

OS - CN - DBMS

CN
DBMS
OS

→ Course Expectation

↳ ~~Theory~~ ≡

Computer Network



Network Model

↓ TCP/IP



Computer Networks

Network → It is a group or system of interconnected people or items. Computers connected wide each other with cables or wireless is called computer networks.

Internet → In a nutshell, internet is a network of computer networks. Complex web of interconnected computer networks.

History Of Internet

advanced research
project → agency

1957 → ARPA → 1960's - 70's → 1980's → 1990's

↓
Soviet
↓

Sputnik

↓
US

Communication
system for
ARPA's computer

to talk

↓
1969

ARPANET

TCP/IP

↓
Internet

CERN

↓
hyperlink
based doc

1990

Tim Berners Lee

WWW

mosaic & NetScape

Protocols

Network

Protocols are a set of rules and regulations setup to communicate and share information over a network.

Ex → HTTP, UDP, TCP, SMTP etc

Packets

In order to share data , we can't send big chunk of data over the network . So we divide the data in smaller chunks , these small chunks are called as packets -

Address

Sending messages over the networks require the destination details. This detail uniquely identify the end system is called as address.

Ports

Any machine could be running many network apps.

In order to distinguish these apps for receiving messages we use ports. (port number)

IP-address + Port
Socket

Port helps you get the packets to specific process on the host

↳ Every process has 16 bit port number

$$0 - 2^{16} = 65535$$

range of
port no.

* 0 - 1023 → well known ports

↳ Ex → Port 80 → http
Port 443 → https

* 1024 - 49152 → registered ports

They are used by specific, potentially proprietary
apps / process that are known but not system
defined.

Sql server → 1433

may: → 27017

* 49152 - 65535 → dynamic ports

Access Networks

These are media using which end systems connect to the internet:

Network Interface Adaptor → It enables a computer to attach to a network. As there are different type of networks, it acts as a single suits to connect to any network.

DSL (Digital Subscriber Line)

→ DSL uses the existing telephone ground work lines for internet connection. Generally DSL is provided by same company which supplies telephone service.

↪ ISP (Internet Service Provider) → It is just a company that provides end users internet. Ex → AT&T

Network Protocol Stack

OSI (+layer)

Application
Presentation
Session
Transport
Network
Data Link
Physical

TCP/IP (s layer)

Application
Transport
Network
Data Link
Physical

Application layer → email server, chat server, browser

Presentation → presentation of data, compression
encryption

Session → User session management

Transport \rightarrow Divides big chunk of data coming from above to small chunks. Single these chunks.

Network \rightarrow how route of packets will be done on the network

Data link layer \rightarrow error / flow control, multiplexing & demultiplexing, handles addressing
Physical

Application layer

Roles →

- 1) Writing / providing data off to the network
- 2) Reading the data from user
- 3) Contains applications that helps users to interact on the network
- 4) Error handling & security can also be done.

Where it exists ??

↳ End systems

↳ Instant messaging

↳ www

↳ voip

↳ email

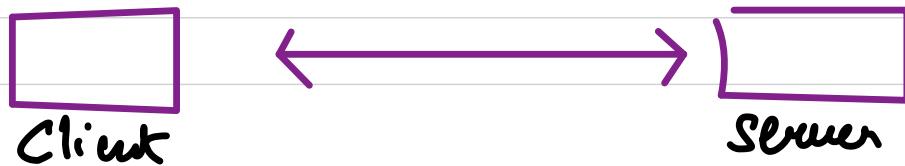
Client - Server Architecture →

↳ It is a 2-level architecture

Client-side

Server side

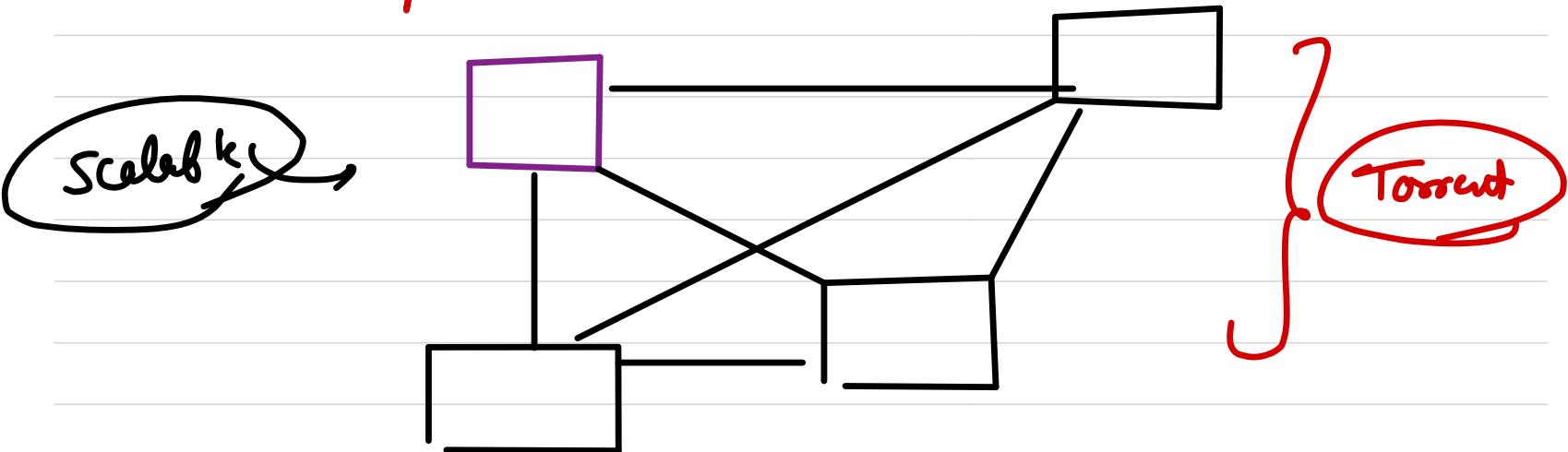
Server → This process controls access to a centralized resource or service such as a website / webapp



Client → frontend where user interacts.

P2P architecture

(Peer to peer)



~~# Hybrid~~ → Combination of client -server &
P2P architecture



It stands for hyper text transfer protocol

↳ Objects → web pages are the main objects that contains other objects.

