Fundamental operators
2. Additional / Derived operators

Basic / Fundamental Operators

1) Selection (σ)
2) Projection (π)
3) Union ($\cup$)
4) Set difference ($-$)
5) Cartesian Product ($\times$)
6) Rename (ρ)

Page No

Saathi

Date __/__/____

2) Additional / Derived operators

a) Natural Join ($\bowtie$)

b) Left, Right, Full, Outer Join ($\bowtie$, $\bowtie$, $\bowtie$)

c) Set intersection ($\cap$)

d) Division ($\div$)

e) Assignment ($\leftarrow$)

(1) Selection Operator

$\sigma_{\theta}(R)$ or $\sigma_{condition}(relational\ name)$

(select tuples from student table whose age is greater than 17)

$[\sigma_{age>17}(student)]$

select student whose name is D

$\sigma_{name="D"}$

select student whose age is greater than 17 and who lives in Delhi

$\sigma_{age>17 \wedge address="Delhi"}(student)$

select tuples from a relation books where subject is Database and Price is 450 or have a publication year after 2010.

$\sigma_{subject="Database" \wedge Price=450 \vee Publication\_year=2010}$

Page No

Saathi

Date __/__/____

2) Projection operator (Π) - Selection operator works on rows and projection works on column. It projects (displays) the particular column (attribute) from a relation.

Def Notation:- $\Pi_{A_1, A_2, A_3 \ldots A_n}(r)$

$\Pi_{(\text{Attribute List})}(\text{relation name})$

- Duplicate rows are automatically eliminated from the result.
- The SQL select command correspond to Project (Π)

Que. Display the columns, roll no and name from student.

→ $\Pi_{\text{roll no, name}}(\text{student})$

Que. Display the age of students in student table

→ $\Pi_{\text{age}}(\text{student})$

Que. Display the roll no. & name of the students whose age is greater than 17.

→ $\Pi_{\text{roll\_no, name}}(\sigma_{\text{age}>17}(\text{students}))$

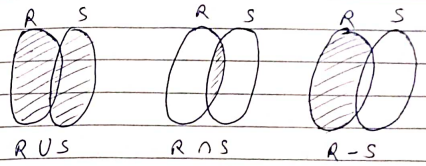
Page No.

Saathi

Date __/__/____

Set Operator in relational algebra

1) Union (U) { Fundamental operator }
2) Set intersection ($\cap$) { Derived operator }
3) Set difference (-) { Fundamental operator }



R

A	B
1	2
2	4

S

A	B
1	2
5	6

R U S

A	B
1	2
2	4
5	6

R ∩ S

A	B
1	2

R - S

A	B
2	4

S - R

A	B
5	6

Que. Display the name of branches in which project 121 or 122 or both are running.

Sol^n $\frac{121}{P}$ RL → Relational algebra

RL $\Pi_{\text{branch}}(\sigma_{\text{PNO}=121}(\text{student})) \cup \Pi_{\text{branch}}(\sigma_{\text{PNO}=122}(\text{student}))$

Page No.

Saathi

Date __/__/____

SQL :-

```
Select distinct branch from student
where Pno = 121 union
Select distinct branch from
student where Pno = 122;
```

Qu. Display the name of branches in which project 121 & 122 are running.

Sol^n RL :-

$$\Pi_{branch}(\sigma_{Pno=121}(student)) \cap \Pi_{branch}(\sigma_{Pno=122}(student))$$

SQL :-

```
Select distinct branch from student
where Pno = 121 intersect
select distinct branch from
student where Pno = 122;
```

Qu. Display the names of branches in which student got 50 marks & project 122 is running.

Sol^n RL :-

$$\Pi_{branch}(\sigma_{marks=50}(student)) \cap \Pi_{branch}(\sigma_{pno=122}(student))$$

SQL :-

```
select distinct branch from student
where marks = 50 intersect
select distinct branch from
student where Pno = 122;
```

Page No.

Saathi

Date __/__/____

Qu. Display the names of branches in which project 121 is running but project 122 is not running.

Sol^n RL :-

$$\Pi_{branch}(\sigma_{Pno=121}(student)) - \Pi_{branch}(\sigma_{Pno=122}(student))$$

SQL :-

```
select distinct branch from student
where Pno = 121 minus / except
select distinct branch from
student where Pno = 122.
```

Cartesian Product

R_1

A	B
α	1
β	2

R_2

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

$R_1 \times R_2$

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

If we write $\sigma_{A=C}(R_1 \times R_2)$ then output will be

A	B	C	D	E
α	1	α	10	a
β	2	β	10	a
β	2	β	20	b

It is made where value of a = c

Page No.

