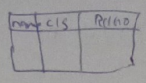
Database

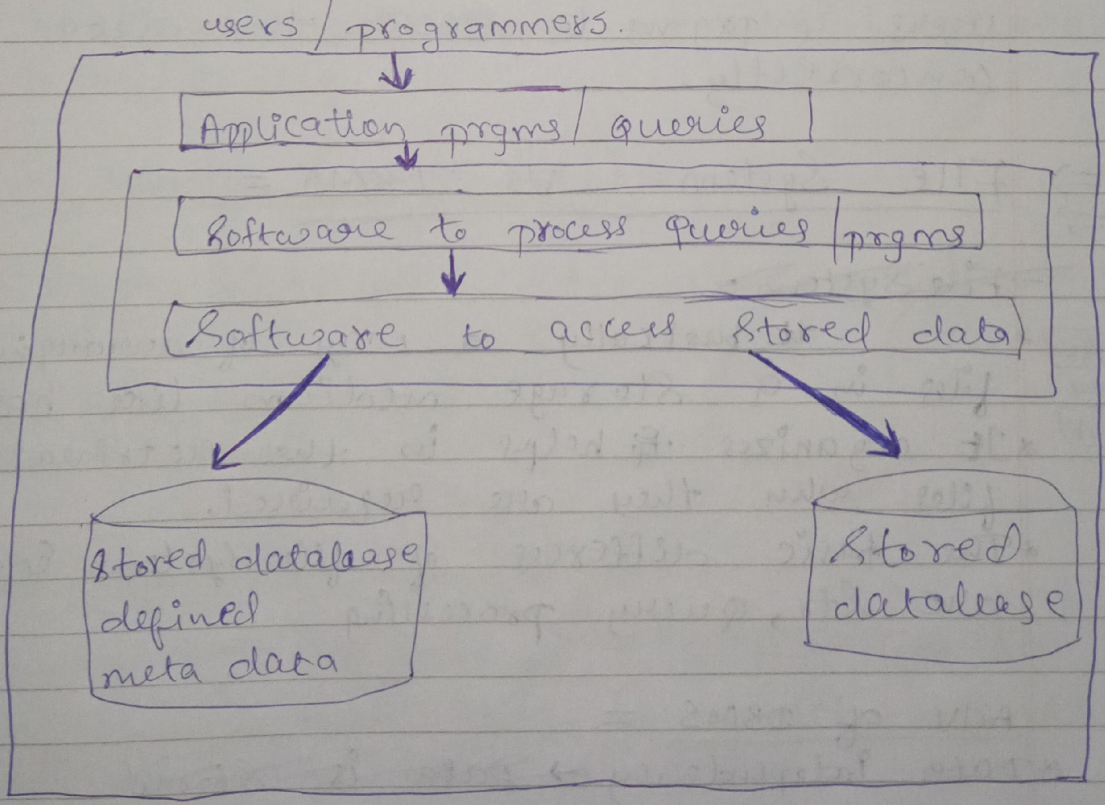


classmate
Date
Page

DBMS

* Data = Raw facts / unprocessed facts.
* Info... = processed facts / meaningful data.
* Database = It is an integrated collection of logically related data into an common pool that provides data for many application.
* Database management system = It is a collection of prgms that enables user to create & maintaine a database. Therefore a general purpose Software System that facilities the process defining, constructing, manipulating & Sharing data bhsed various applications.

*



(D) → database → easy to access data.

=> Functionality of (D) :-

1) Define → Defining a (D) involves specifying the datatypes, str. & constraints for the data to be stored in the (D).

2) Construct → (adding data)
Constructing the (D) is the process of storing the data itself on some storage medium (i.e) control by DBMS.

3) Manipulating a (D) → It includes such (s as querying the (D).
(update, alt, querying)

4) Sharing the (D) → It allows multiple users & prgms to access the data concorrectly.

=> File System VS DBMS =

~~File System~~

* It basically a way of arranging the files in a storage medium like harddisk
* It organizes & helps in the retrival of files when they are required.
* The basic differces of file system & DBMS are str, query processing

Adv of DBMS =
* Data Independency → ~~Data is stored~~
* Efficient file access
* Data Security

Root node → parent (N) → child node

Disadv of DBMS →
* over head cost of DBMS
* Impact of failures.