Some other objects can be mp3 files, jpg, jpeg etc

Every object has a URL

URL Uniform resource locator

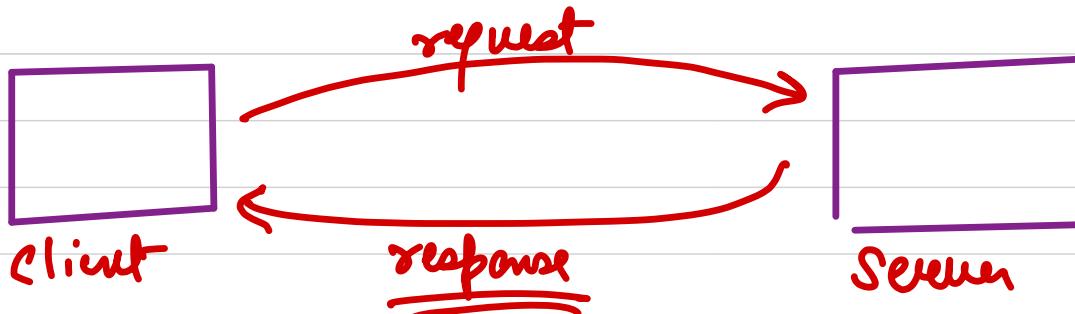
- protocol
- hostname
- location of file / object
- arguments

http://flipkart.com /image/23.jpg ?q=s0

Protocol Hostname Filepath Extra args

HTTP

It defines the whole procedure on how client & server will interact.



The first msg → http request]

The second msg → http response]

http is also
Algorithm :
request-response
protocol

→ http is stateless protocol. (servers donot store
any information about the client.)

So, a lot of application layer protocols depend on lower level protocols of transport layer

→ In transport layer there are of main protocols

① TCP

② UDP

→ HTTP depends on TCP

There are two type of HTTP Connections -

- 1) persistent http
- 2) non-persistent http

HTTP request and response msgs

HTTP req Msg →

any http message are plain ASCII text

- ↳ host
- ↳ method
- ↳ status code
- ↳ Response-poly

⋮

≡

Method

Get

URL

www.booky.com

There are multiple http methods :

- 1) Get → request some data
- 2) Post → put some data on the server
- 3) Put
4) Patch] → update data on server
- 5) Delete → deletes an object at a given url

* user-agent → It specifies the client. Used when server has different web pages that exist for different devices:

- * Accept-any → specifies the preferred lang
- * Connection : close

HTTP Status code:



Convention



SOAP

REST

GRPC

XML

JSON

Protocol buffer

click
()
{
key : val
:
}
y
Seen

deal anything as a resource
→ deal anything as an action

Torrent

DNS

Cooking

SMP

Cookies

- ↳ These are mainly concerned towards privacy.
- ↳ HTTP is a stateless protocol, & a lot of times user session is reqd.

Q: How cookies work ??

cookies are unique identifier strings. These are set by the server through http header.

as soon as a cookies is stored, it is sent along
with subsequent http rep to the same server.
this allows server to know who is contacting it
and hence serve the content accordingly.

* Set-Cookie header →

When a server wants to set a cookie it includes "Set-Cookie : value" in the http response.

this value is stored in the cookie file of browser

Email: SMTP

⇒ for executing the functionality of email, SMTP (Simple Mail Transfer protocol) is used. One more protocol named POP3 is used in combination with SMTP. One is used to send emails that are stored in the user's inbox & other is used to retrieve emails sent to a user.

SMTP also used TCP protocol from transport layer.

Connection for SMTP is setup on port 25.

Mail clients gives the actual UI for end users to send and receive mail. viz gmail, outlook etc

How SMTP works ??

- * When an email is sent , it is sent to the sender's SMTP server using SMTP protocol
(Also the SMTP server is configured in the mail clients)
- * The SMTP server places the email on a message queue.

- * Then SMTP Server initiates a connection with receiver's SMTP Server and conducts an initial SMTP handshake.
- * Then finally it sends the email to recipient's SMTP server.
- * The email is downloaded from receiver's SMTP server & then the client shows the mail.

*Sending
the email*

SMTP → push protocol

*Download
the email*

POP3/IMAP → pull protocols

If recipient SMTP server is offline, the sender SMTP server tries again & again after some delta mins. There is a set threshold after

which it stops sending the email & marks it not delivered.

POP3

POP (Post office protocol)

It downloads emails in 4 phases.

- 1) Connect
- 2) Authorize
- 3) Transaction
- 4) Update

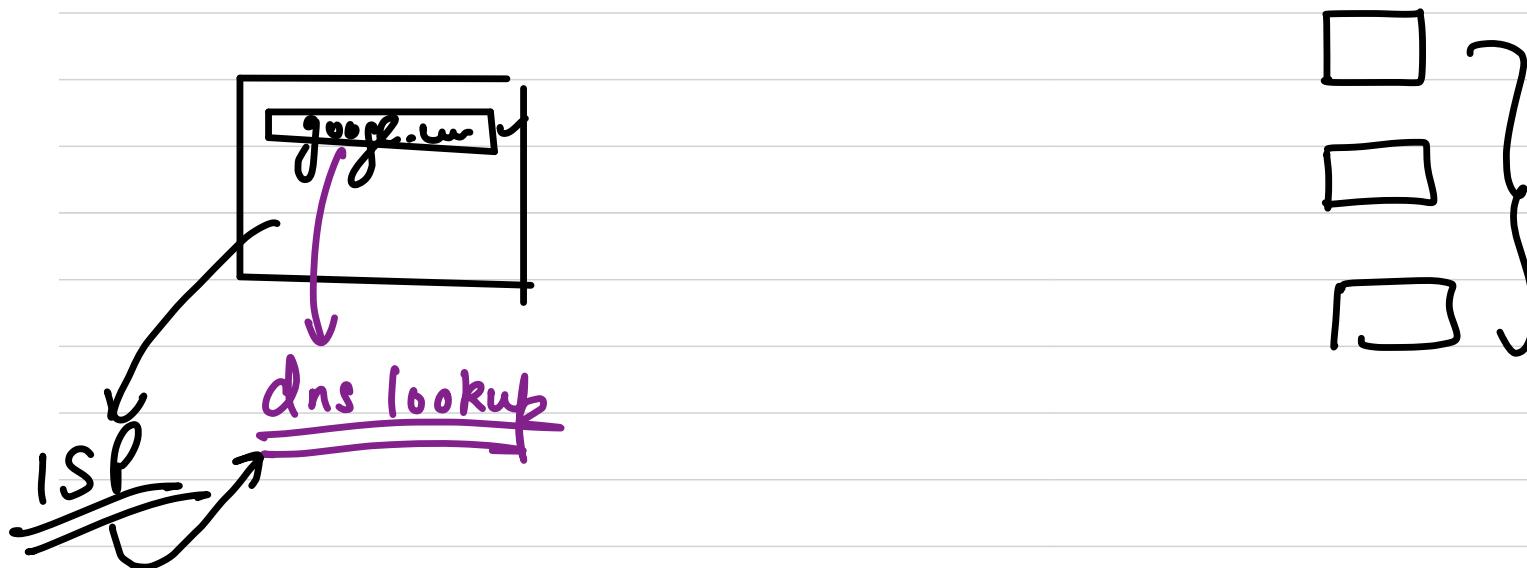
Two modes of POP

- download & keep
- download & delete

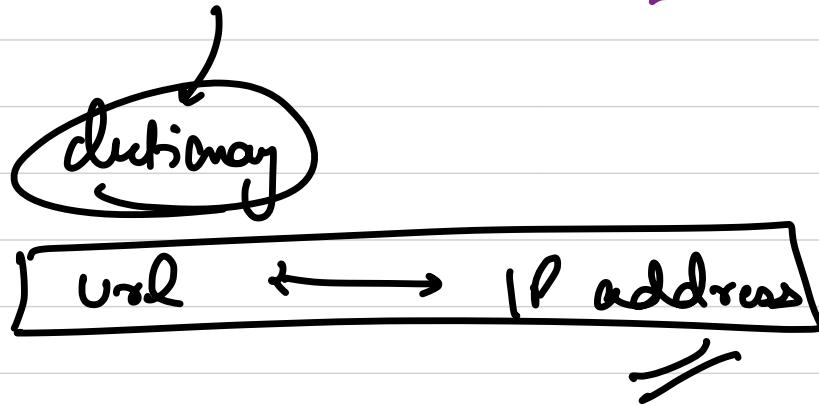
IMAP (Internet message access protocol)

