Date: 09/03/2025

CSE3121: Database Management System

Data: Data is simply a (value) of real world (things or object). Raw facts (কাঁচা তথ্য). Entities / Entity (এন্টিটি)

things or object can be form in (concrete) or (abstract)

concrete → person (Name)

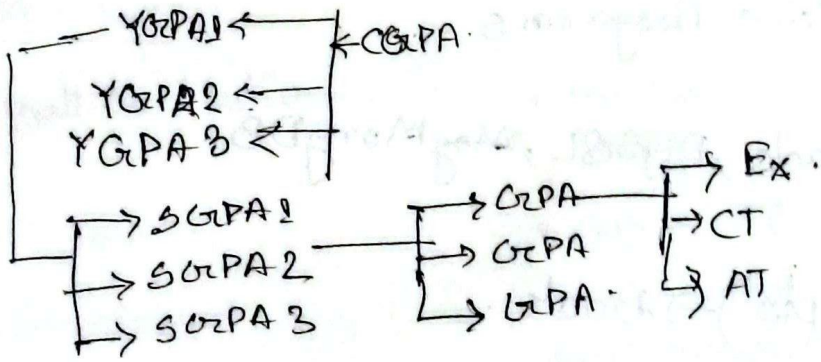
abstract → Account No.

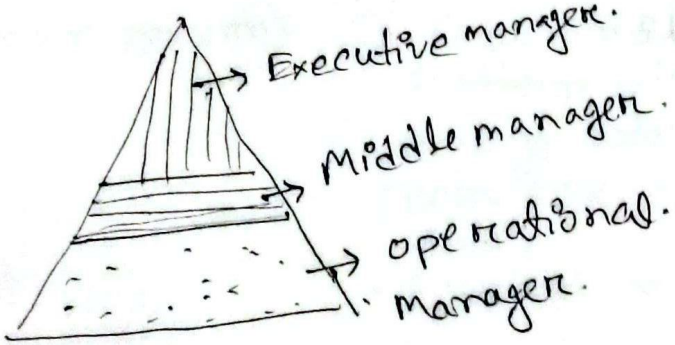
Example:

Cus Name	Acc.no.	Balance.
karim	A-101	9000
Rahim	H-314	01711 254379
User Name	House no.	Cell phone No.

Information: An information is the processed form of data which is meaningful and useful to the users. That is

Example: Result of a student (CGPA) | Data → Processed → Information.





Management level.

interrelated.

Database: – A database is a collection of data of a particular enterprise (Product base or service based)

Airbase	Database	Airbase
1. Land	1. Data Interaction (Entering)	বিমান ওঠানো নামানো নিয়ন্ত্রণ করা
2. Take off	2. Data Deletion.	
3. Update/ upgrade	3. Data update/upgrade.	
4. Searching	4. Data Searching or Query ↓ Most important.	
	→ SQL (Structure Query language)	

DBMS: A DBMS is one or more database and a set of programs to access data from those database. That is –

DBMS = { Database (s) + Programs.

Example : MS Access, Oracle, MySQL, ~~Mog~~MongDB.

(DataBase System Concepts) → Book.
↓
Abraham Silberschatz

Date: 11/03/2025

Application or use of Database/DBMS :-

1. Education → Hall management system.
   → Library management system.
2. Bank and Insurance
3. Health care.
   a. Medical → Patient Management.
   b. Clinic →
   c. Pharmacy → Drug management system.
4. E-Commerce.
5. Social Media.
   and so on.
6. Under sea Exploration
7. Space Exploration.
8. Sattelite Data Analysis.
9. Government Project.
10. Manufacturing company.

Structured and Well organized form of data → Database.

Database	Data Mining	Big Data.
• Small to Medium.	• Medium to Large	• Large to extremly Large.
• KB to GB	• GB to TB	• TB to EB.
• structured	• structured and semi structured	• Semi structured and unstructured.
• Data (main concern)	• Data and hidden pattern exist among data. (Knowledge about Data)	• Data and Intellegence.
• present	• present and past	• past present past and future.

Date: 08/04/2025

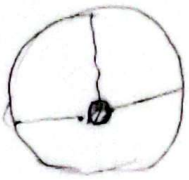
Introduction (Slide)

cv.docx
salary.xlsx
Presentation.pptx
} Typical file.

- Crammeenphone.accdb (access database)

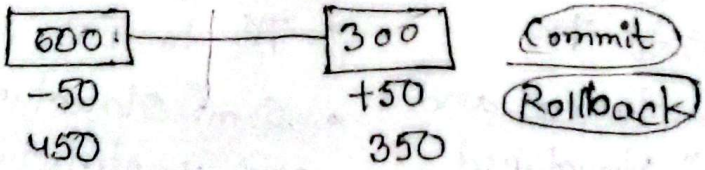
Major Drawbacks or disadvantage of Traditional or Conventional or typical file processing system.

1. Data Redundancy and Inconsistency

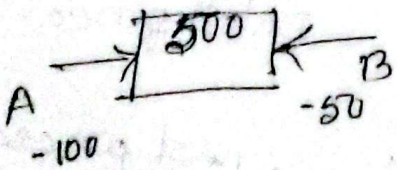


Data should be in center

2. Difficulty in Accessing Data.
3. Integrity problems.
   (constraint) (বাধা)
4. Atomicity problem.



5. Concurrent Access Anomalies.



Date: 10/04/2025

View of Data

Data Abstraction (ডাটা অ্যাবস্ট্রাকশন)

- Hide the details or complexity of the database from common computer users

* Three Level of Data Abstraction

- figure.

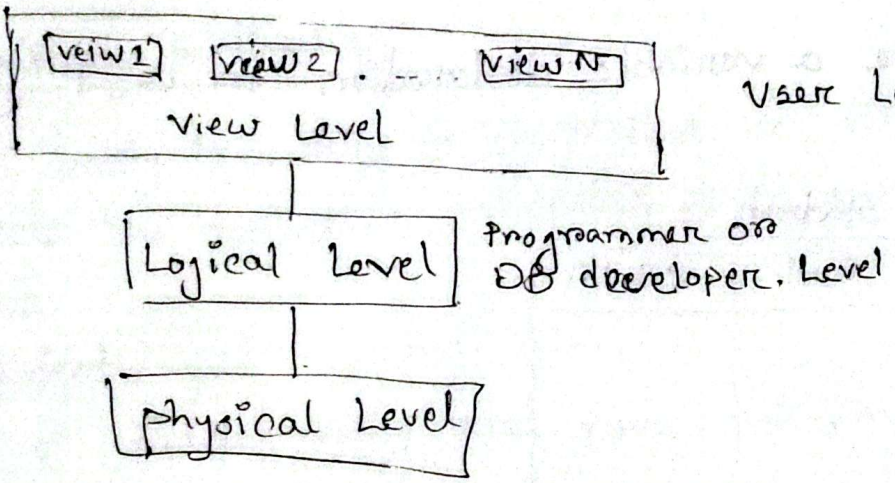


Fig: Three Levels of Data Abstraction.

Physical Level:

- Complex low level of data Abstraction. In this level How the database is actually stored in physical memory those are associated.

Logical Level:

- Logical level describes what data will be used and what is the relationships among those data. The another name of Logical level. is conceptual level.

View level:

- View level describes only a part of the database not the entire database. In this level complexity is totally removed.

Database schema and Instance

→ Schema importance

Database schema: The overall design of database is called its schema.

Schema is like a variable declaration in a programming language.

Student

Name	Roll	CGPA

Instance: An instance is the collection of data is stored in a database table at a particular moment.

Date: 13/04/2025

Data Model.

The structure of a database is called its data model. Different Data models are available in DBMS.

Some important models are.

1. Hierarchical Data model } → Backdated model.
2. Network Data Model.
3. Entity Relationship Data model } → Present days model.
4. Relational Data Model.

E-R Model (Entity Relationship)

Entity: An entity is a thing or object that is distinguishable from all other objects. The descriptive properties of an entity are called its attributes.

Example: Student

Attributes of student entity are name, roll, marks etc.

Relational Model:—

A Relational model of database is a table based model that is here data is organized in tabular form. Each table contains some rows and columns.

Example:—

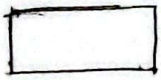
Student

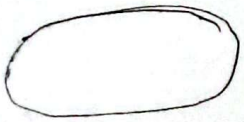
Name	Roll	Marks
Karim	101	75
Rahim	102	93.

Pictorial or Diagramatic Model.
→ ER Model.

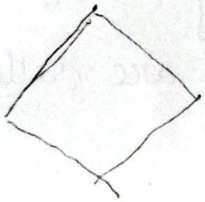
E-R Model Terms	Relational Model Term
Attribute	Feild.
Entity	Record. (Tuple)
Entity set.	Table (Relation)

ER Diagram.

 → Entity set. or strong Entity set.

 → Attribute.


↳ weak entity set.

 → Relationship set.

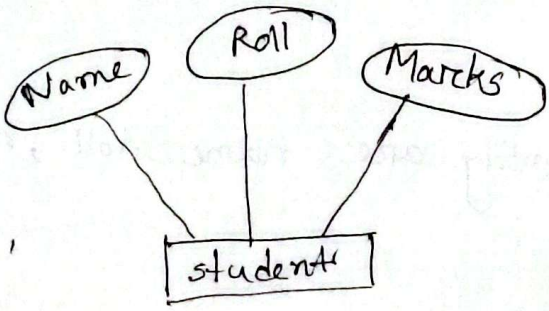


Fig: ER diagram of a student entity set.

Date: 15/04/25

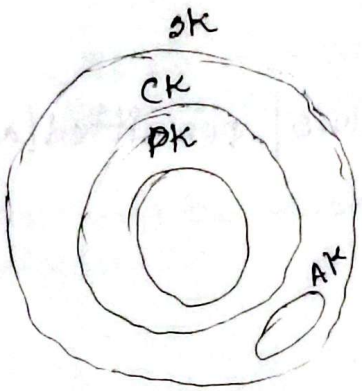
Database keys:-

Database keys are one or more feilds by which we can uniquely identify all the records from a table or relation.

Major

Types of keys:-

1. Super key.
2. Candidate key.
3. Primary key.
4. Alternative key.
5. Foreign key.



Super key: All possible combination of keys in a table are called.

super keys:

Examples:

{Roll}, {Roll, Name}
{Roll, Marks}
{Roll, Name, marks}

student

Name	Roll	Marks

Candidate key: Minimal super key is called a candidate key

Examples: {Roll}

student

Name	Roll	Marks	Phone no

* Primary key: The candidate key which does not contain any null value is called primary key.