=> Database User =

1) Naive user
2) Application prgmer
3) Sophisticated user (query writing people)

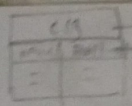
=> DBA ((d) Administrator) =
* A DBA is the info technition responsible for directing & performing all activities related to maintaining (d) enumt.
* A DBA makes sure an organisation (d)s & related applications operate functionally & efficiently.

=> (d) Models (design / str) =

① Hierarchical (m) tree like str (one to many)
② Ntwrk (N) graph like str (many to many)
③ Entity relationship
④ Relational

1) Hierarchical (m) =

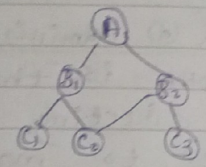
* Datas are organised in a tree like str with a single root node to which all the othr datas are linked.
* Hierarchy starts from root node & expands like a tree adding child nodes to parent nodes.



* Child node will only have a single parent node.
* It is a 1 to many relationship.

2) Network (m) =

* It is a many to many relationship.
* Datas are organised in a graph like str & are allowed to have more than 1 parent node.



3) Entity-Relationship (m) =

* Relationships are created by dividing obj of interest into entities & its characteristics into attributes.
* Diff entities are related using relationships.

Student		
Name	Cls	Rollno.

Entity
attributes.

4) Relation (m) =

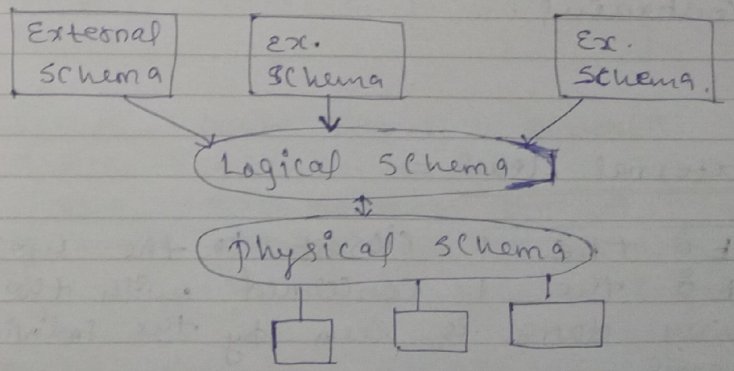
* Data is organised in 2D tables & the relationship is maintains by storing a common field.
* Basic str of data in a relational model is tables.

=> Schemas & Instance =

* Schemas = Discription of a database [overall design of a db] which is specified during (d) design & it is not expected to change frequently.
  It specifies the names of the entity & attributes & the relationship among this entities.

* Instance =
  The data in the (d) at a particular movmnt in time → (d).
  It is also → the crnt set of occurance/ instance in (d).

=> 3 Scheme Architechture =



→ Physical schema =

* Internal level also → physical level. It is the 1 closest to physical storage.
* It is the 1 concern with the way the data is physically stored.
* The internal level has an internal schema which describes the physical storage str of db.
* The internal schema uses a physical data model & describes the complete details of data storage & access paths for the db.

→ Logical schema / Conceptual schema =

* The conceptual level has a conceptual schema which describes str of the whole database for a community of users.
* The conceptual schema hides the details of the physical storage str & concentrates on describing entities, relationship, user operation & contraints.

→ External level =

* It is the 1 closest to the users
* It is the 1 concerned with the way data is seen by the individual users

⇒ Data Independence =

* capacity to change the schema at 1 level without having the change the schema at the higher level → D.I
* 2 types of D.I →
  a) Logical D.I = used to separate external from the logical level
  b) physical D.I = used to separate logical levels from physical level.