- Emails are kept on the Server & not deleted
- local copies of the emails are cached on each client
- If an email is deleted by user manually then only it gets deleted from server:

What happens when you write www.google.com
on the browser ??



DNS (domain name server)





hosts.txt (URL - IP)

all the hosts were in a file `hosts.txt`, maintained by

a network information center

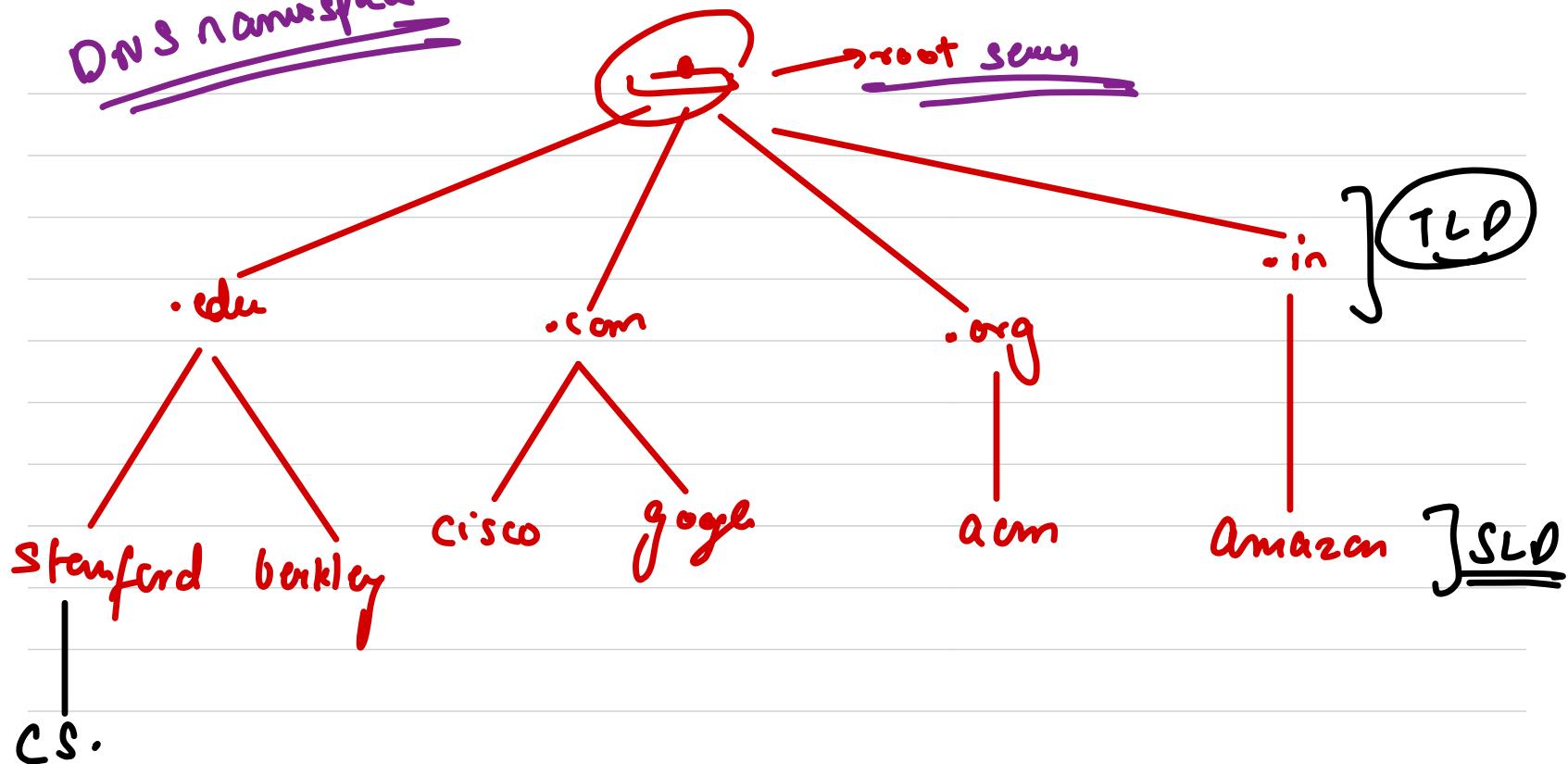
→ using FTP → they first download new copy
of hosts.txt

Domain Name System (dns)

- ↳ maps names to address
- ↳ was able to handle huge records.
- ↳ Robust to failures.