Foreign key:

Alternative key:

[DOMAIN: Possible values | permitted/allowed values]

Date: 17/04/2025

Database Languages

Four types of Languages :—

1. Data Definition Language (DDL)
2. Data Manipulation Language (DML)
3. Data Control Language (DCL)
4. Data Transaction Language (DTL)

Data Definition Language (DDL): DDL defines the schema (skeleton of database.) of a database and its tables.

Student (Name, Roll, Marks)

Some SQL commands of DDL operation: CREATE, ALTER, DROP, TRUNCATE, RENAME, COMMENT.

Data Manipulation Language (DML):-

perform

By DML we can insert, update, delete, query and other important operations. Some DML commands:- INSERT, UPDATE, DELETE, SELECT.

There are Two types of DML.

1. Procedural DML (what data will be manipulated and how it will be manipulated) → (P/L SQL → TRIGGER, CURSOR)
2. Non Procedural DML.
   (It specifies only what data will be manipulated without knowing how will it manipulated)

Data Control Language (DCL):-

DCL is associated to the security of the database.

(DBA) → Database Administrators.

Some important DCL commands:- GRANT, REVOKE.

GRANT: Give access of the database.

REVOKE: To restrict or reject or disallow the access withdraw.

Data Transaction Language:-

DTL is associated to data transaction operation. Some important DTL commands:-

COMMIT: After a successful transaction.

ROLLBACK: After an unsuccessful transaction.

SAVEPOINTS: To save a ∞ COMMIT or ROLLBACK for future execution.

E-R Model:- Date: 24/04/25

Attributes:- Descriptive properties of an entity is called its attributes

Ex: For a student entity possible attributes are: Name, Roll, Registration no, session etc.

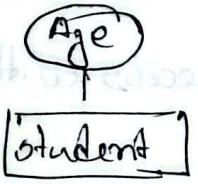
Types of attributes:

1. Simple attribute vs Composite attribute.
2. Single valued attribute vs Multivalued attribute.
3. stored valued attribute. vs Derived attribute
4. key valued attribute vs Non-key attribute.

Simple attribute vs Composite attribute.

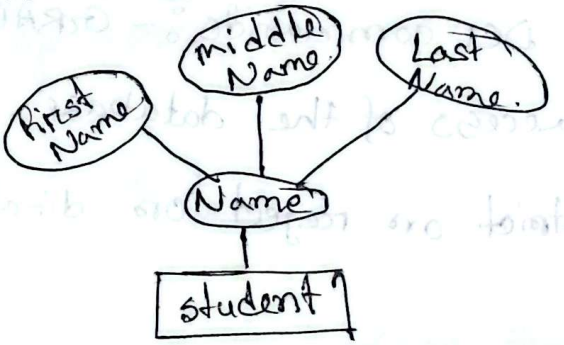
→ Cannot be divided into sub-parts

→ Ex: Age



→ Can be divided into sub parts.

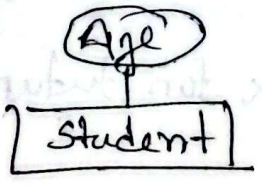
→ Ex: Name



Single valued vs Multivalued. Attribute.

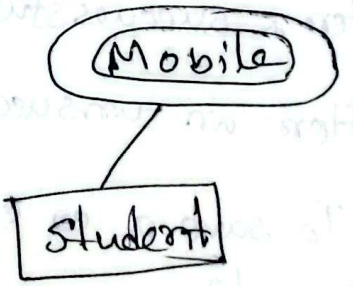
→ It contain only one value

→ Ex: Age



→ It contain more than one values.

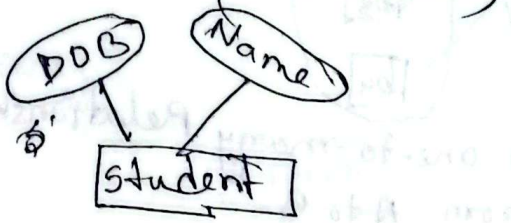
Ex: Cell phone no.



Stored attribute Vs Derived Attribute.

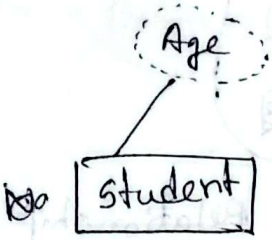
→ Need not to be derived value from other attributes

Ex: Name, Roll, DOB (Date of Birth)



→ Value will be derived from other attributes.

Ex: Age



Key Attribute Vs Non-key Attribute.

→ By this attribute we can distinguish all entities from an entity set

Ex: Roll, NID etc.

→ By this attribute we can not distinguish all the attributes from an entity set.

Ex: Name

Mapping Cardinality-

Mapping cardinality or cardinality ratio express the number of entities which can be associated to an entity of an entity set.

Types of mapping cardinality.

1. One to One (1-1)
2. One to Many (1-M)
3. Many to one (M-1)
4. Many to Many (M-M)

One to One:

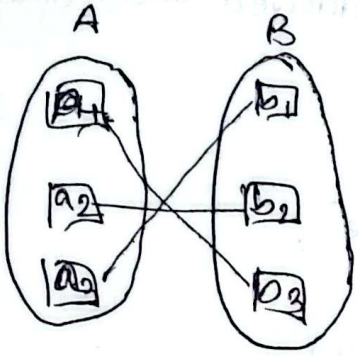


Fig: A one-to-one Relationship.

Ex: Student NID.

One-to-Many:—

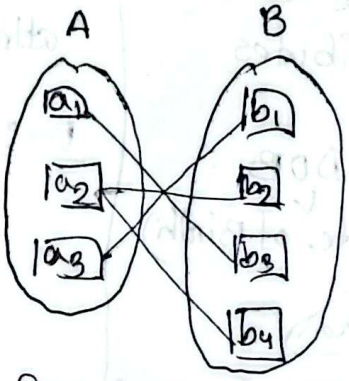


Fig: A one-to-many Relationship from A to B.

Ex: Account - customer

Many to One:

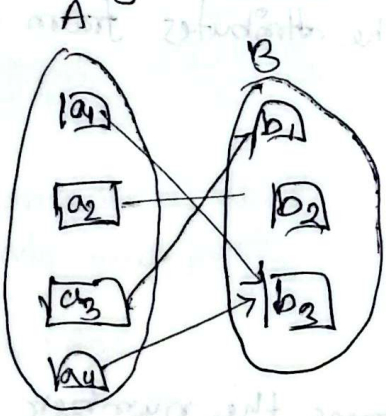


Fig: A Many-to-one Relationship from A to B.

Many-to Many

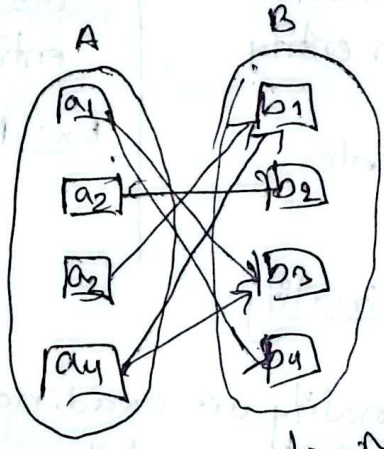


Fig: A Many-to-Many Relationship from A to B.

Degree of Relationship:

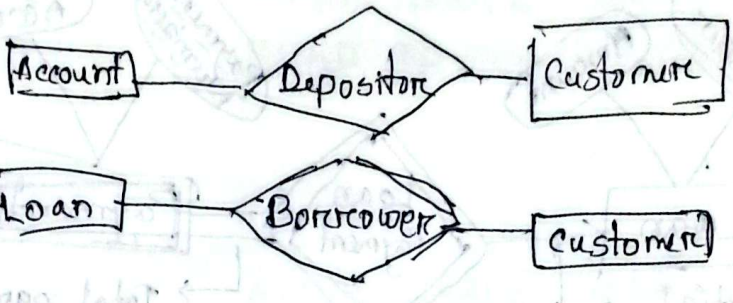
It express to the number of entity set which can be associated to an entity set via a relationship set.

Types of Degree of Relation ship:—

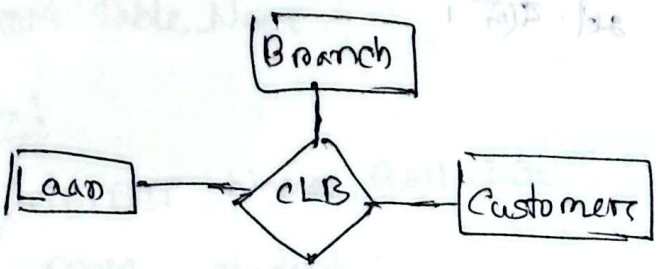
1. Unary. Relationship
2. Binary Relationship
3. Ternary Relationship
4. N-ary Relationship.

1. Unary:

2. Binary



3. Ternary:

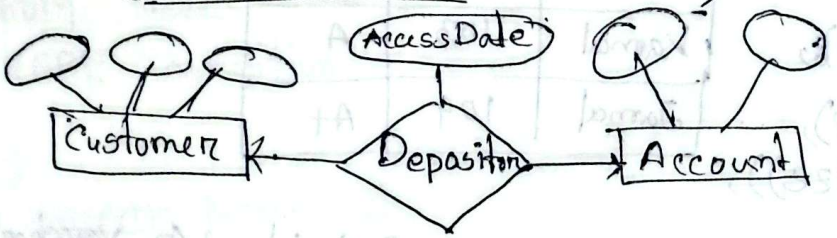


Date: 27/04/25

Customer (Customer Name, Customer street, Customer city)

Loan (Loan Number, Amount)

Account (Account number, Balance)

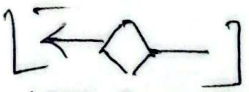


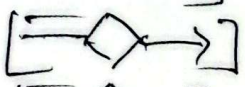
Customer Name	Accountno

- Descriptive attribute (It is a special attribute of a relationship set)

→ 1-1 Relationship.

