C-1, 2 W-1 22/06/25

CSE - 31

Md. Siam Ansary

Room: 9B02

OH: Sun 12:10 - 1:50

Wed 11:20 - 1:50

Quiz: 3rd, 6th, 9th, 12th week

Syll: Whatever is done at that time

Front End: Visual representation

Back End: DB

DBMS: Software package to store and manage data

ACID Property:

i) Atomic: Nothing is lost

ii) Consistency: Value is same

iii) Isolation: Multiple Read/Write is not possible

iv) Durability: Durable (idk mean)

Abstraction Level:

i) Physical: The literal physical record

ii) Logical: Relationship among data (digital data type record)

iii) View: Front end representation

Schema:

i) Physical
ii) Logical
iii) Sub
} Arrangement of levels

Instance: Content at a specific time

i) Single: Single keeps only one instance

ii) Oracle: Keeps multiple instances

Data Independence: Data can be changed without affecting other layers

i) Physical } Exactly what it sounds like
ii) Logical }

DB Users:

i) Application Programmers : Makers
ii) Sophisticated Users : Analyst
iii) Specialized Users : Admin
iv) Naive Users : Everyone Else

Data Definition Language: Language to define the data

- ~~Used~~ At the time of creation
- Primary Key: Unique ~~id~~ data

Data Manipulation Language: Language to manipulate the data

~~i) Pure:~~ - How a data is manipulated (procedural)
- which data is manipulated (declarative)

- i) Pure: Algebra
- ii) Commercial: SQL

DB Architecture:

i) Centralized: Single DB which is connected to every device (small scope)

ii) Client - Server: DB is stored in a server which is accessed from elsewhere

iii) Parallel: Multiple DB is stored in multiple nodes

iv) Distributed: Multiple DB in multiple places but are connected

L-1, W-1 DB Lab 23/06/25

*check file for queries (commands)

varchar(n) → string of n characters

select * from table → select distinct column from table

↳ all values

↳ table name

→ all

→ display

insert into table values (data_1, data_2, ...)

↳ corresponding

→ table name

→ self explanatory

* to enter in specific columns, we need to explicitly specify

select * from table where 'Condition' and/or 'condition 2'

↳ ~~adds~~ filters only those which fit condition

... where Col in (a, b)

↕ same as

... where ~~Col in~~ Col = a or Col = b

... where Col between (a, b) → $a \leq x \leq b$ is printed

... where Col like 'start % end'

↳ matches like stand and end

[check file for the rest]

Super Keys

ID

Email

ID, Email

Email, Name

Candidate Key

ID

Email

PK ~~ID~~ ID

Alternative Key Email

Car

Car	Engine No	Chasis No.	Reg	Color	Model

Candidate Key: Car Engine No.
Chasis No.
Reg

} Multiple attributes can be PK

PK: Reg

C-2, W-1 DB theory 25/06/25

Entity Relationship Diagram

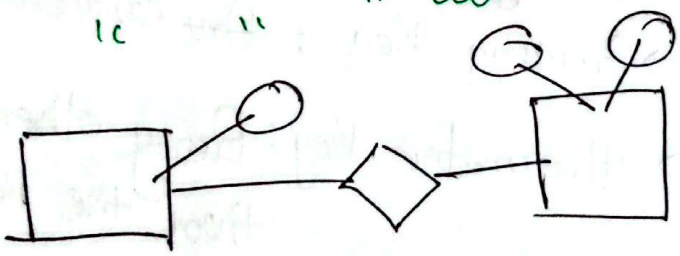
*Requirement Analysis:

Idea → High Level Description → Relationship Schema: DDL DDM → RDBMS → Implementation

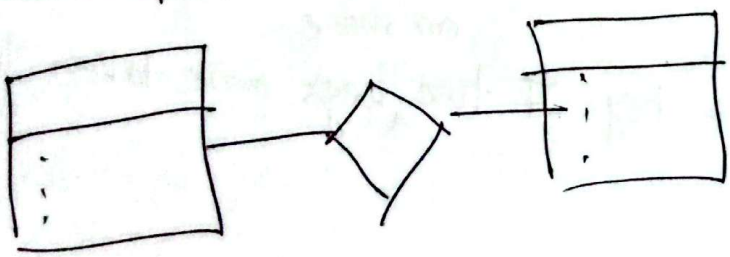
3 parts: i) Entity: Table → nouns →
ii) Attributes: Columns →
iii) Relationship: Connections → verbs →