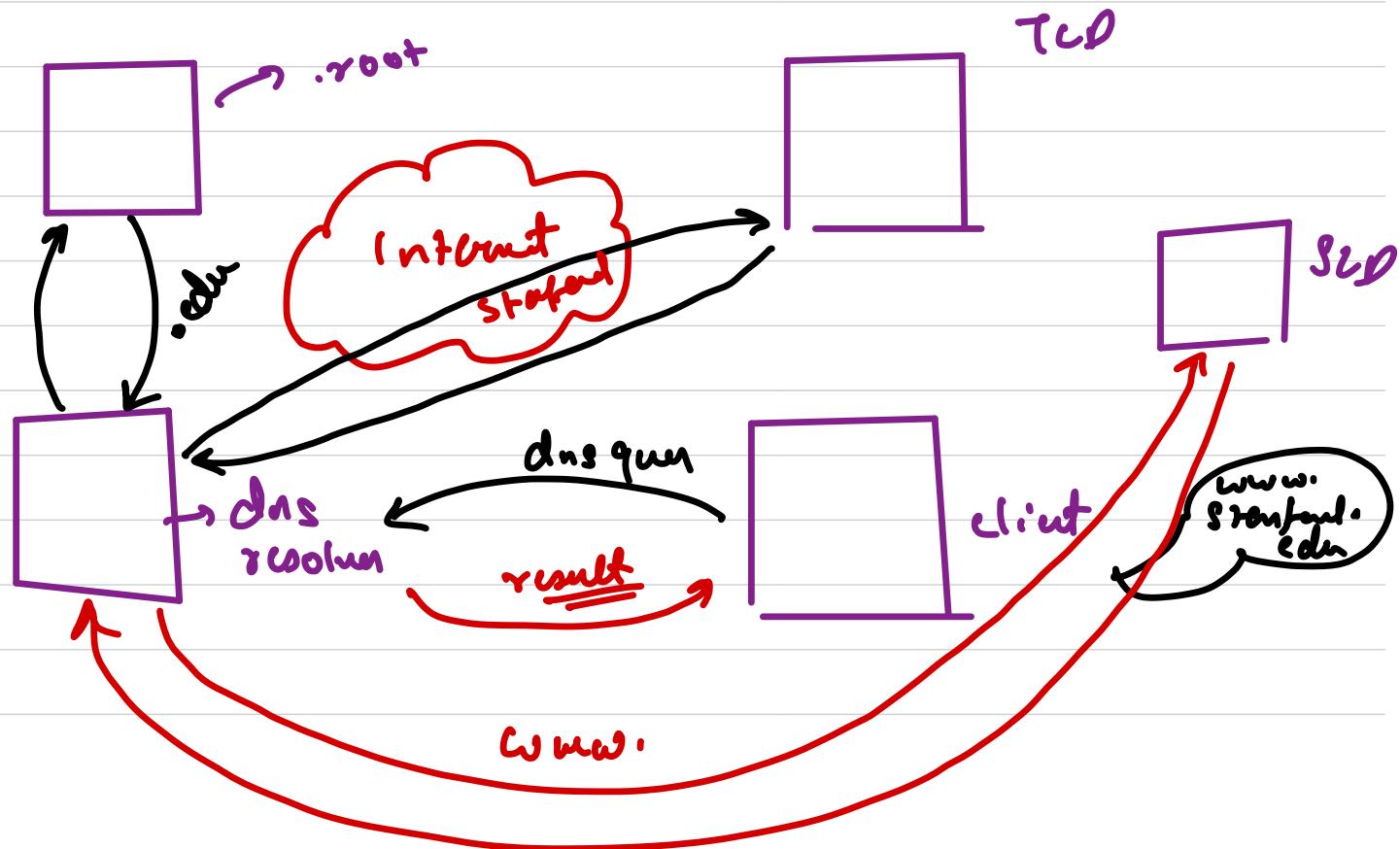
It is a tree Based Structure

DNS namespace



DNS query

read only



TORRENT

→ It is a protocol for P2P file sharing. A bit
Torrent client is an application that uses this
protocol.

- actually it not only follows P2P but instead
follows hybrid architecture
- Allows easy access for sharing large files.

- A bit torrent client requests files from multiple clients in parallel.
- Small chunks of data is called pieces.
- If a client successfully downloads a piece, torrent tells all other clients , that this piece is now also available in this new client & can be downloaded from this client too.

→ The collection of collaborating clients are
called "Swarms"

Torrent file

→ A client joins a swarm by downloading a .torrent file

→ Gives info about the file being shared,

like how big it is, size of its pieces, how

to start interacting with other clients etc

→ Gives info about TRACKER

→ When a client joins the swarm, it req
a list of clients from the tracker, & starts
communication with these clients over
TCP (initially acts as a leecher)

→ When the size of swarm increases, we can also
use tracker less torrent. (distributed ht)

~~Qn~~ what exactly torrent does ??

① It breaks the file by shared into N pieces.

→ for better perf → 256 kb - 1 Mb

→ It uses TCP so the sharing of pieces is

reliable.

② To ensure pieces integrity → torrent attaches

SHA-1 hash to each piece

→ Peers exchange whatever pieces they have.

Peer download the rarest piece first

if any piece is unavailable in all peers,
no one can download it.

This is called as "Rarest first policy"

TFT policy (Tit for tat)

→ You send data to peers, who send you data.

The peers who contribute more can download faster, this creates incentives for seeders.

TRANSPORT LAYER

Transport Layer, takes messages from network
to applications & vice - a - versa.

It segments the data in small manageable pieces
called as "Segments" or "Datagrams".
It allows logical app-app delivery,

It multiplies & demultiplies data.

Where it exists

↳ Transport layer also exists on end systems.

In transport layer there are 2 main protocols

- 1) TCP → Transfer Control protocol
- a) UDP → User datagram protocol

TCP

→ While using TCP small pieces of data are called segments

Ensures reliable and in-order delivery of segments

Delets modifications on the packets during delivery & corrects them.

Slower than UDP

Ex → HTTP, Email, FTP

UDP

While using UDP small pieces of data are called data grams

does not ensure reliable delivery of data grams.

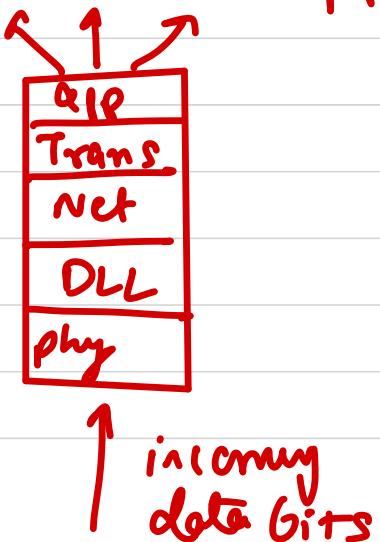
It also detects modifications but doesn't correct them.

Faster than TCP

Ex → VoIP, live streaming

Multiplexing And Demultiplexing

* Demultiplexing → It is the process of delivering the correct packets to correct apps from one stream.



* Multiplexing → It allows data to be sent to more than one dest. host via a single medium.

How all of this is managed ??

↳ Port no.:s help in multiplexing & demultiplexing.

Congestion

When more packets than the network bandwidth is sent through it, then the performance drops & loss of data occurs. This is called as Congestion

Impairments in Network Layer:

↳ Segments can be corrupted / lost / damaged / reordered / duplicated etc

Check sums → It is an error-detection mechanism

We can keep an arithmetic sum of all bytes of segment.

After sending we can match the checksum on receiver's end with the one on sender's end.