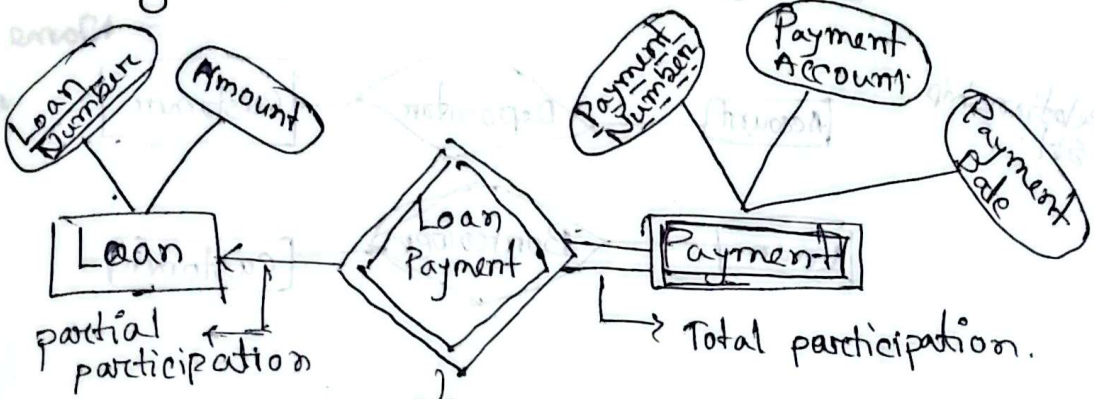
- Mapping cardinality or cardinality Ratio

 1-M Relationship

 M-1 Relationship.

 M-M Relationship.

→ Strong Entity set Vs. Weak Entity set



• Primary key
• partial key (weak entity set এর attribute) Its not complete

→ weak entity set ⇐ Indentifying Entity set এর।

P-1 5000 26.4.25
P-1 5000 26.4.25

Date: 20/4/25

Table Creation:-

Syntax to Create a Table.
(CReate Table command)

CREATE TABLE Table_Name
(feild 1 data type (Size),
Field 2 datatype (size), ...
Field N data-type(size));

student

Name	Roll	Gr
karim	101	A+
Rahim	102	B
kamal	103	A
Jamal	104	A+

Ex:

CREATE TABLE student (Name Varchar(25), Roll int, Gr varchar(3));

Describe student ;

Syntax to Insert Records in a Table.

INSERT INTO Table_Name
(Field 1, field 2, ... field N) ⇒ optional
VALUES (value 1, value 2, ... value N);

Ex:

```
INSERT INTO Student
(Name, Roll, LGr)
VALUES ('Karim', 101, 'A+');
```

Syntax to Query :-

```
SELECT Field1, Field2, ... FieldN
FROM Table_Name.
```

Ex:

```
SELECT Name, Roll, LGr
FROM Student;
```

or

```
SELECT * (All Field এর Data দেখাবে)
FROM Studen;
```

Query using Condition.

```
SELECT Field1, Field2, ... FieldN
FROM Table_Name
WHERE condition;
```

Ex:

```
SELECT Name, LGr.
FROM Student.
WHERE LGr = 'A+';
```

For Delete a collumn

```
ALTER TABLE table_Name.
DROP COLUMN colurmn_Name.
```

For ADD

```
ALTER TABLE Table_Name
ADD COLUMN columName-
        Datatype.
```

change:

```
UPDATE table_Name.
Set Colum_Name = value
WHERE condition.
```

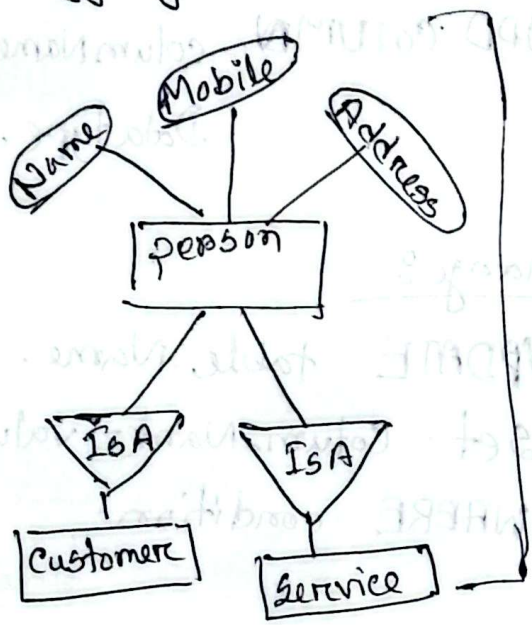
Date: 4/5/25

Homework:

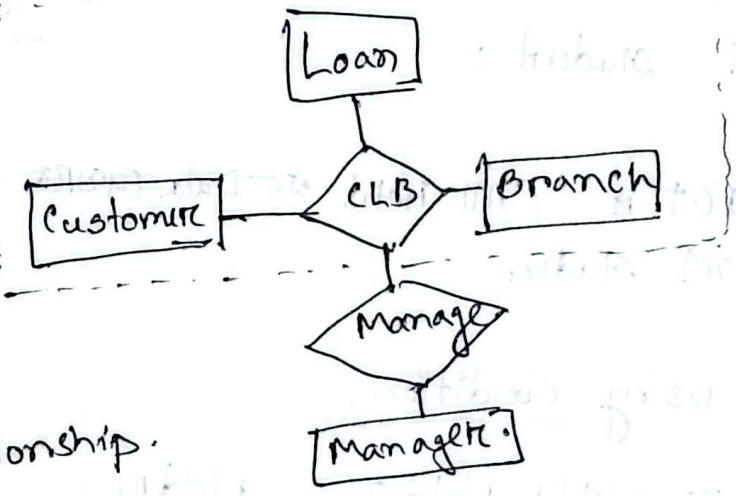
- Extended E-R features :—

→ specialization] → Inheritance.
→ Generalization]
→ Aggregation: Relationship is used as an entity set.

- specialization (Top down approach)
- Generalization (Bottom up approach)



- Top down approach
- Bottom up approach
- Specialization.



- Relationship among relationship.
(Aggregation এর মূল theme.

New chapter: —

Relational Algebra (Input Relation, output Relation)

- Formal query language.
- Procedural query language.
- Father E.F codd (Father of DBMS)
- 1970s
- Predecessor of SQL.

Relational Algebra Operations:

1. Basic or fundamental operations.
2. Additional or Derived operation.

Basic or fundamental operation:

→ Select, Project, Union, set difference, Cartesian product, Rename
(Select → where; Project → select; Cartesian product → Cross product.)

Additional or Derieved operation:

→ Intersection, join, Division, Assignment.

[Rename, assignment কাজের নাই]

Questions: এগুলো কেন Additional or derived বলা হয় -

$A \cap B = A - (A - B)$

- Join (very important)
  ↳ derieved using cartesian product.

Fundamental operations

- Select
- Project
- Rename

] Unary operation

Depositor

Cus Name	Acc.No.
Karim	A-101
Rahim	A-102
Kamal	A-103

Borrower

Cus Name	Loan No
Karim	L-101
Rafiq	L-102
Shafiq	L-103

Customer

cus Name	Cus city	Mobile.
Karim	Dhaka	017...
Rahim	Rajshahi	019....
Kamal	Khulna	018....

Select

$\sigma_{cus\ city = "Dhaka"}$ (Customer)

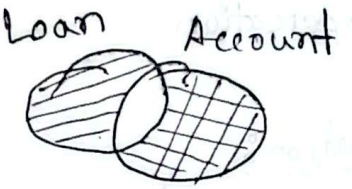
Project.

~~$\sigma_{cus\ city = "Dhaka"}$ (customer)~~

$\Pi_{cus\ Name}(\sigma_{cus\ city = "Dhaka"}(customer))$

Union

- Find out the customers who have a loan or an account or both.



A ∪ B

- $\Pi_{cusName}$ (Depositor) ∪

$\Pi_{cusName}$ (Depositor) ∪ $\Pi_{cusName}$ (Borrower)

Set Difference

$\Pi_{cusName}$ (Depositor) − $\Pi_{cusName}$ (Borrower)

- Find out the customers who have an account but no loan.
- Find out the customer who have loan but not an account.
- Find out the customer who have both loan and account.

Date: 05/05/25

Branch (Branch Name, Branch City, Asset)

Customer (Customer Name, Cus street, Cus City)

Account (Branch Name, Account No, Balance)

Loan (Branch Name, Loan no, Amount)

Depositor (Customer Name, Account No)

Borrower (Customer Name, Loan No)

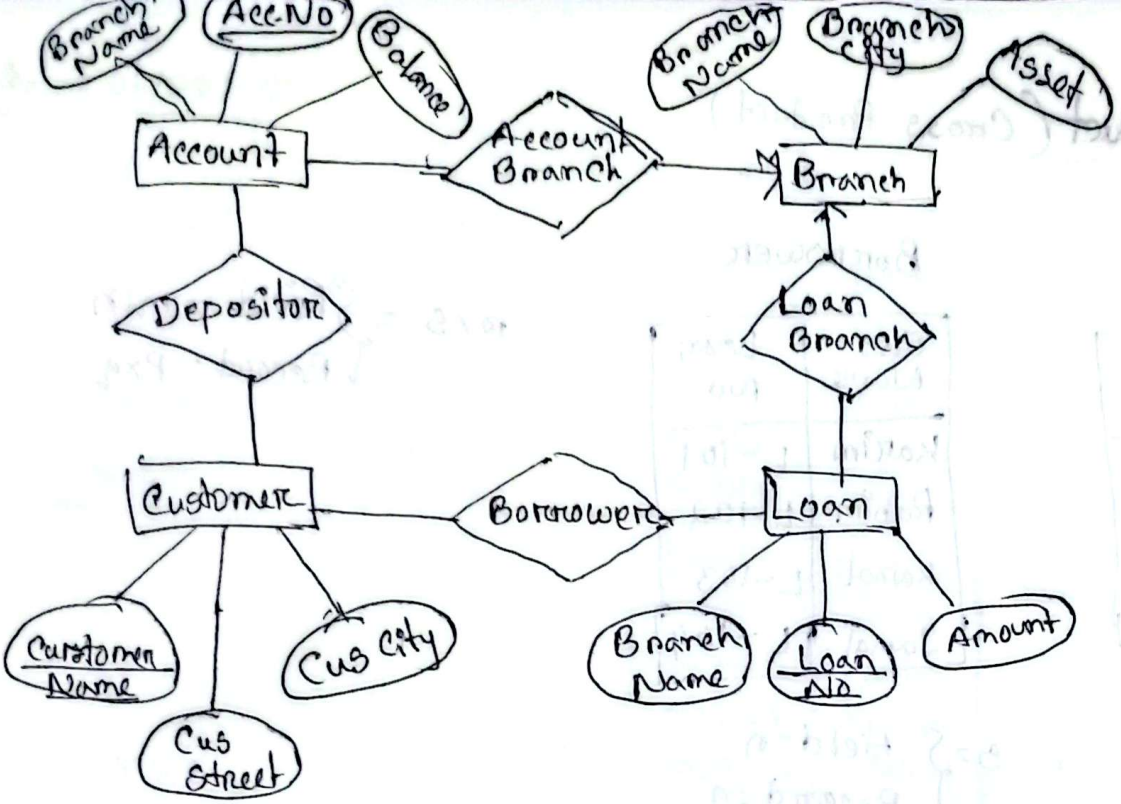


fig: An E-R diagrame for a bank.