*Some ~~ent~~ nouns may be the DB
and " " " attributes

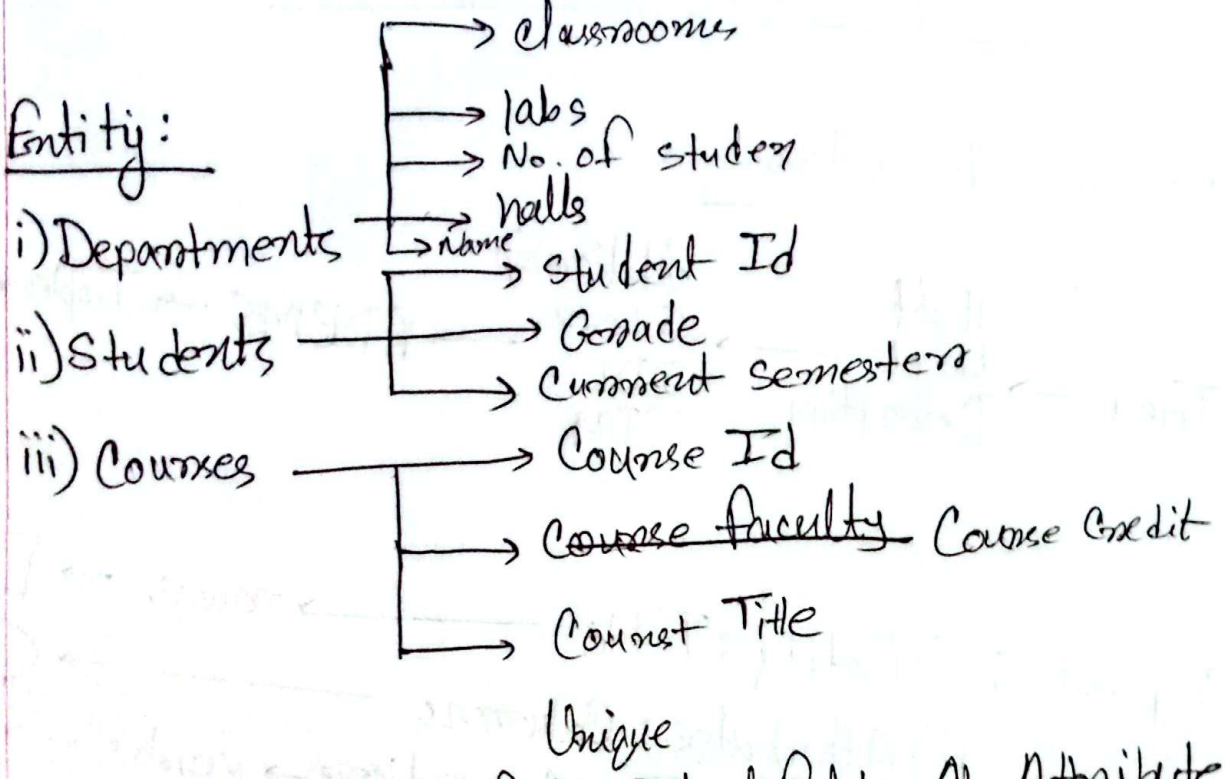
2 reps:



and



Must is . - - receive grades

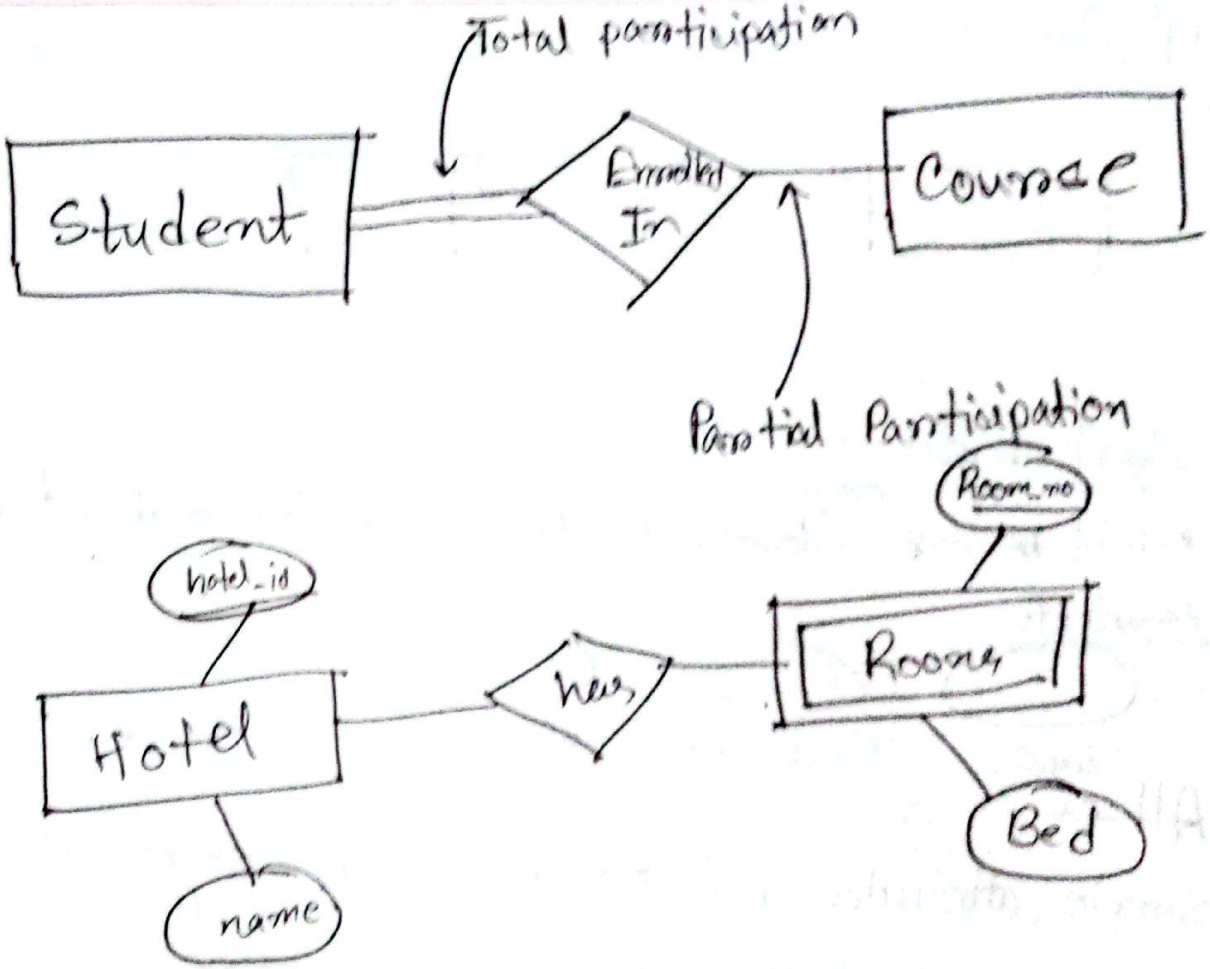


Unique

* Super keys: ~~Identifyable~~ Identifiable ~~Ch~~ Attribute
  - Candidate key: Shortest Super keys
    - Primary key: ~~Uni~~ Candidate key used to identify
    - Alternative key: Every other ~~the~~ candidate key than the Primary key

* Composite key: If two or more keys are taken to be unique

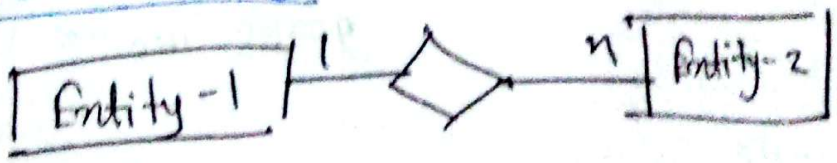
(3,4-91)
C-5, W-2 DB 02/07/25



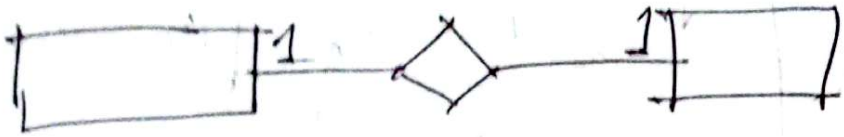
Cardinality Constraints of Relationship

Many to Many:

One to Many



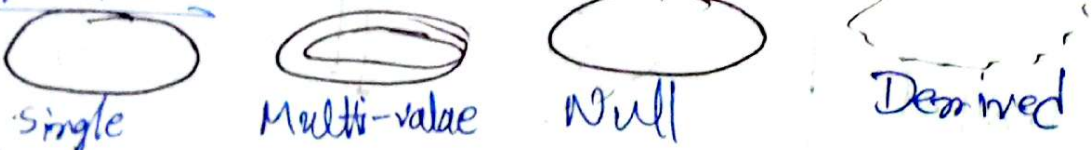
[1] One to One:



Weak Entity:

* Will be one (dominant) to many (subordinates)

Attribute:



~~All si~~

Simple attributes will not always be single-valued as single-valued attributes can give more info

ID: 20220204003

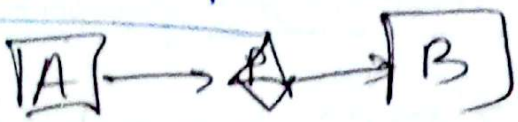
↓

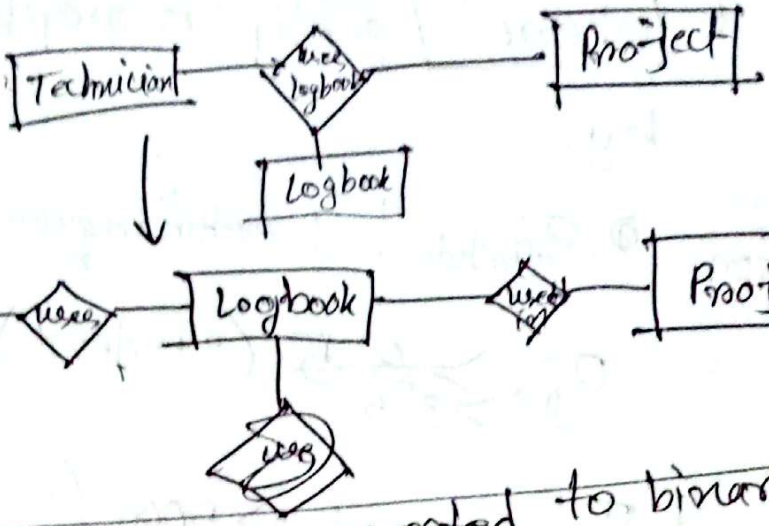
Single but not simple

2022 - 02 - 04 - 003

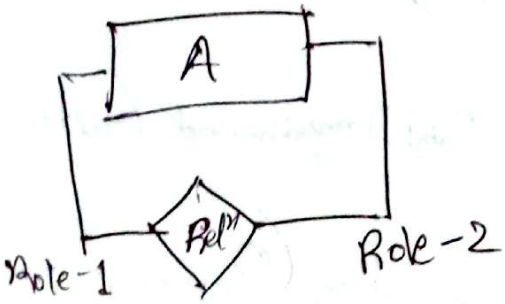
all individually gives more info

Degree of Relationship:

* Binary : 

* Ternary → 

* All ternary can be converted to binary
* Better to make binary relationship

* Unary: 

* N-ary : 3 +

Relational Algebra

select * where query is replaced by,

Projection = $\sigma_{\text{attribute}}$ [ ] condition/requirement (Table)

$\sigma_{gpa \geq 3.5}$ (student)

and $\sigma_{age < 25 \wedge salary > 5000}$ (employee)

$\sigma_{name = \text{"man \%"}}$ (employee)

* for specific attributes,

Projection = Π_{attr} (σ_{attr} [ ] requirement (table))