Retransmission timer →



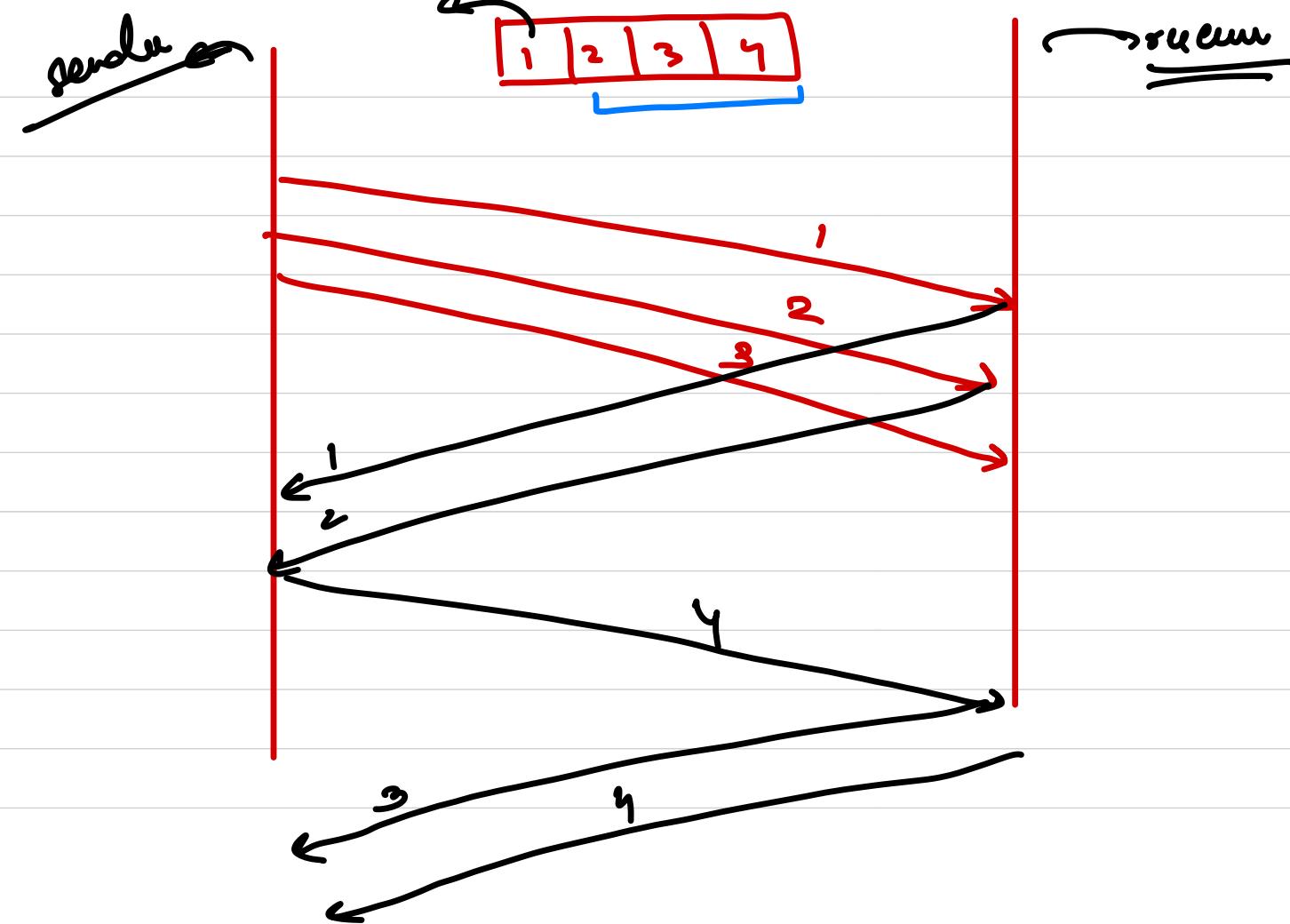
Sequence Number → unique id no for any symbol

Pipelining → apps can generate data at a multi
giga rate than expected by network.

for TCP → reliable message comm.

* Sliding Window

→ It is the set of consecutive no.'s that the sender can use when transmitting segment without being forced to wait for ack.



Detection & re-transmission of packets cannot be handled just by steady random protocol.

Go-Back N

It is defined at two spots in a network

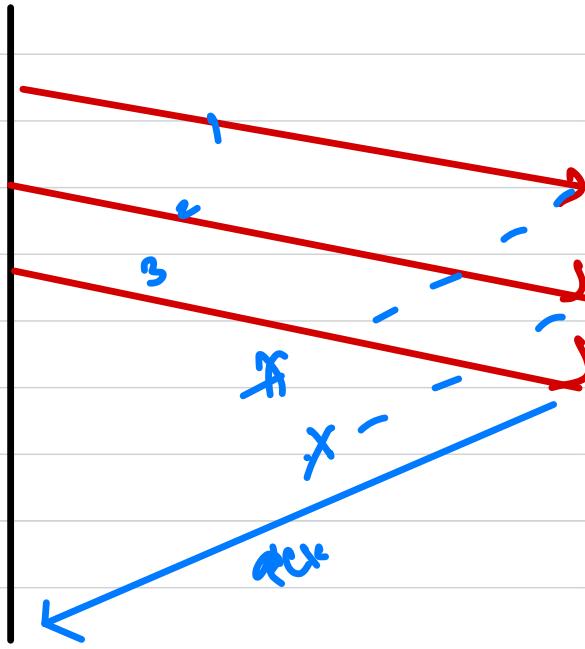
- 1) Sender
- 2) Receiver

→ Receiver →

- ↳ It only accepts those packets that come in seq
- ↳ Any out of sequence packet is discarded.
- ↳ When it receives a data segment, it sends an ACK containing Seq no of last in-seq segment.

Gender a

success



→ Sender → Segments are sent based on sliding window

Sender must wait for ACK if the buffer is full

When the Sender receives an ACK it removes all

previous acknowledged segments.

Selection Repeat

- This also uses sliding window like go-back-n.
- Window size should be less than or equal to half of the Cq no.

↓
x

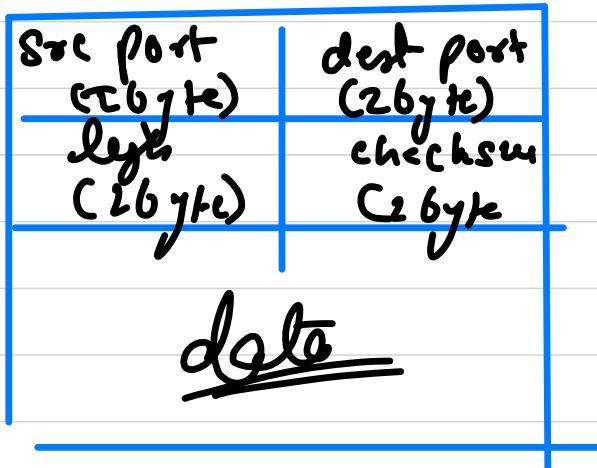
Sender retransmits unack packets after a timeout.

Receiver ack all correct packets.