If we write $\Pi_A(\sigma_{D=20}(R_1 \times R_2))$ then o/p will be.

A	B	C	D	E
α	1	β	20	
β	2	β	20	

A
α
β

Que. $\sigma_{author = "Korth"}$ (Books × Articles)

Sol^n Above relation shows all the book & articles written by "Korth".

Rename Operator

- Used to rename the output of a relation
- Suppose we want to do cartesian product then same product then one of the table should be renamed with another name.

- R×R (Ambiginity will be there)

R

A	B
α	1
β	2

R×R

R·A	R·B	R·A	R·B
α	1	α	1
α	1	β	2
β	2	α	1
β	2	β	2

- If we name second relation R to S then we write $R \times \rho_S(R)$

R

A	B
α	1
β	2

S

A	B
α	1
β	2

$R \times \rho_S(R)$

R.A	R.B	S.A	S.B
α	1	α	1
α	1	β	2
β	2	α	1
β	2	β	2

- $\rho_{x(A_1, A_2, A_3, \ldots, A_n)}(E)$, Here E relation is renamed to x and attributes are changed to $A_1, A_2, A_3 \ldots A_n$

- $\rho_{A_1, A_2 \ldots A_n}(E)$, Here attributes get renamed to $A_1, A_2 \ldots A_n$

Que. Query To Find The Female students from student relation & rename the relation student as Female_student and the attributes of students - RollNo, Sname as Sno, Name

Student

RollNo	SName	Gender
1	Neha	F
2	Suman	F
3	Ram	M
4	Shyam	M
5	Rohan	M

Rename →

FemaleStudent

Sno	Name
1	Neha
2	Suman

Sol: $\rho_{\text{Female}(Sno, Name)}\left(\Pi_{Fatherno, Sname}\left(\sigma_{Gender='F'}(Student)\right)\right)$

Que: Query to Rename attributes Name, Age of table Person to N, A

Sol: $\rho_{(N,A)}(Person)$

Que: Query to rename the table name "project" to "work" and its attributes to P, Q, R

Sol: $\rho_{Work(P,Q,R)}(Project)$

Que: Query to rename the 1st attribute of the table "Student" with attributes A, B, C to P

Sol: $\rho_{(P,B,C)}(Student)$

Que: Query to rename the table name "Loan" to "L"

Sol: $\rho_{L}(Loan)$

Banking Example

branch (branch_name, branch_city, asset)

customer (customer_name, customer_street, customer_city)

account (account_number, branch_name, balance)

loan (loan_number, branch_name, amount)
(F)

depositor (customer_name, account_number)
(F) (F)

borrower (customer_name, loan_number)
(F) (F)

Que: Using Selection operator find all loans made at "Perryridge Branch"

Sol: $\sigma_{branch\ name = \text{"Perryridge Branch"}}(Loan)$

Que: Find all loans of over $1200

Sol: $\sigma_{amount > 1200}(Loan)$

Que: Find all tuples who have taken loans of more than $1200 made by the Perryridge Branch

Sol: $\sigma_{branch\ name = \text{"Perryridge Branch"} \cap amount > 1200}(Loan)$

Saathi

Date __/__/____

Que. Find all loans no. and the amount of a loans.

Sol: $\Pi_{loan\_number, amount}$ (Loan)

Que Find those customers who lives in "Harrison"

Sol: $\Pi_{customer\_name}$ ($\sigma_{customer\_city = "Harrison"}$ (customer))

Que Find the loan_no. for each loan of an amount > $1200.

Sol: Π_{loan_number} ($\sigma_{amount > 1200}$ (loan))

~~Find those customers who lives in~~

Union

Find the names of all the customers who have a loan, account or both from the bank

Sol: $\Pi_{customer\_name}$ (borrower) $\cup$ $\Pi_{customer\_name}$ (depositor)

Qn

Find the names of all the customers who have loan & a account from the bank.

Sol: $\Pi_{customer\_name}$ (borrower) $\cap$ $\Pi_{customer\_name}$ (depositor)

Page No.

Saathi

Date __/__/____

Que. Find the name of all customers who have a account but no loans from the bank.

Sol: $\Pi_{customer\_name}$ (depositor) $-$ $\Pi_{customer\_name}$ (borrower)

Que. Find the names of all customers who have a loan at the "Perryridge branch".

Sol: $\Pi_{customer\_name}$ ($\sigma_{branch\_name = "Perryridge"}$ ($\sigma_{borrower.loan\ number = loan.loan\ number}$ (borrower x loan)))

Que Find the names of all customers who have a loan at the "Perryridge branch" but don't have an account at any branch of the bank.