$\Pi_{name, age, salary}$ ($\sigma_{salary > 5000}$ (staff))

*Rename = $\rho_{\text{old name} \Rightarrow \text{new name}}$ (Table) [attrs]

$\rho_{\text{new name}}$ (Table) [table]

[#] Quiz - 1 : 09/07/25
Syllabus : Up to now

C-6, W-4

13/07/25

Binary Operation Compatibility

* UNION or INTERSECTION can only be done with similar data types and number of columns

* U shows only once distinctly

* CROSS JOIN = Cartesian Product (X)

↳ Cartesian Product repeats column

$P_B \rightarrow R.B(R)$

$P_B \rightarrow S.B(S)$

↳ This creates useless column if similar column

$\Pi_{name}(\sigma_{Borrow.Loan\_No = Loan.Loan\_No \wedge Branch = "Gulshan"}(result\_table))$

SELECT name FROM Borrower CROSS JOIN Loan WHERE Branch = 'Gulshan' ~~AND Bo~~ AND Borrower.Loan_No = Loan.Loan_No.

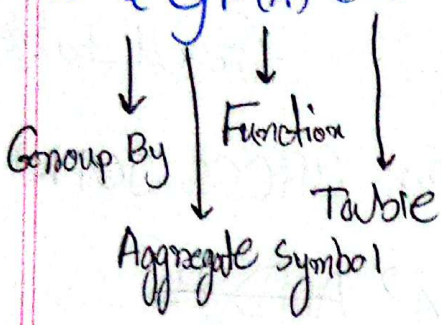
* Other name of inner join is natural join ($\bowtie$)

* $\bowtie$ Inner Join
  $\bowtie$ Left "
  $\bowtie$ Right "
  $\bowtie$ Full "

* Alt way of writing: θ join

$\Pi_{attrs}(Table1 \bowtie Table2)_{condition}$

* $\mathcal{G}_{GF(A)}(R)$



* ~~Max~~ Big queries can be summed up using:
  * Let,

$$\text{Condition} = T_1 \longleftarrow \text{Condition 1}$$

$$\text{---} = T_2 \longleftarrow \text{Condition 2}$$

$$\vdots$$

$$\pi \ldots \left(\sigma_{T_1 \wedge T_2}(T_3)\right)$$

C-8, W-5 23/07/23

Database Security (theoretical)

*Database in different capacities:

i) Application Support: Version control etc

ii) Secured Storage of Sensitive Information

iii) Online Transaction Processing: (OLTP): Transactions that happen online

iv) Data warehousing: Large data stored for future analysis

*Security Layers:

i) Server Level Security: Physical security of servers

ii) Network Level security: Encryptions

iii) Operating System Security: OS of the db servers

*Types of Backup:

i) Full Backups : Complete copy of the data

ii) Differential Backup : ~~Cha~~ Copy of change

iii) ~~Transaction~~ Log Backups : Logs are kept

Quiz - 2 : ~~tue~~ 30/07/25

Syllabus : Upto today

C-9, W-6 27/07/25

Classification of Physical Storage Media

* Priorities:

i) Speed
ii) Cost
iii) Reliability

Storage: i) Volatile: Loses data when power is off
ii) Non-volatile: does not lose data when power is off

* Storage Hierarchy:

i) Primary: Volatile → High speed
ii) Secondary } Non-volatile → Slow speed
iii) Tertiary }

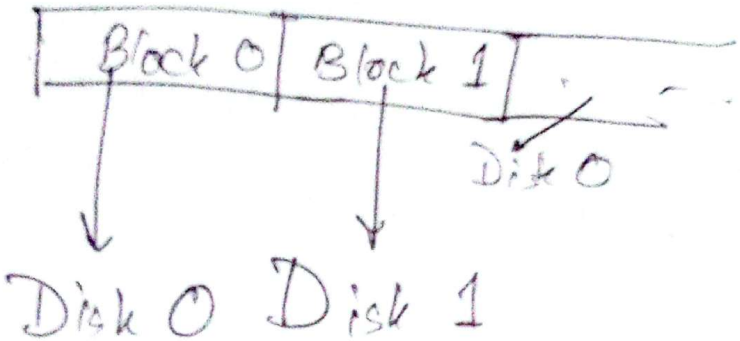
* Mass storage

RAID: Redundant Array of Independent Disks

→ Multiple smaller disks together to make bigger

* RAID level 0: Striping:

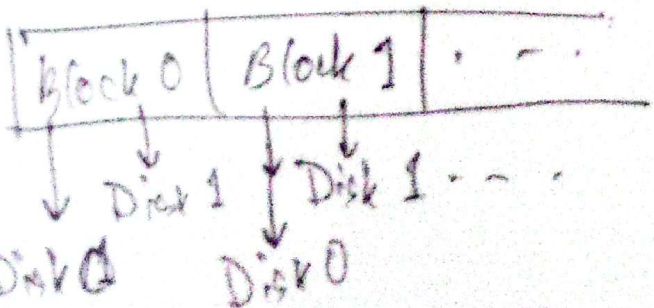
→ Data is divided and stored in different disks



→ No redundancy so no backup

↳ ∴ No full recovery

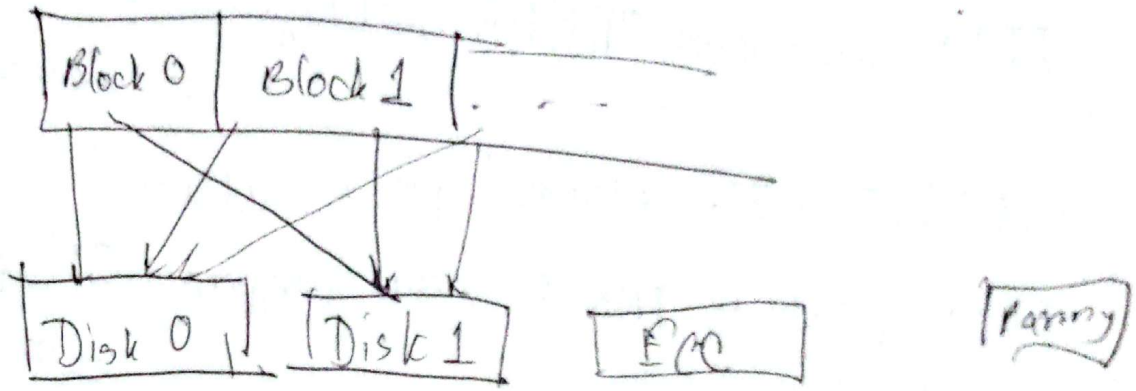
* RAID level 1: Data is stored in one disk but duplicated



→ same read but 1000 write

RAID-1

RAID Level 2: ~~Striping~~ but with error correction



Implementations: i) Software
ii) Hardware

Mean time between failures, MBTF = $\frac{\Sigma(\text{downtime} - \text{uptime})}{\text{No. of failures}}$

C-10, W-7 03/07/25

Transaction Concept

* unit of program execution that possibly updates various data items

* ACID
  - Use to restore/revive everything when ~~op~~ crashes
  - Single transaction at an account at once
  - values must be equal
  - all or nothing
    - may be temporarily inconsistent

* Transaction State:

i) Active
ii) Partially Committed
iii) Failed
iv) Aborted
v) Committed

* Concurrent Executions: Multiple transactions at once

* Schedules: Sequence of instructions

T_2 : A = 100

TEMP = 100 × 0.1 = 10

A = 100 − TEMP = 90

B = 200

B = 200 + 10

B = 210

T_1 : A = 90

A = 90 − 50

A = 40

B = 210

B = 210 + 50

B = 260

To solve this issue, we finish A's work of T_1 and T_2 and then B's work of T_1 and T_2

C-11, W-7 06/07/25

Query Processing

* Parsing and translation
  - → User queries in high level language
  - → Translated expression for physical level
  - → Check syntax and create parse tree
* Evaluation ⟶ Create Evaluation plan
* Optimization ⟶ Choosing the ~~best out~~ one with the least cost

Select name from Emp where salary > 10000

$\sigma_{salary>10000}(\Pi_{name}(Emp))$

$\Pi_{name}(\sigma_{salary>10000}(Emp))$