UDP (user data gram protocol)

→ Non reliable

header of
data gram {
 2
 8 bytes

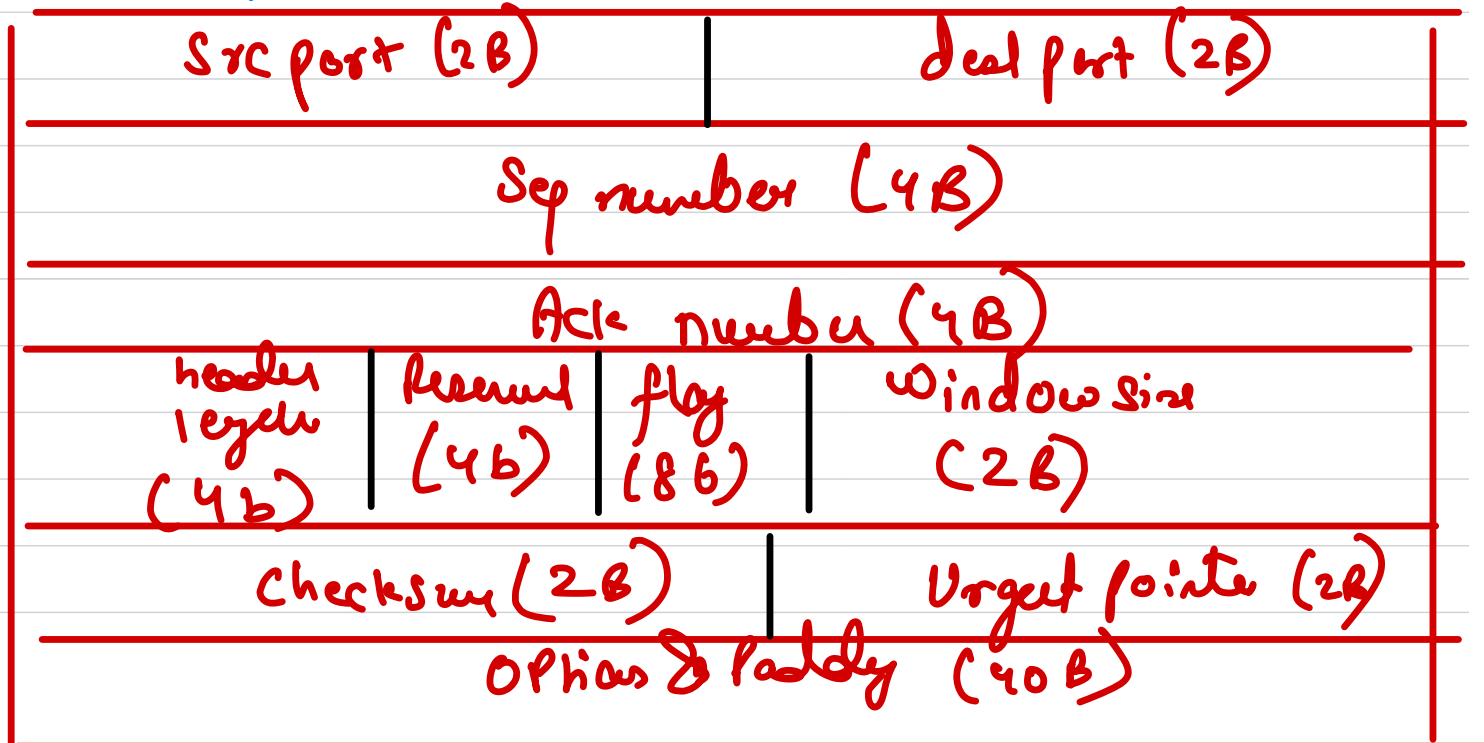


} UDP datagram

TCP (Transmission control protocol)

→ > ~~header~~

→ ~~header of TCP~~





ACK → acknowledgement

RST → reset

SYN → Synchronisation

FIN → finish

wget flag

push

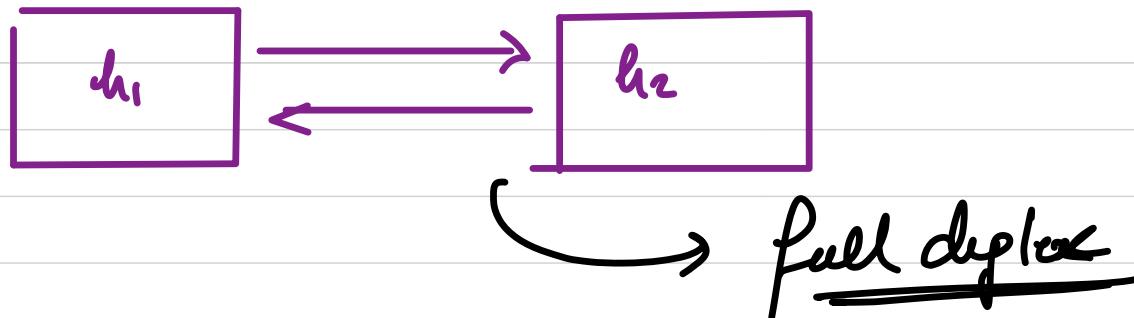
TCP

Reliable

Connection Oriented

Error Control

flow control



How to setup a TCP connection

Three way handshake →

1) Initiating a Connection →

SYN flag set

Sq no → random value

}

→ send by Sender

2) Responding to initial connection msg

SYN flag set

ACK flag is set

ACK no. is set



SYN+ACK

Segment

3) Connection established ACK \Rightarrow Sender sends to receiver.

\downarrow
~~SYN~~ is not set

ACK is set

Network Layer

Main objective is to allow end systems to exchange info through intermediate systems called as Router. The unit of info in network layer is called packet.