Union, Set Difference.

২টি set এর মধ্যে Union, Set Difference করতে.

Conditions:

1. The relations must be same arity that is they must contain same number of feilds.

$$\Pi_{\text{customer Name}}(\text{Depositor}) \cup \Pi_{\text{customer Name}}(\text{Borrower})$$

[Here Depositor and Borrower must have same number of feild].

2. The domain of ith attribute (feild) of relation r is same as the domain of ith attribute of relation s

Cartesian Product (Cross Product)

depositor.

Cus Name	Acc No.
Karim	A-101
Rahim	A-102
Kamal	A-103

Borrower

Cus Name	Loan No
Karim	L-101
Rahim	L-102
Kamal	L-103
Jamal	L-104

$r \times s = \begin{cases} \text{Field} = m+n \\ \text{Record} = p \times q \end{cases}$

$r = \begin{cases} \text{Field} = m \\ \text{Record} = p \end{cases}$ $s = \begin{cases} \text{Field} = n \\ \text{Record} = q \end{cases}$

$\Pi_{\text{customerName}}$ (depositor x borrower)

Natural Join

$\bowtie$ → Natural Join symbol.

Condition:

1. Cartetian product of relations.
2. Perform a selection operation on common fields.
   ↓
   Equality.

$\Pi_{\text{depositor.cusname}}$ ($\sigma_{\text{depositor.cusname} = \text{borrower.cusName}}$ (depositor x borrower))

↓

depositor $\bowtie$ borrower.

example-book — depositor $\bowtie$ borrower $\bowtie$ branch.

Date: 06/05/25

r

Name
Karim
Rahim
Kamal

×

s

Name
Karim
Rahim
Kamal

r × s

r.Name	s.Name
Karim	Karim
Karim	Rahim
Karim	Kamal
Rahim	Karim
Rahim	Rahim
Rahim	Kamal
Kamal	Karim
Kamal	Rahim
Kamal	Kamal

Customer.

Cus Name	City
Karim	Dhaka
Rahim	Rajshahi
Kamal	Khulna
Jamal	Dhaka
Rafiq	Rajshahi
Shafiq	Khulna

r × s

$$\Pi_{r.Name}\left(\sigma_{r.name = s.Name}(r \times s)\right)$$

- Select city from Customer → Dhaka, Rajshahi, Khulna, Dhaka, Rajshahi, Khulna.
- Select distinct city from Customer → Dhaka, Rajshahi, Khulna.

Division Operation:

- All or Every বোঝালে Division operation হবে।

A(x,y) / B(y), denotes or extracts the tuples of A for which x is related to (all) y of B.

sid	cid
s_1	c_1
s_2	c_1
s_1	c_2
s_3	c_2

Enrolled.

cid
c_1
c_2

Course

$$\Pi_{sid}(Enrolled) - \Pi_{sid}\left(\left(\Pi_{sid}(Enrolled) \times \Pi_{cid}(course)\right) - Enrolled\right)$$

Enrolled (sid, cid) / course (cid)

sid
s_1
s_2
s_3

×

cid
c_1
c_2

⇒

$s_1 c_1$ ✗
$s_1 c_2$ ✓
$s_2 c_1$ ✗
$s_2 c_2$
$s_3 c_1$
$s_3 c_2$ ✓

output: s_1

```
Select Account.Customer_Name.
From Account, Depositor.  => Cartesian product
Where Account.CustomerName = Depositor.Customer_Name => Natural join.
```

```
Select Account.customer Name, Balance.
From Account JOIN Depositor.
ON customer_Name
```

```
Select Account.Customer_Name.
From Account, Depositor.  => Cartesian product
Where Account.CustomerName = Depositor.Customer_Name => Natural Join.
```

```
Select Account.Customer Name, Balance.
From Account JOIN Depositor.
ON customer_Name
```

Date: 13/05/25

Extented Relational-Algebra Operations

Outer Join operations

1. Left Outer Join (operator : ⟕)

Depositor

CusName	Acc No
Karim	A-101
Rahim	A-102
Kamal	A-103

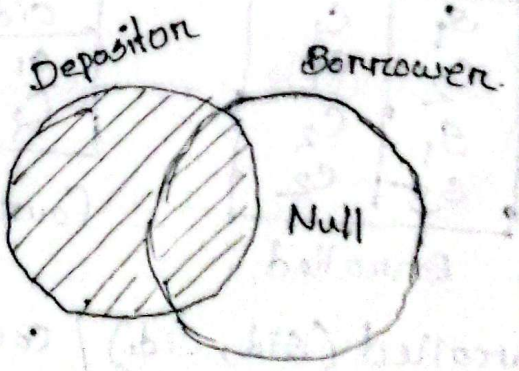
×

Borrower.

Cus Name	LoanNo
Karim	L-101
Rahim	L-102
Jamal	L-103

$\sigma_{Depositor.CusNam = Borrower.CusName}$

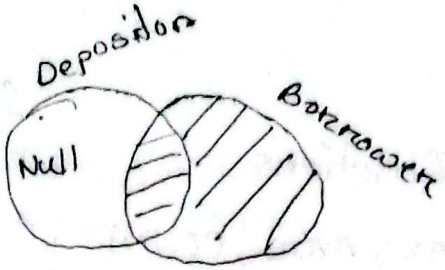
Depositor ⟕ Borrower.



CusName	Acc No	Loan No
Karim	A-101	L-101
Rahim	A-102	L-102
Kamal	A-103	Null

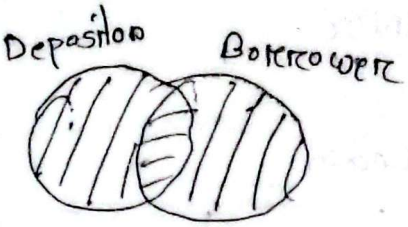
2. Right-Outer Join (operator: ⟖)

Depositor ⟖ Borrower



CusName	AccNo	LoanNO
Karim	A-101	L-101
Rahim	A-102	L-102
Kamal	Null	L-103

3. Full-Outer Join (operator: ⟗)



Depositor ⟗ Borrower

CusName	AccNo	LoanNo
Rahim	A-101	L-101
Karim	A-102	L-102
Kamal	A-103	Null
Jamal	Null	L-103

Database Modification :

1. Deletion operation.

$r \leftarrow r - E$

where E is a relational Algebra expression

Ex:

$loan \leftarrow loan - \sigma_{amount \geq 5000 \text{ AND } amount \leq 1000}(Loan)$.

2. Insertion operation (Insert into values)

$r \leftarrow r \cup E$

$Customer \leftarrow Customer \cup \{("Jobbar", "Jallabad", "Chattagram")\}$

Date: 14/05/25

SQL

* Aggregate functions:
  MIN, MAX, AVG, COUNT
* GROUP BY
  HAVING.

Constraints

* PRIMARY KEY
* FOREIGN KEY
* NOT NULL
* CHECK.

Loan

Branch Name	Loan No	Amount
Rajshahi	L-101	90000
Dhaka	L-102	7000
Khulna	L-103	8000

Borrower

Cus Nam	Loan No.
Karim	L-101
Rahim	L-102
Kamal	L-103

Loan LEFT outer JOIN Borrower ON Loan No.

Loan RIGHT OUTER JOIN Borrower ON Loan No.

Loan INNER JOIN Borrower ON Loan No.

Loan Natural Inner Join Borrower ON Loan No.

Functional Dependency

valid

Sid	S.Name
01	Karim
1	Karim

valid

Sid	SName
1	Karim
2	Karim

valid

Sid	SName
1	Karim
2	Rahim

Invalid

Sid	SName
1	Karim
1	Rahim

(Determinant attribute)

$x \longrightarrow y$ (Dependent attribute)

x determines y
or
x goes to y

where x and y are one or more attributes.

sid → sName

sid	sName
1	karim
1	karim

$t_1.x$, $t_2.x$ → sid; $t_1.y$, $t_2.y$ → sName

If $x \to y$

if $t_1.x = t_2.x$ then
$t_1.y = t_2.y$

$t_1.x \neq t_2.x$.

sid → sName.

sid	sName
1	karim
2	karim

$t_1.x$ → sid; $t_1.y$ → sName

$t_2.x$ → sid; $t_2.y$ → sName

$x \to y$

x	y
1	a
2	b
3	a
4	b
2	a/b

Invalid

$t_1.x$ → 2; $t_1.y$ → b

$t_2.x$ → 2; $t_2.y$ → a/b

valid.

Types of FD :—

1. Trival functional Dependency
2. Nontrival functional Dependency
3. Multivalued functional Dependency
4. Transitive functional Dependency

Trivial FD

If $y \subset x$ then $x \to y$ is called a trivial functional dependency. $x \cap y \neq \phi$

If $y \not\subset x$ then $x \to y$ is called a Non trivial functional dependency

$x \cap y \neq \phi$

SName SRoll → SMarks
(Non trivial)

SName SRoll → SRoll
trivial

SName, SRoll → SName SMarks
(semitrivial)

Date: 15/05/25

Armstrong's Axioms/Inference Rule(FD)

Roll	Name	Marks	Dept
1	a	78	CSE
2	b	60	EEE
3	a	78	CSE
4	c	80	EEE
5	b	60	MSE

Primary Rules:

1. **Reflexivity**: if $x \to y$ then
   $y \subset x$
   * Roll, Name → Name

2. **Transitivity**: if $x \to y$ and
   $y \to z$ then
   $x \to z$

   if Roll → Name and Name → Marks then
   Roll → Marks.

3. **Augmentation**: if $x \to y$ then
   $x.z \to y.z$

   if Roll → Name then.
   Roll. Marks → Name. Mark.

Secondary Rules:

4. **Union**: if $x \to y$ and $x \to z$ then.
   $x \to yz$

   Roll → Name.
   Roll → Marks
   Roll → Name Marks.

5. **Decomposition/Splitting**: if $x \to yz$ then
   $x \to y$ and $x \to z$.

6. **Pseudotransitivity**: if $x \to y$ and $yz \to A$ then
   $xz \to A$

Possible FD:
- $xy \to z$
- $x \to z$
- $y \to z$

4. Composition : if $x \to y$ and $A \to B$ then

$xA \to yB$.

Armstrong's Axioms.

F.D. $\xrightarrow[\text{Axioms}]{\text{Armstrong}}$ Attribute closure

→ candidate keys.
→ Normalization.
→ Data Redundancy overcome.

Date: 19/05/25

Attribute closure :—

$R(A, B, C, D)$

$F.D = \{A \to B, B \to C, C \to D\}$

X
↓
One or more attributes

X^+ = a set of attributes determines by X.