⇒ Database Language =

Data def lang	Data manipulation Lang	Data control lang	Transmission control lang.
1) CREATE	SELECT	GRANT	COMMIT
2) ALTER	INSERT	REVOKE	ROLL BACK
3) DROP	UPDATE		
4) TRUNCATE	DELETE		
5) RENAME			

str
- create → create obj in db.
- Alter → Alter the str of db
- DRop → delte obj from db
- Truncate → remove all rec from table.
- Rename → rename an obj

I Data def lang = (DDL)

(t) → table

* DDL → Data Def Lang.
* used to define (d) str / pattern
* used to create schema, tables, Indexes, etc.

1) CREATE → used to create objects in (d).

II Data manipulation Lang = (DML)

* DML → Data Manip Lang.
* used for accessing & manipulating data in (d).

1) SELECT → retrive data from (d).
2) INSERT → Insert data into a ~~(d)~~ table.
3) Update → update existing data within a table.
4) DELETE → dlt a record frm (t).

III Data Control Lang = (DCL)

• (giving permission in (d).)
* used to control privilage in a (d).
1) GRANT → gives user access privilage to the (d).
2) REVOKE → takes back permission from user.

[grant → permission കൊടുക്കാൻ
revoke → " deney cheyyan.]

IV Transmission Control Lang = (TCL)

* used to run the changes made by DML statemnts.

1) COMMIT → Saves the work done (change ചെയ്തതിൻ ശേഷം കൊടുക്കാൻ)
2) ROLLBACK → Restores the (d) to the original (change ചെയ്യുന്നതിൻ മുൻ ഉണ്ടായിരുന്നതിലേക്ക് തിരികെ പോകും)

=> Interface = (I) (5 types)

* medium used to interact with DBMS.

1) Menu-based (I) = (menu select ചെയ്യാൻ)

* This (I) prsnt the user with list of options → menus

2) Form-based (I) = (form type)

* It displace a form to each other.
* users can fillout all forms entries to insert a new data, they can fillout certain entries

3) Graphical user (I) = (GUI)
(diagramatical (I))

It typically displace a schema (str of database) to the user in diagramatical form.

4) Natural lang (I) =

It accepts requests written in eng / some other lang.

5) Speech input & output =

limited use of Speech as an input query & speech as an answer to the question / result of request is becoming

(E) → Entity

common place.

⇒ Basic concept of Entity relationship Model :-

→ **Entity** :-

* It is a thing / sub in the real world that is distinguishable from all other obj.
* It has set of properties & the values for some set of properties may uniquely identify an entity.

→ **Attributes** :- (A).

* An (E) is represented by a set properties → (A).
* (A) are discriptive (pro) processed by each member of an (E) set.
* 6 types →
  1) composite (A) (name → 1st letter, 2nd letter)
  2) Simple (A) (A) cannot subdivide (A).
  3) Single valued (A)
  4) Derived (A)
  5) multi valued (A)
  6) Stored (A)

(1) Composite (A)	(2) Simple (A)
(A) that can be divided into further (A)s. eg → Name can be divided by 1st name, 2nd name.	(A) that cannot be divided into further (A)s. eg → Salary cannot be divided to other.

(3) Single valued (A)	(4) Multivalued (A).
Most (A)s have a single value for an particular (E). Such (A)s → S.V.(A). eg → age is a single valued (A) of a person.	It can have a set of values for an particular (E). eg → language of a person.
(4) Derived (A)	(5) Stored (A).
(A) that can be derived from other (A)s. eg → age of a person.	(A) from which the value of other (A)s are derived. eg → DOB of a person.

⇒ Entities =

* **Types** →
  1) Tangible → Physically
  2) Intangible → Logical

* **Entity Set** = Grp of entity having some (A).

* Entity type = Representing entity with a name.

eg → Student.

Rollno	name	Branch
101	Sam	CS
102	Ram	BCA
103	Sam	BBA

→ fig 1.1

① entity set → Student.

② entity type → 101 Sam CS
103 Sam BBA
(having common entity)

* entity type → 2 types.

a) Strong entity type = (E) type having key attribute.

b) weak E. type = (E) type not having a key (A).

eg → ① Strong E. type → 101, 103. (fig 1.1)

②

name	branch
Sam	CS
Ram	BCA
Sam	BBA

→ eg for weak E. type (no key (A))

fig 1.2