Services of Network Layer →

1) Unreliable Connection less Services →

2) Connection oriented and reliable

The main responsibility of Network layer is
routing of packets -

Static Routing

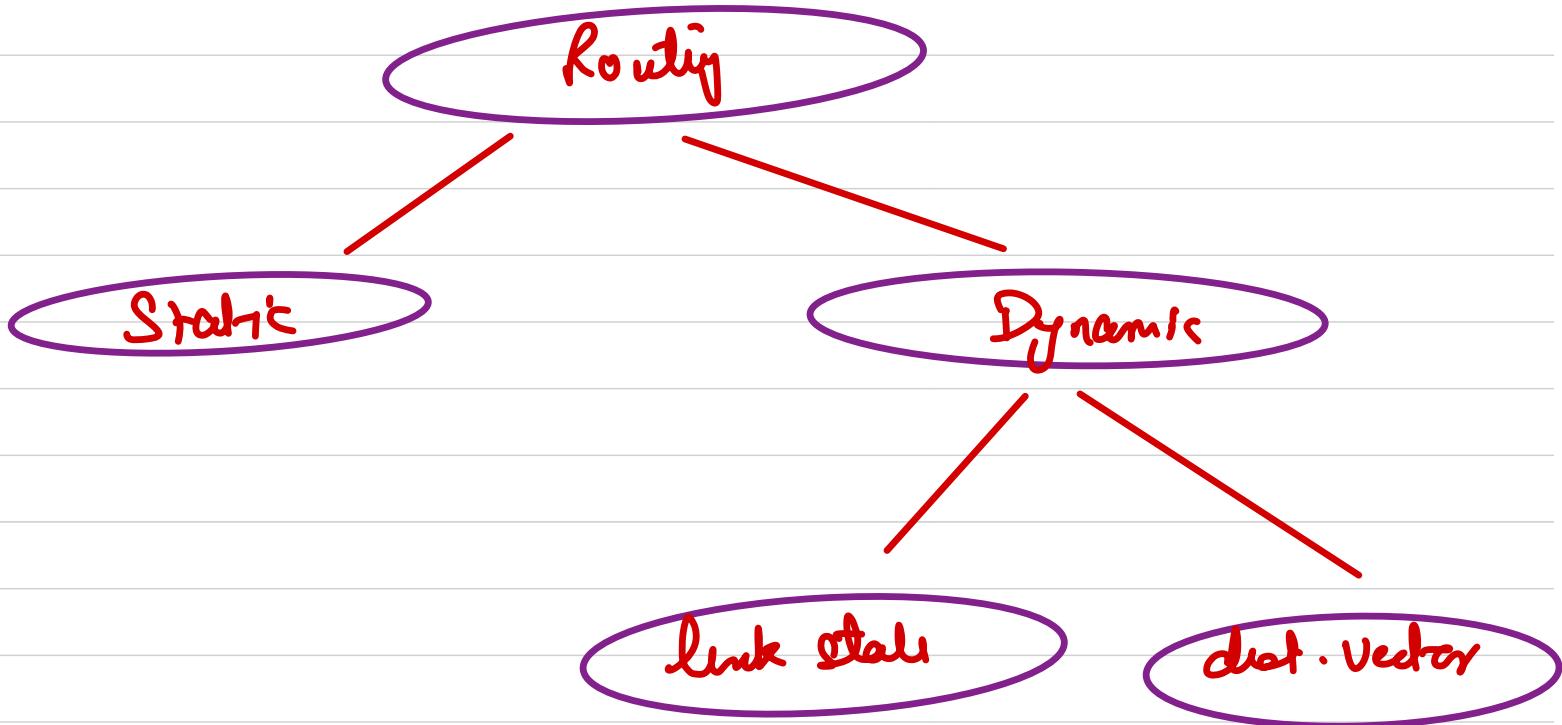
Dynamic Routing

Static Routing → Here we have manually computed routes which manually managed in routing table.
It is good for small network → + does not adapt automatically to changes in network.

Dynamic routing \Rightarrow They have adoptable routing mechanism & routing tables -

There are 2 types of dynamic routing -

- ① Link State
- ② Distance Vector



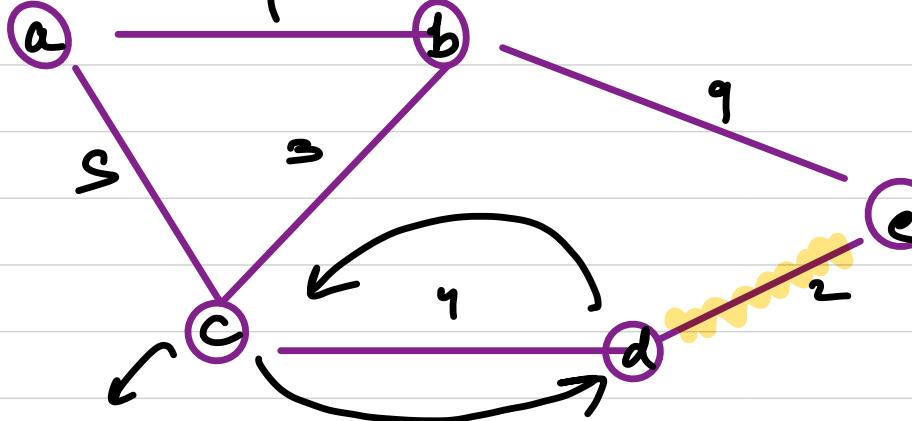
Distance Vector Routing

* Initial State → Each router or node, maintains a routing table that initially contains estimated cost of the ~~Neighbours~~ Neighbours

Bellman Ford algo

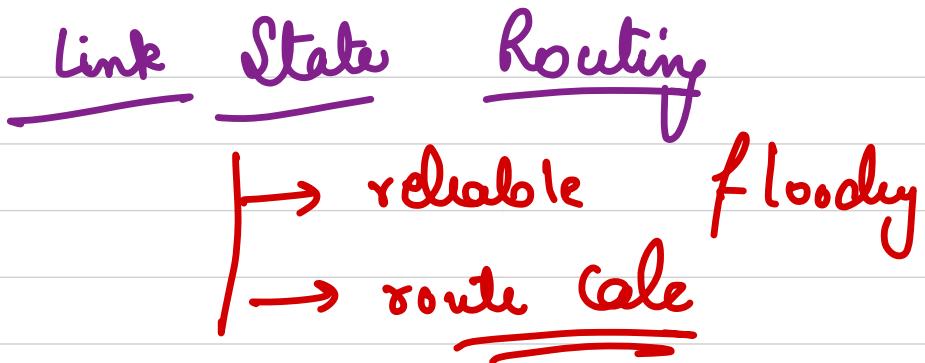


dest	cost	next
c	5	c
b	1	b



dest	cost	next node
a	4	b
b	3	d
d	4	e
e	12	
		b

Count to infinity problem

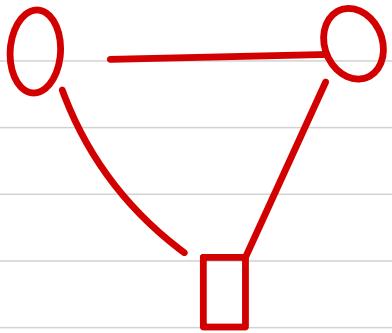


Reliable flooding → Whenever any router boots, it

sends a hello msg to the neighbors.

Every node after reliable flooding sends LSP to all routers in network

link state packet



It uses flooding to transmit it everywhere

Phase 2 → route calc → Dijkstra's

Data Link Layer

Responsibilities of DLL is →

To receive packets from network layer & dealy with providing hop to hop communication between entities that are connected to physical layer.

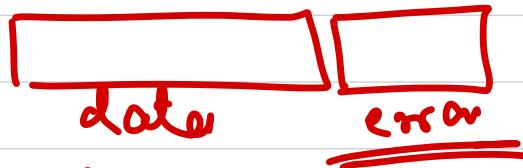
Three types of DLLs

- 1) Only 2 points of communication are involved - P2P DLL
- 2) Broadcast Multi access → used in LAN.
- 3) Non-Broadcast Multicast.