$A \to B$
$A \to A$
$A \to C$
$A \to D$

$A \to B$
$B \to C$.
$A \to C$
$C \to D$

$\therefore A^+ = \{A, B, C, D\}$

Super key : If an attribute closure X^+ determines all the attribute of Relation R. Then X will be called a super key of R.

$B \to B$
$B \to C$
$C \to D$
$B \to D$

$B^+ = \{B, C, D\}$

Similarly,

$C \to C$

$C \to D$

$C^+ = \{C, D\}$

Similarly.

$D \to D$

$D^+ = \{D\}$

2^n

$(AB)^+ = \{A, B, C, D\}$

$AB \to AB$

$AB \to ABC$

$AB \to ABCD$

AB is a super key

$A^+ = \{A, B, C, D\}$

AB is SK but not CK.

$(ABCD)^+ = \{A, B, C, D\}$

$ABCD \to ABCD$

A	AB	BC	ABC
B	AC	CD	ABD
C	AD	BD	ACD
D			

$FD = \{A \to B, B \to C, C \to D, D \to A\}$.

$A \to A$

$A \to B$

$B \to C$

$A \to C$

$C \to D$

$A \to D$

$D \to A$

$A \to A$

A, B, C, D are candidate key.

$FD = \{A \to B, C \to D\}$

$(ACD)^+$

$ACD \to ACD$.

$ACD \to$

AC

A, B, AC

Date: 20/05/25

$R(A, B, C, D, E)$

$F.D = \{A \to B, D \to E\}$

$S_K = ABCDE^+ = \{A, B, C, D, E\}$

$S_K \leftarrow ACDE^+ = \{AC\ D, E, B\}$

$S_K \leftarrow ACD^+ = \{A, C, D, B, E\}$

A	AC
C	AD
D	CD

$A^+ = \{A, B\}$

C_K ACD (একটি key- Candidate key)

Prime Attributes $= \{A, C, D\}$

Non prime Attributes $= \{B, E\}$

$R(A, B, C, D)$

$F.D = \{A \to B, B \to C, C \to A\}$

S_K $ABCD^+ = \{A, B, C, D\}$

$S_K \leftarrow ACD^+ = \{A, C, D, D\}$

$S_K \leftarrow CD^+ = \{C, D, A, B\}$

Proper subset of $CD = \{C\}, \{D\}$

$C^+ = \{C, A, B\}$

$D^+ = \{D\}$

$C_K = CD$

$S_K \leftarrow D \to C_K$

Prime Attributes $= \{C, D, B, A\}$

Non prime $= \{\phi\}$

$B^+ = \{B, C, A\}$

$D^+ = \{D\}$

$S_K \leftarrow AD \to C_K$ (BD)

$A^+ = \{A, B, C\}$

$D^+ = \{D\}$

$\therefore C_K = \{CD, BD, AD\}$

R(A, B, C, D, E, F) Find out the possible candidate key.

$F.D = \{AB \to C,\ C \to DE,\ E \to F,\ D \to A,\ C \to B\}$

Sol^n:

$S_K \leftarrow ABCDEF^+ = \{A, B, C, D, E, F\}$

$S_K \leftarrow ABDEF^+ = \{A, B, D, E, F, C\}$

$S_K \leftarrow ABDE^+ = \{A, B, D, E, F, C\}$

$S_K \leftarrow BDE^+ = \{B, D, E, F, A, C\}$

BDE Not candidate key.

BD is Candidate key.

Prime attribute = {B, D, C, A}

Non prim attribute = {E, F}

$C_K = \{BD, C, AB\}$

B	BD	$B^+ = \{B\}$
D	BE	$D^+ = \{D, A\}$
E	DE	$E^+ = \{E, F\}$

$BD^+ = \{B, D, A, C, E, F\}$

↳ SK

$B^+ = \{B\}$

$D^+ = \{D, A\}$

$C_K = BD$

$S_K \leftarrow CD \quad BC \to SK$

$S_K \leftarrow C^+ = \{C, D, E, F, A, B\} \to SK$

C super key as well as candidate key

C

$AB \to$ SK and CK

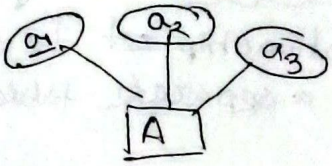
$A^+ = \{A\}$

$B^+ = \{B\}$

Date: 25/05/25

How to convert an ER Diagram into Tables:-

Rule-1: E-R Diagram with simple attribute.

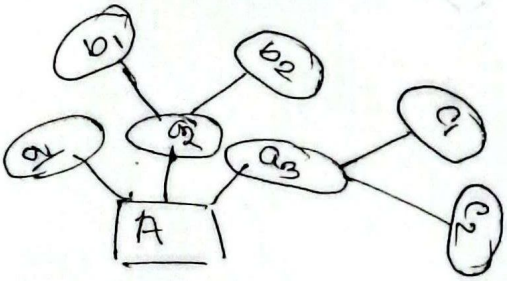


Convert ER diagram into a table as the mentioned entity set with its attributes

Table: A

a_1	a_2	a_3

Rule-2: ER diagram with composite attribute.

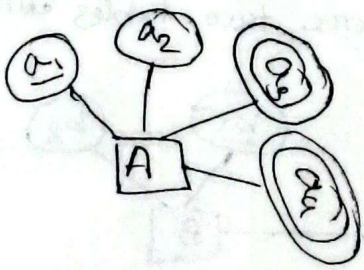


Convert ER diagram into a table by considering the final attributes of the composite attribute not the composite attribute itself.

Table: A

a_1	b_1	b_2	c_1	c_2

Rule-3: E-R diagram with multivalued attributes.



Convert a seperate table with all multi-valued attributes along with the primary key attributes of the entity set.

Table: A_1

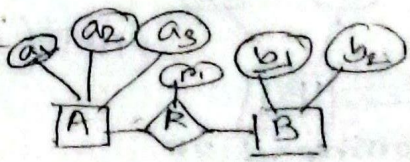
a_1	a_2

Table: A_2

a_1	a_3	a_4

Rule-4: ER Diagrams with relationship set.

Convert entity set and relationship set with separate Tables where the relationship set will contain the primary key attributes of the entity sets along with the descriptive attributes of the relationship set (if any)



R

a_1	b_1	r_1

A

a_1	a_2	a_3

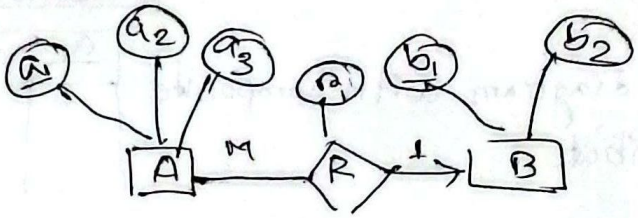
B

b_1	b_2

Rule-5: Binary Relationship with mapping Cardinality.

(a) **For M-M relationship:** Convert the entity sets and relationship set to a separate tables. Here three tables will be converted

(b) **For M-1 relationship or For 1-M Relationship:** Merge the entity set of the many side (M) with the Relationship set to construct a single table and one side (1) with a separate table. Here two two table will be constructed



Schemas:

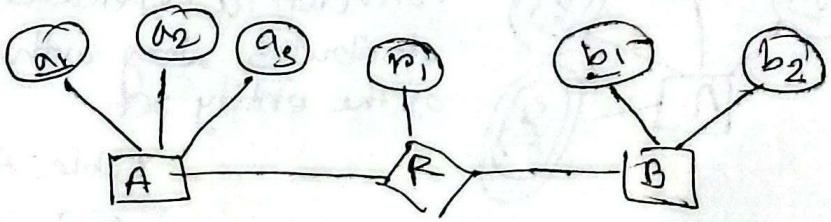
(1-M) $A(\underline{a_1}, a_2, a_3)$

$BR(\underline{b_1}, b_2, r_1, a_1)$ FK.

(M-1) $B(\underline{b_1}, b_2)$

$AR(\underline{a_1}, a_2, a_3, r_1, b_1)$

(c) **For 1-1 relationship set:** Merge any one of entity set with the relationship set to construct a single table and construct a separate table with the other entity set, Here two tables will be constructed.



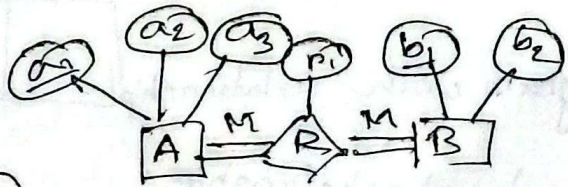
Schemas:

$A(\underline{a_1}, a_2, a_3)$

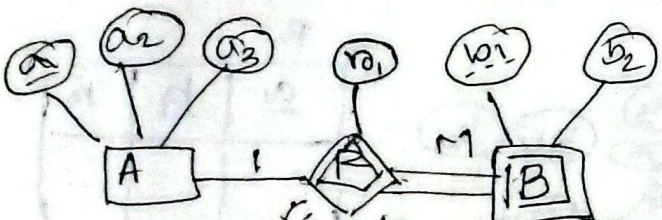
$BR(\underline{b_1}, b_2, a_1, r_1)$

or, $AR(\underline{a_1}, a_2, a_3, b_1, r_1)$

$B(\underline{b_1}, b_2)$



(Total participation)



Weak relationship set or identifying relationship set.

$A(\underline{a_1}, a_2, a_3)$

$BR(\underline{b_1}, b_2, \underline{a_1}, r_1)$

$EARB(\underline{a_1}, a_2, a_3, \underline{b_1}, b_2, r_1)$

Date: 17/06/25

Normalization:-

Anomalies: 3 type

1. Insertion Anomaly
2. Update Anomaly
3. Delete Anomaly

Sid	S Name	Dept	Building	Room No
1	Karim	CSE	4th SC	101
2	Rahim	EEE	1st SC	301
3	Kamal	ACEE	2nd SC	205
4	Jamal	CSE	4th SC	101
5	Rafiq	EEE	1st SC	301
6	Shafiq	CSE	4th SC	101
Null	Null	Civil	1st SC	203

Student | Dept

→ Insertion Anomaly.

1	Karim	CSE	5th SC	401
6	Shafiq	CSE	5th SC	401

→ Update anomaly

3	Kamal	Civil	2nd SC	205

→ Delete Anomaly.

FK

Sid	S Name	Dept
1	Karim	CSE
2	Rahim	EEE
3	Kamal	ACEE
4	Jamal	CSE
5	Rafiq	EEE
6	Shafiq	CSE

PK

Dept	Building	Room No
CSE	4th SC	101
EEE	1st SC	301
ACEE	2nd SC	205

Normalization: Normalization is a process which a larger table is decomposed into several smaller tables to remove or minimize the insertion, update and deletion anomalies.