Sol: $\Pi_{customer\_name}$ ($\sigma_{branch\_name = "Perryridge"}$ ($\sigma_{borrower.loanno. = loan.loanno}$ (borrower x loan))) $-$ $\Pi_{customer\_name}$ (depositor)

Page No.

JOIN OPERATOR AND ITS TYPES

- Cartesian product of 2 relation gives us all the possible tuples that are paired together.
- But it might not be feasible in certain cases to take a cartesian product where we encounter huge relations with thousand of tuples having a considerable large no. of attributes.
- Query also become lengthy so we use join operator.

Join = Cartesian product + selection

Symbol ⋈

$$A \bowtie_C B = \sigma_C (A \times B)$$

C — condition; σ_C — selection; $(A \times B)$ — cartesian

R

A	B
1	a
2	b

S

B	C
a	3
b	4

R × S

A	R.B	S.B	C
1	a	a	3
1	a	b	4
2	b	a	3
2	b	b	4

R ⋈ S

Natural Join

A	B	C
1	a	3
2	b	4

Page No.

i) Inner Join 2) Outer Join

i) Inner Join :- It contains those tuple that satisfy the matching condition

ii) Outer Join :- It contains those tuple that satisfy the matching condition along with some or all tuples that satisfy the matching condition. It contains all rows from either 1 or both relation.

Types :-

Outer Join	Inner Join
Extension of join	a) Theta Join (θ)
→ Left outer join	b) Equi Join
→ Right outer join	c) Natural Join
→ Full outer Join	

1) Theta Join / conditional Join :-

$< \ > \ <= \ >= \ = \ \neq$

Notation $A \bowtie_\theta B$

where θ = predicate / condition

$$A \bowtie_\theta B = \sigma_\theta (A \times B)$$

S1

Sid	Name	Rating	Age
22	Dustin	7	45
31	Lubber	8	55
58	Rusty	10	35

R1

Sid	bid	day
22	101	10/10/96
58	103	11/12/96

- $S_1 \bowtie_{S1.Sid < R_1.Sid}$

S1.Sid	Name	Rating	Age	R_1.Sid	bid	day
22	dustin	7	45	58	103	10/12/96
31	Lubber	8	55	58	103	10/12/96

$S_1 \bowtie_{S1.Sid < R_1.Sid}$ Equivalent to $\sigma_{S_1.Sid < R_1.Sid}(S_1 \times R_1)$

- $S_1 \bowtie_{S_1.sid = R_1.Sid}$

S_1.Sid	Name	Rating	Age	R_1.Sid	bid	day
22	Dustin	7	45	22	101	10/10/96
58	Rusty	10	35	58	103	10/12/96

Natural Join

$A \bowtie B$

Comparison Operator not used.

Natural Join can only be performed if there is atleast one common attribute that exist between two relation.

- Attributes must have the same name & domain
- Doesn't use any comparison operator.
- Notation : $A \bowtie B$
- The result of the natural join is the set of all combination of tuples in two relations that are equal to their common attribute name.
- Natural join is equal to Cartesian product + Selection + Projection.
- $r \bowtie s$ is defined as $\Pi_{r.A, r.B, r.C, s.D, s.E}(\sigma_{r.B=s.B \wedge r.D=s.D}(r \times s))$

in which $r = (A, B, C, D)$ and $s = (B, D, E)$

Resulting schema $r \bowtie s = (A, B, C, D, E)$

r

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

s

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	ε

$r \bowtie s$

A	B	C	D	E
α	1	α	a	α
α	1	γ	a	γ
α	1	γ	a	γ
δ	2	β	b	δ
α	1	α	a	γ

Course

Cid	Course	Dept
CS01	Database	CS
ME01	Mechanics	ME
EE01	Electronics	EE

HOD

Dept	Head
CS	Rohan
ME	Sam
EE	Zia

Cid	Course	Dept	Head
CS01	Database	CS	Rohan
ME01	Mechanics	ME	Sam
EE01	Electronics	EE	Zia

$\Pi_{Cid, course, course.Dept, Head}(\sigma_{course.dept = HOD.dept}(course \times HOD))$

Que $S_1 \bowtie R_1$

S_1

Sid	Name	Rating	Age
22	Dustin	7	95
31	Lubber	8	55
58	Rusty	10	65

R_1

Sid	bid	day
22	101	10/10/23
58	103	10/12/23

$S_1 \bowtie R_1$

Sid	Name	Rating	Age	Bid	day
22	Dustin	7	95	101	10/10/23
58	Rusty	10	65	103	10/12/23

Que. Find the names of all customer who have a loan at the bank along with the loan no. and the loan amount.