OLTP, OLAP → Normalized Data

→ DeNormalized Data.

Q. → what is Normalization.

→ why Normalization necessary.

↓

To reduce redundancy

Date: 18/06/25

First Normal form (1NF)

sid	sName	Address	Phone No
1	Karim	Boalia, Rajshahi	P_1, P_2
2	Rahim	Paba, Rajshahi	P_3
3	Kamal	Motihar, Rajshahi	P_4, P_5, P_6
4	Jamal	Puthia, Rajshahi	P_7

Conditions of 1NF :-

1. The attributes must contain atomic value.
2. The values in a column must be in same domain
3. No column Name will be same.
4. No records will be duplicate.
5. No ordering in Rows and columns

Address	
Thana	District
Boalia	Rajshahi
⋮	⋮
Puthia	Rajshahi

sid	Phon1	Phon2	Phon3
1	P_1	P_2	Null
2	P_2	Null	Null
3	P_3	P_5	P_6

Atomic value separation.

Q. Write the conditions of 1NF (for exam.)

Date: 22/06/25

Second Normal Form (2NF)

(*) R(A, B, C, D, E, F)

F.D = {A → B, B → C, C → D, D → E}

$(ABCDEF)^+$ = {A B C D E F}

= {AF}

S_K = {ABCDEF} AF

AF^+ = {AF}

= {A, B, C D E F}

AF = {A}

= {F}

A^+ = {A, B C D E}

F^+ = F

C_K = AF.

Prime attributes = {A, F}

NPA = {B, C, D, E}

[2NF হবে না].

Conditions:

1. If it is in 1NF
2. There is no partial dependency in the relation.

Partial dependency: Proper subset of CK → Nonprime attributes (NPA)

A → B	F → B
A → C	F → C
A → D	F → D
A → E	F → E.

(*) R(A, B, C, D)

F.D = {AB → CD, C → A, D → B}

$(ABCD)^+$ = {A B C D}

= {ABCD}

S_K = AB

= {A} {B}

A^+ = {A}

B^+ = {B}

C_K = ~~AB~~ C B ~~AD~~

S_K = BC

= {D} {C}

C^+ = {C, A}

D^+ = {D B}

C_K = AD.

C_K = BC.

PA = {A, B, C, D}

NPA = {Φ}

This is 2NF

R(A,B,C,D)

FD = {A → B, B → C, C → D}

Prime attribute = {A}

{ABCD}⁺ = {A, B, C, D}

= {A}

NPA = {B, C, D}

S_K = A

C_K = A.

Proper subset of A is ϕ

Single C_K will be 2NF.

(*) R(A, B, C, D) [A.W 2NF হবে না]

F.D = {A → B, B → D}

(ABCD)⁺ = ABCD

= AC

S_K = AC

A^+ = {A}

C^+ = {C}

Attribute closure :-

$X^+ \rightarrow$ set of attributes

R(A, B, C, D, E)

F.D = {A → B, B → C, C → D, D → E}

A → C B → D C → E
A → A B → E C → C
A → D B → B C → CDE
A → E B → BCDE
A → ABCDE

$A^+ = \{A, B, C, D, E\}$

$AD^+ = \{A, D, B, C, E\}$

$B^+ = \{B, C, D, E\}$

$CD^+ = \{C, D, E\}$

$AB^+ \rightarrow \{A, B, C, D, E\} \rightarrow Sk$

(*) R(A, B, C, D, E)

F.D - {A → B, D → E}

$ABCDE^+ = \{A, B, C, D, E\} \rightarrow Sk$

$ABDE^+ = \{A, B, D, E\}$

$ACDE = \{A, C, D, E, B\} \rightarrow Sk$

$ACD^+ = \{A, C, D, B, E\} \rightarrow Sk, Ck$.

A AC {AC, B}
C CD {C, D, E}
D AD

$ABCD E^+ = \{A, B, CD, E\}$

$ACDE^+ = \{A, B, C, D, E\}$

$ACD^+ = \{AB, C, D, E\}$.

prime Attributes = {A, C, D}

(*) F(A, B, C, D)

FD - {A → B, B → C, C → A}

$ABCD^+ = \{A, BC, D\}$

$ACD^+ = \{A, C, D, B\}$

$CD^+ = \{C, D, A, B\} \rightarrow Sk$.

$CD^+ = \{C\}\ \{D\}$

$C^+ = \{C, A, B\}$, $D^+ = \{D\}$

prime attributes = {C, D, B, A}.

CD
↙
BD → Sk, Ck
$B^+ = \{B, C, A\}$
BD
AD → Sk, Sk.
$A^+ = \{A, B, C\}$

R(A,B,C,D)

Prime attributes = {C, D, A, B}

F.D = {AB → CD, D → B, C → A}

ABCD⁺ → {A, B, C, D}

ACD⁺ ⇒ {A, C, D, B}

CD⁺ ≡ {C, D, A, B} → SK, CK.

C⁺ = {C, A}

D⁺ = {D, B}

CD

AB⁺ = {A, B, C, D} → SK

A⁺ = {A}

B⁺ = {B}

⊛ R(A,B,C,D,E,F)

F = {AB → C, C → DE, E → F, D → A, C → B}

C → D, C → E, C → F

AB → F

ABDEF⁺ = {A, B, C, D, E, F}

ABDEF⁺ = {AB, CD, E, F}

ABF⁺ = {A, B, F, C, D, E}

AB⁺ = {A, B, C, D, E, F} → SK, CK.

A⁺ = {A}

B⁺ = {B}

Prime attributes

{A, B, D, C}

AB

DB → SK, CK.

D⁺ = {D, A}

AB

AC → SK, not CK.

C⁺ = {C, D, E, B, A}

⊛

Date: 24/06/25

Third Normalization (3NF)

Sid	District	Country	Code
1	Rajshahi	Bangladesh	
2	"	"	
3			

District, country → code
NPA → NPA.

A → B
A → C
B → C
NPA NPA.

Conditions:

1. If it is in 2NF
2. No transitive dependency in the relation. That is
   Non ~~determinate~~ Prime attrib
   NPA → NPA.

- A relation is in 3NF if and only if on of the following is exist in the non-trivial FD.
  1. L.H.S is SK
  2. R.H.S is PA.

⊛ R(A, B, C, D)

F.D = {A → B, B → C, C → D}
NPA → NPA.

{ABCD}⁺ = {A, B, C, D}
= {A}

A⁺ = {A, B, C, D} → SK.

CK = A

PA = {A}
NPA = {B, C, D}

This not form a 3NF.

⊛ R(A, B, C, D, E, F)

F.D = {AB → CDEF, BD → F}
PA → NPA. NPA → NPA.

ABCDEF⁺ = {ABCDEF}
= {AB}

AB → SK
{A} {B}
A⁺ = {A}, B⁺ = {B}

CK = AB

P.A = {A, B}
NAA = {CDEF}

This not form a 3NF.

(#) R (A, B, C, D, E)

F.D = { A → B, B → C, C → D, D → A } .

ABCDE$^+$ = { A B C D E }

= AE .

S_K = AE .

{A} {E}

A^+ = { A, B, C, D }

E^+ = { E }

AE = C_K

DE

D^+ = { D, A, B, C }

DE

CE

C^+ = { C, D, A, B } .

CE

BE

B^+ = { B, C, D, A }

∴ PA = { A, E, D, C, B }

NPA = Φ

This form a 3NF .

Date: 26/06/25

Boyce-codd Normalization form (BCNF)

When there is multiple overlapping C_K, AB, BC, CD

E.F codd (DBMS এর জনক)

$A \to B$
$PA \to NPA$
$B \to C$

[A BCNF is an enhanced version of 3NF]

Conditions:

1. If it is in 3NF
2. For each non-trivial F.D $x \to y$ if x is a S_K

⊛ $R(A, B, C)$

F.D = { $A \to B$, $B \to C$, $C \to A$ }

$PA = \{A, C, B\}$

$(ABC)^+ = \{A, B, C\}$

This form BCNF

$S_K = \{A\}$
$C_K = \{A\}$

A → B
A → C

$B^+ = \{B, C, A\}$

$C^+ = \{C, A, B\}$

$S_K, C_K = B$

$S_K, C_K = C^+$

⊛ $R(A, B, C, D, E)$

F.D = { $A \to BCDE$, $BC \to ACE$, $D \to E$ }

$(ABCDE)^+ = \{A\ B\ C\ D\ E\}$

$S_K = \{A\}$

$C_K = \{A\}, \{BC\}$

$PA = \{A, B, C\}$

$NPA = \{D, E\}$

This form 2NF

A → BC
AB → AC

$B^+ = \{B\}$

$C^+ = \{C\}$

$A^+ = \{A, B, C, D, E\}$

(

④ R(A, B, C, D, E)

F.D = {AB → CDE, D → A}

(ABCDE)$^+$ = {A B C D E}

S_K = {AB}

{A} {B}

A^+ = {A}

B^+ = {B}

C_K = AB

P.A = {A, B, D}

NPA = {C, D, E}

A B D^+ = {D, A, B}
|
D B

This form a 3NF.

Date: 29/06/25

Fourth Normalization form (4NF):

Multivalued dependency (MVD)

* If a relation R(x, y, z) exist the following dependency:
x →→ y and x →→ z but there is no relation between y and z.

* To be a multivalued dependency a relation must contain at least three colums.

Conditions:-

1. If it is in BCNF
2. No-multi valued dependency of attributes.

Decomposition of Relation

Student:

sid	course	Hobby
1	C#	Reading
1	DBMS	Travelling
1	C#	Travelling
1	DBMS	Reading
2	Compiler	Swimming
2	Network	Gardening
2	Compiler	Gardening
2	Network	Swimming

sid	course
1	C#
1	DBMS
2	Compiler
2	Network

sid	Hobby
1	Reading
1	Travelling
2	Swimming
2	Gardening

Two types of decomposition:—

1. Dependency preserving decomposition
2. Lossless join decomposition. → 1NF, 2NF, 3NF.

Lossless Join:

R(A,B,C)

A	B	C
1	2	1
2	2	2
3	3	2

R(A,B,C)

$R_1(A,B)$ ∪ $R_2(B,C)$

A	B
1	2
2	2
3	3

⋈

B	C
2	1
2	2
3	2

A	B	B	C
1	2	2	1
1	2	2	2
1	2	3	2
2	2	2	1
2	2	3	2
3	3	2	2
3	3	3	2
3	3	3	2

$R_1.B = R_2.B$

A	B	C
1	2	1
1	2	2
2	2	1
2	2	2
3	3	2

Spurious Tuples.

A	B
1	2
2	2
3	3

A	C
1	1
2	2
3	2

A	B	C
1	2	1
1	2	2
1	3	2
2	2	1
2	2	2
3	3	2

A	B	A	C
1	2	1	1
1	2	2	2
1	2	3	2
2	2	1	1
2	2	2	2
2	2	3	2
3	3	1	1
3	3	2	2
3	3	3	2

Lossless Join Decomposition

1. The decomposition will be based on C_K or S_K of the relation R. That is it will be common in both the decomposed relations.
2. For relation R, if $R_1 \cup R_2 = R$.
3. For relation R, if $R_1 \cap R_2 \neq \phi$

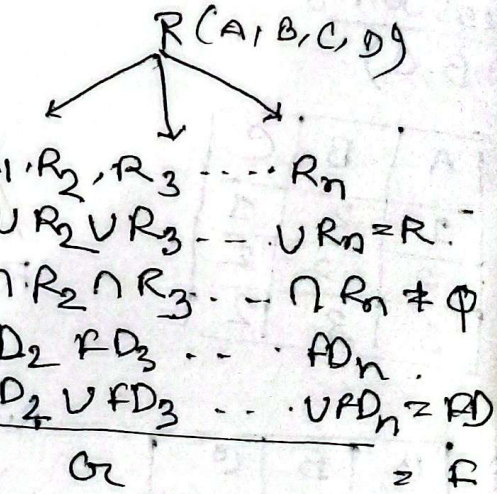
Date: 01/07/25

Dependency Preserving Decomposition:-

A	B	C
1	1	1
2	1	2
3	2	1
4	2	2

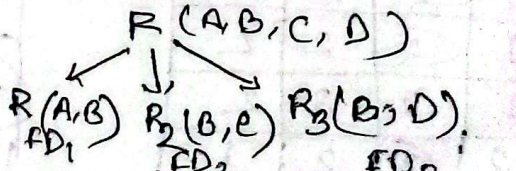
$A \to B$
$A \to C$ $\Big\}$ $A \to BC$
$B \to C$ ✗ $BC \to A$

[Should be non-trivial]



R (A, B, C, D).

F.D = {A → B, B → C, C → D, D → B}



$B^+ = \{B, C, D\}$

$C^+ = \{C, D, B\}$

AB	BC	BD
✗ A → A	✓ B → C	✓ B → D
✗ B → B	✓ C → B	✓ D → B
✓ A → B		
✗ B → A		

$FD_1 = \{A \to B\}, \{B \to C, C \to B\}$

$FD_3 = \{B \to D, D \to B\}$

$FD_1 \cup FD_2 \cup FD_3$

$= \{A \to B, B \to C, C \to B, B \to D, D \to B\} = FD$.

Therefore this decomposition is a dependency preserving decomposition.

(7) R(A,B,C,D)

F.D = {AB → CD, D → A}

AB	BCD
× A → B	B → CDX
× B → A	C → BDX
	D → BC
	BC → D
	BD → C ✓
	CD → B

R(A,B,C,D)

$R_1(AB)$ $R_2(BCD)$

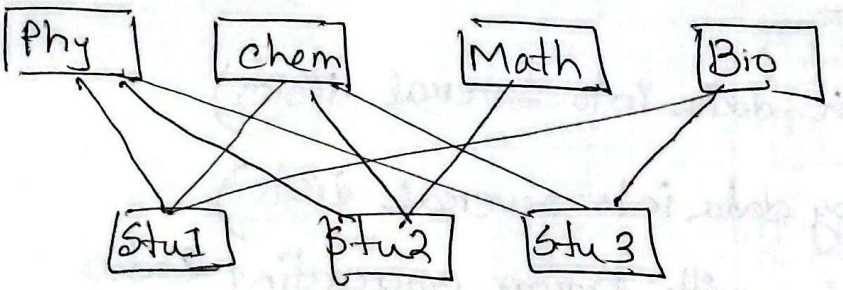
FD_1 ∪ FD_2

∅ ∪ {BD → C}

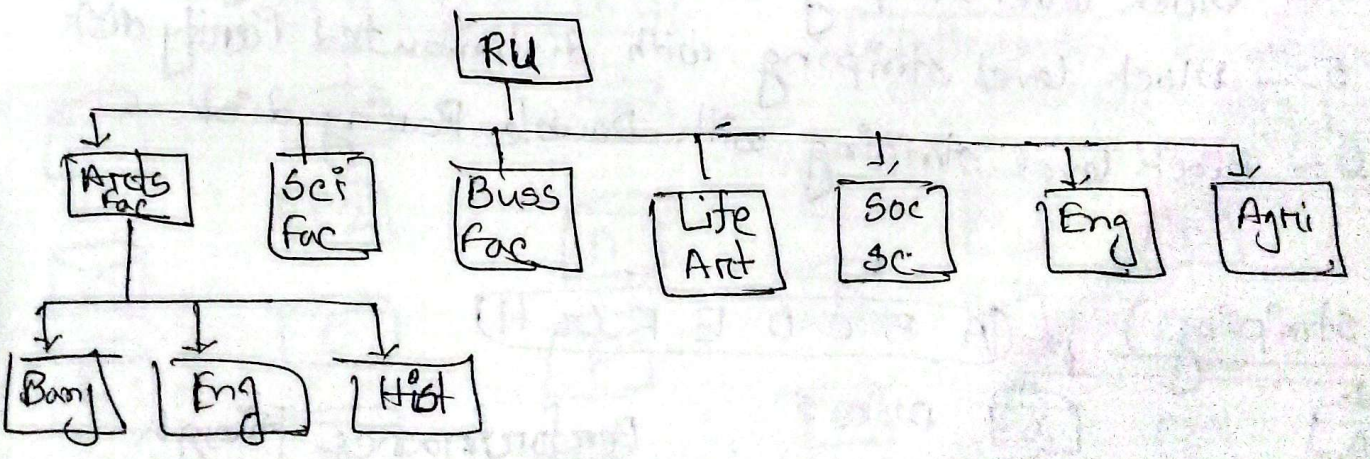
This is not dependency preserving decomposition.

Date: 8/07/25

* Find mapping card



[Network model]



→Meta Data (Data about data)

Date : 13/07/25

File Storage / Desk Storage :

RAID - Redundant Arrays of Independent Disks.
- Redundant Arrays of Inexpensive Disk.

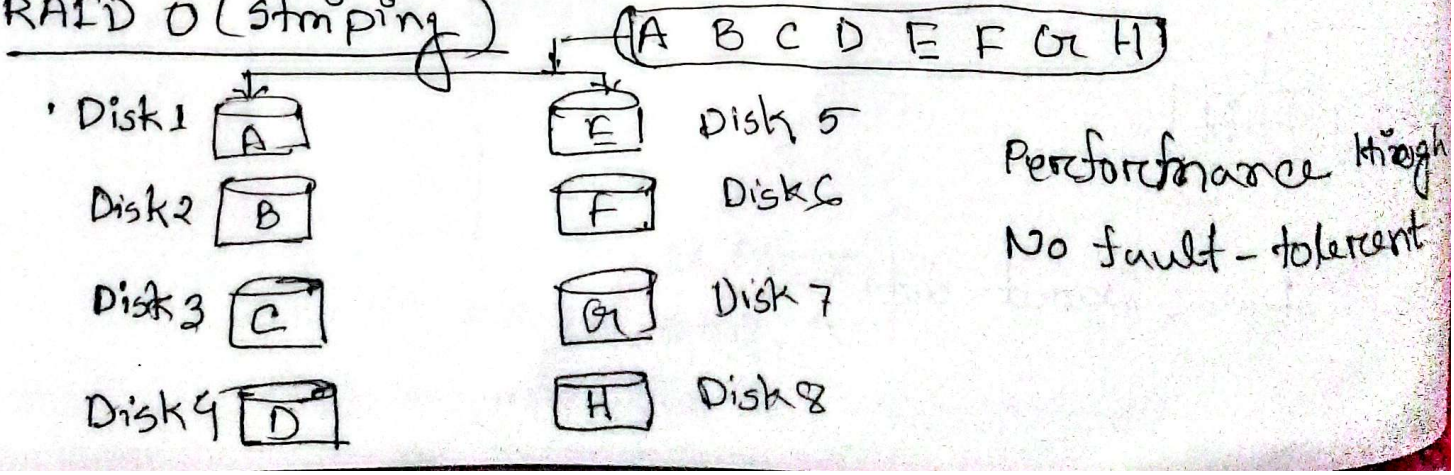
Independent :- All the disk will be handle, read / write, indep

! Performance, Availability.
↓ Faster ↳ Fault-tolerant (24/7)

* Different RAID levels :-

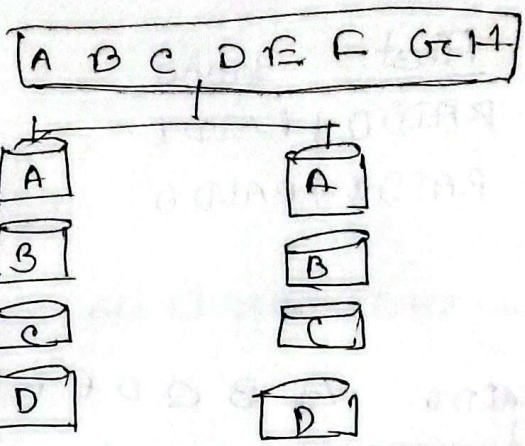
1. RAID 0 — striping (split data into several disks)
2. RAID 1 — Mirroring (Copy data into several disk)
3. RAID 2 — Bit level striping with Error correcting Codes
4. RAID 3 → Byte level striping with dedicated parity disk
5. RAID 4 — Block level striping with dedicated parity disk
6. RAID 5 — Block level striping with distributed Parity d
7. RAID 6 — Block level striping with Double Parity disk

RAID 0 (striping)



Performance High
No fault-tolerent

RAID 1 (Mirroring)

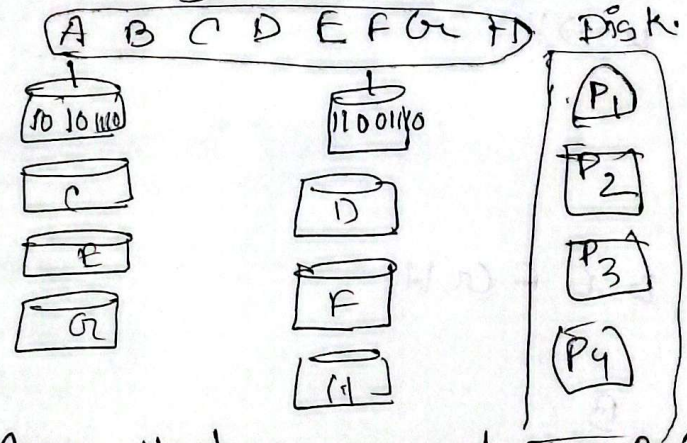


Performance – low.
Security – High.