Sol^n Using Cartesian Product

(Borrower x loan)

$\sigma_{borrower.Loan\,no = loan.Loan\,no.}$ (borrowers x loan)

Ans $\Pi_{customer\_name, amount, loan.loan\_no}(\sigma_{b}$

Page No.

$\Pi_{customer\_name, loan.loan\_no, amount}(\sigma_{borrower.Loan\,no = loan.Loan\,no}$ (borrowers x loan))

Using Natural Join

$\Pi_{customer\_name, loan.loan\,no, amount}$ (borrower $\bowtie$ loan)

Que. Find the names of all branches with customers who have an account in bank and who live in "Harrison"

Sol^n

Que. Find all customers who have both a loan & an account at the bank.

Sol^n $\Pi_{customer\_name}$ (borrower $\cap$ depositor)

Que. Find the names of all customers who have a loan at the Perryridge branch.

$\Pi_{customer\_name}(\sigma_{branch\,name = "Perryridge}$ (borrower $\bowtie$ loan))

Page No.

Outer Join

Outer Join = Natural Join + Extra information (from left table, Right table, both table)

Three kinds of outer joins are :-

- Left outer join ⟕
- Right outer join ⟖
- Full outer join ⟗

Course

Cid	Course
100	Database
101	Mechanics
102	Electronics

HOD

Cid	Name
100	Rohan
102	Sara
104	Jiya

Course ⟕ HOD

Cid	Course	Name
100	Database	Ram
101	Mechanics	Shyam
102	Electronics	Sandy

Course ⟖ HOD

Cid	Course	Name
100	Database	Rohan
102	Mechanics	Sara
104	NULL	Jiya

Course ⟗ HOD

Cid	Course	Name
100	Database	Rohan
101	Machines	NULL
102	Electronics	Sara
104	NULL	Jiya

- In left outer Join all the tuples from the left relation R_1 are included in the resulting relation the tuple of R_1 which donot satisfy join condition will have values as NULL for the attribute of R_2.
- In Right outer join all the tuples from the right R_2 are included in the resulting relation to tuples of R_2 which don't satisfy join condition will values as NULL for the attributes of R_1.
- In Full outer Join all the tuples from both left relation R_1 & Right relation R_1 are included in the Resulting in the tuples of both relation R_1 & R_2 which donot satisfied join condition there respected unmatched attributes are made NULL.
- Division Operator (÷ or /)

A

x	y
a	1
b	2
a	2
d	4

÷

B

y
1
2

⇒

A ÷ B

x
a

A	
S.no	P.no
S_1	P_1
S_1	P_2
S_1	P_3
S_1	P_4
S_2	P_1
S_2	P_2
S_3	P_2
S_4	P_2
S_4	P_4

B1
P.no
P_2

B2
P.no
P_2
P_4

B3
P no
P_1
P_2
P_4

A / B2
S no
S_1
S_2
S_3
S_4

A / B2
S.no
S_1
S_4

A / B3
S no
S_1

It is a derived operator
Suited to query that include the keyword 'all' 'every' like 'atall' 'for all', 'in all', 'at every' 'for every' or 'in every'.

Eg:- Find the Person that has account in all banks of a particular city.

Find the student who have register for every fest.

Division operator can be applied if and only if
- attributes of B is proper subset attribute of A.
- The relation return by will return those tuples from relation A which are associated to every B's tuples.

□ Assignment Operator

$$A \div B = \pi_x(A) - \pi_x((\pi_x(A) \times B) - A)$$

then using assignment operator we may write:

$$temp1 \leftarrow \pi_x(A)$$
$$temp2 \leftarrow \pi_x((temp1 \times B) - A)$$
$$result \leftarrow temp1 - temp2$$

Assignment operator provides a convinience to express complex query.

Assignment must always made to a temporary relation variable.

Modification of Database

i) Deletion
ii) Insertion
iii) Updation

□ Deletion
Expressed as $r \leftarrow r - E$ where r - relation
E - relational algebra expression

eg:-
Delete all account records in the Perryridge branch?

$$account \leftarrow account - \sigma_{branch\_name = "Perryridge"}(account)$$

Date __/__/____

Saathi

Delete all loan records with amount in the range 0-50.

$\Rightarrow$ $loan \leftarrow loan - \sigma_{amount \geq 0 \text{ and } amount \leq 50}(loan)$