RAID-2, 3, 4, (outdated)
RAID-5 (Most important)

RAID 2 :- Hamming code
- Not efficient.

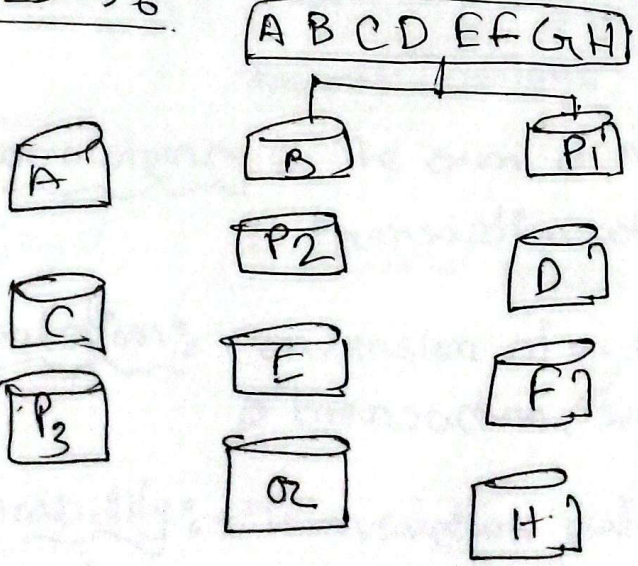
RAID 3 : 1 or 0 (Byte level)



- Single error correction

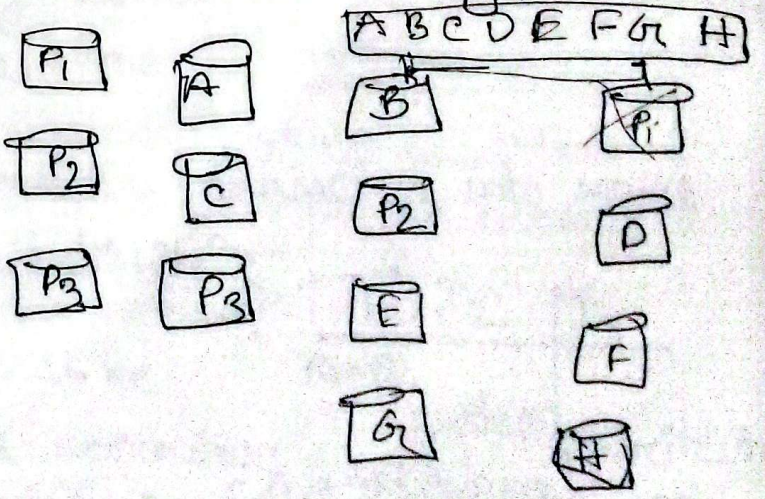
RAID 4 : More than 1 character or Byte.

RAID 5 :



RAID 6 :

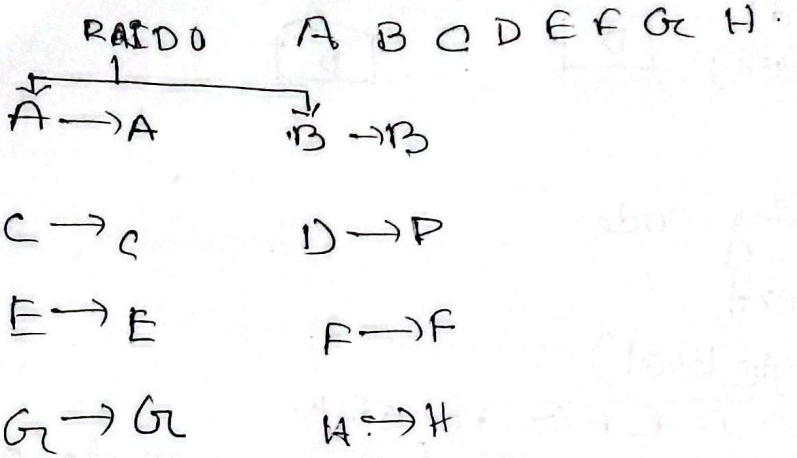
[Difference between RAID 5 is Double Parity]



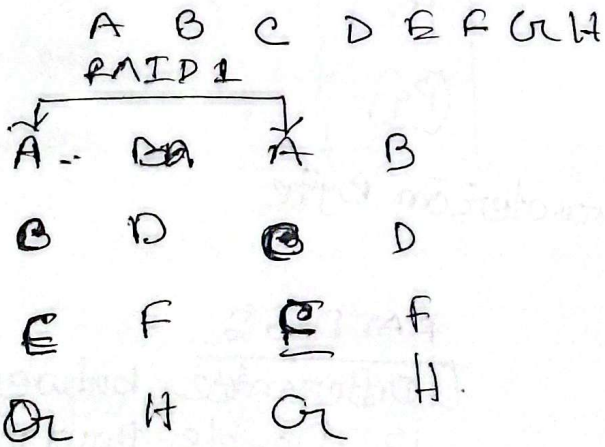
Nested RAID:

	First	Second
1. RAID 01	= RAID 0	+ RAID 1
2. RAID 10	= RAID 1	+ RAID 0

RAID 01 :-



RAID 10:



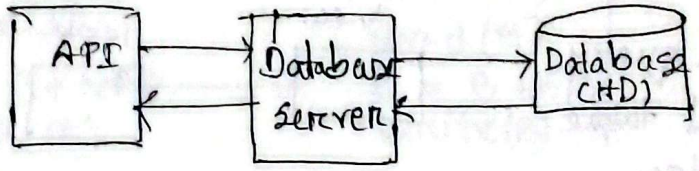
Date: 14/07/25

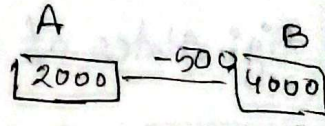
AU

Transaction

Transaction: Transaction is a set of operations used to perform a logical unit of work.

Ex: ATM Transaction, Credit card transaction etc.





Read(A)
A := A-500
Write(A)
Read(B)
B := B+500
Write(B)
Commit.

ACID Properties of Transaction:-

A = Atomicity.
C = Consistency.
I = Isolation.
D = Durability.

Atomicity: Either all or none. (commit or rollback)

Resume / Restart.

Consistency: The sum of money before transaction and after transaction will be same.

Isolation: Conversion of a parallel transaction into serial transaction (serializability)

T_1	T_2	T_3
Read(A)		
	Write(B)	Write(A
A=A-50	B=B+50	Write(B

Durability: - Transaction data will be saved to permanently in the secondary storage (HDD) until a next update is done.

Date: 17/07/25

Transaction states:-

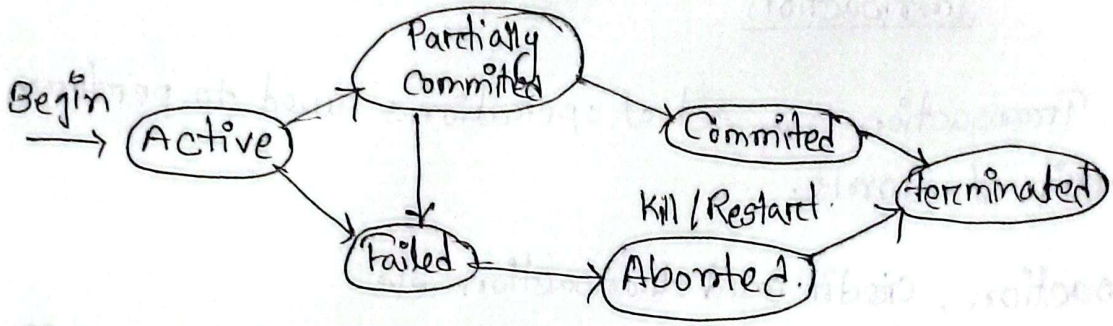
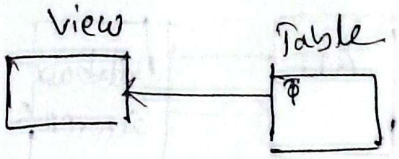


Fig: States of transaction.

- View → stored query → virtual table
  - → updatable view
  - → Non updatable view
  - → meta-



- How to create view?

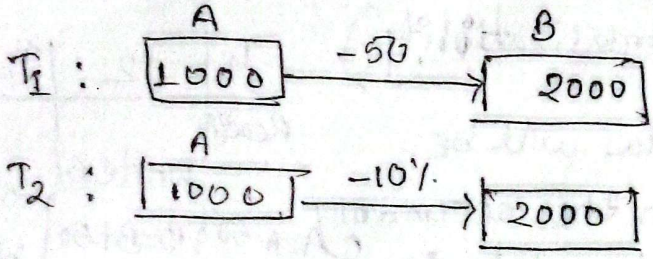
```
Create view V,
As select stuID
From student.
```

- Serializability
  - → conflicting
- Cascading

Schedule: Schedule is a chronological sequence of execution of execution of transaction.

Transaction scheduling:

1. Serial scheduling (Consistency always correct)
2. Parallel or concurrent scheduling.



T_1	T_2
Read (A)	
A = A - 50	
Write (A)	
Read (B)	
B = B + 50	
Write (B)	
	Read (A)
	temp = A * 0.1
	A = A - temp
	Write (A)
	Read (B)
	B = B + temp
	Write (B)

Fig: Serial schedule T_1 followed by T_2

T_1	T_2
	Read (A)
	temp = A * 0.1
	A = A - temp
	Write (A)
	Read (B)
	B = B + temp
	Write (B)
Read (A)	
A = A - 50	
Write (A)	
Read (B)	
B = B + 50	
Write (B)	

Fig: Serial schedule T_2 followed by T_1

T_1	T_2
Read (A)	
A = A - 50	
Write (A)	
	Read (A)
	temp = A * 0.1
	A = A - temp
	Write (A)
Read (B)	
B = B + 50	
Write (B)	
	Read (B)
	B = B + temp
	Write (B)

Fig: Concurrent schedule.
T_1 followed by T_2
(Consistent)

T_1	T_2
Read (A)	
A = A - 50	
	Read (A)
	temp = A * 0.1
	A = A - temp
	Write (A)
	Read (B)
Write (A)	
Read (B)	
B = B + 50	
Write (B)	
	B = B + temp
	Write (B)

Read-Write conflict, gain 50.
(Inconsistent)