⇒ Error detection → Recovery

Sender ← → Receiver

- 1) the sender adds some redundant info (say n bits) as an error detection code.



parity

Even
Odd

HTTPS

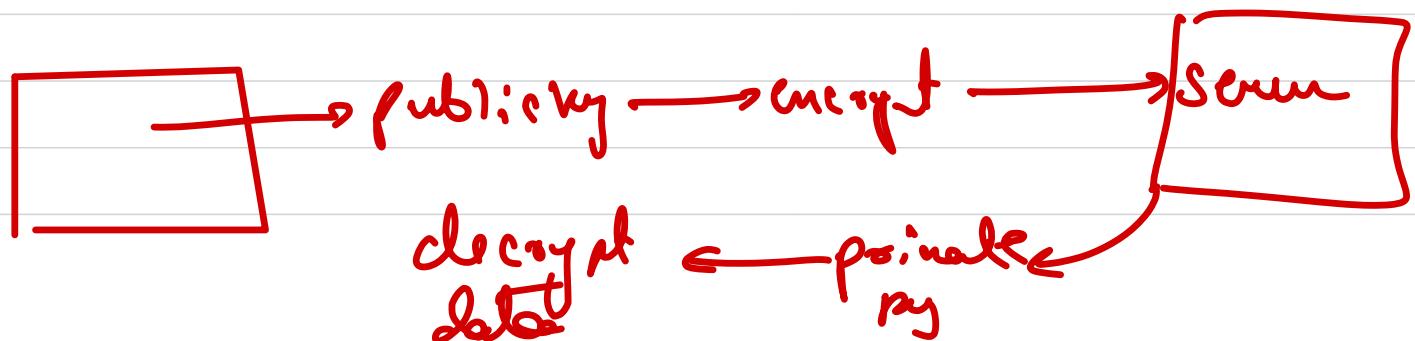
HyperText Transfer protocol Secure.

Secure version of http. It encrypts data to increase security of data transfer.

https uses a protocol called as TLS (Transport layer security) was formerly known as SSL (Secure Sockets Layer).

This protocol uses asymmetric public key infrastructure for encryption. This security system uses two different keys to encrypt communication between 2 parties.

i) Public key → this key is available to every one who wants to interact with web site users in secure way. Info that is encrypted using public key can only get decrypted by private key.



2) private key → it is controlled by website owner and is kept private from the network. this key lies on the servers to decrypt info encrypted by public key.

How https is diff from HTTP

TLS/SSL Encryption over HTTP.

SSL certificate

user

public
key

SSL handshake

SSL Certificate

It contains public key of the ~~2~~ key info.

- ↳ domain name
- ↳ org / person to which certi was issued
- ↳ certificate authority
- ↳ digital signatures
- ↳ Expiry of certi
- ↳ ~~public key~~

TLS & SSL handshake

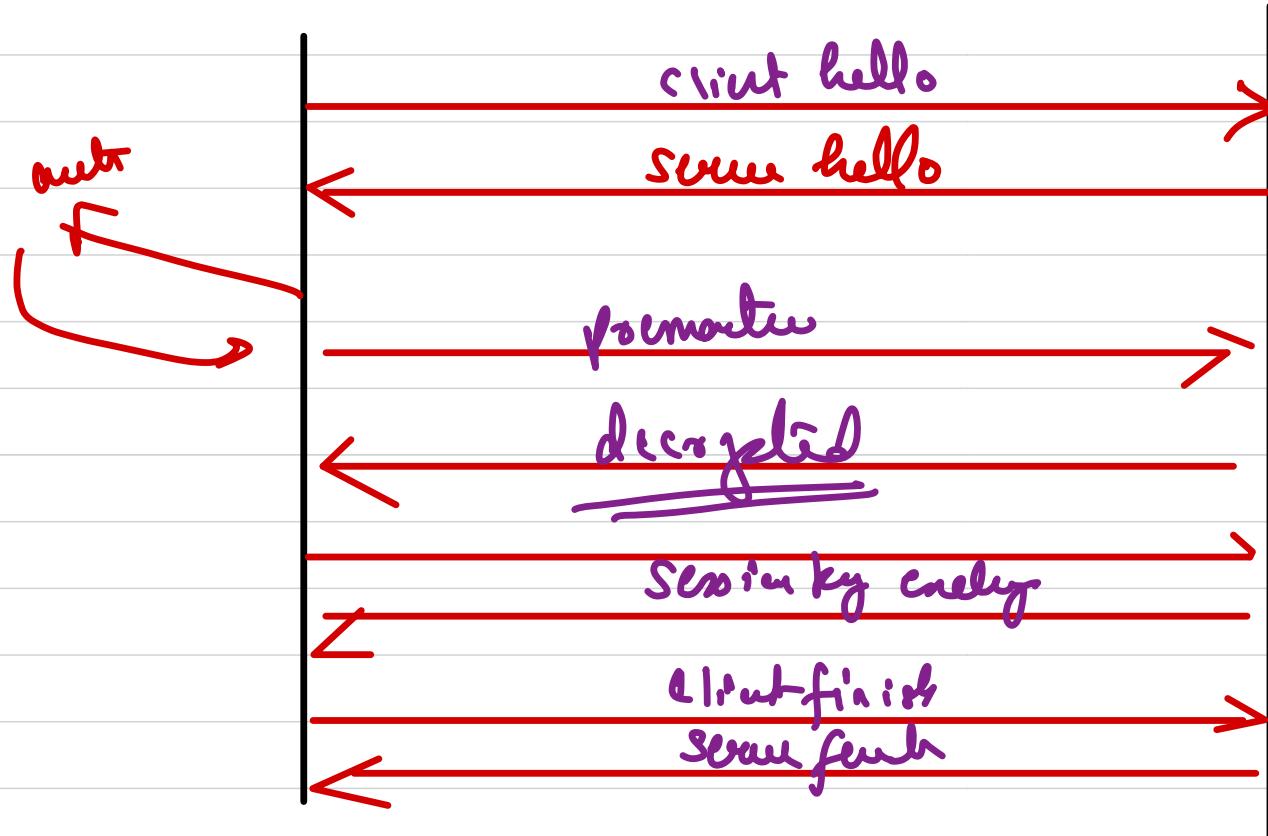
TLS handshake

SSL handshake



Client

Server



- 1) Client hello msg → version of TLS supported
encryption algos supported
random bytes of data → client random
- 2) A Server hello msg → SSL certi
encryption algos
random bytes of data → server random
- 3) Authentication → Client verifies the certi →

- 4) Premaster Secret \rightarrow Client sends another set of random bytes encrypted by public key
- 5) Private key used \rightarrow Server decrypts the premaster
- 6) Session key created \rightarrow Both client & server generate session keys from client random & server random & premaster secret.
- 7) Client is ready \rightarrow Sends a finished msg that is encrypted with Session key.

8) Server is ready \rightarrow Server also sends finished msg
to client encrypted with session key ~~key~~

Database Management System

In the earlier times → ~~file based systems~~

Problems

- 1) Data redundancy
- 2) Data inconsistency
- 3) Security issues
- 4) Concurrent access
- 5) Difficult data access

* **Databases** It is a shared collection of data

Properties of Databases

- # DB represents real world entities and relationships , which is also called as universe of discourse
- # there is some logical consistency in the data
- # DB are created for specific users-

facilities provided by database:

- 1) Define the data in logical manner
- 2) Easy manipulation of data
- 3) Easy sharing of data between multiple people & apps
- 4) Smooth way of adding or deleting data.



Students



Courses



Department



Data Models

To hide the implementation details of a database we use data models which helps us in achieving data abstraction.

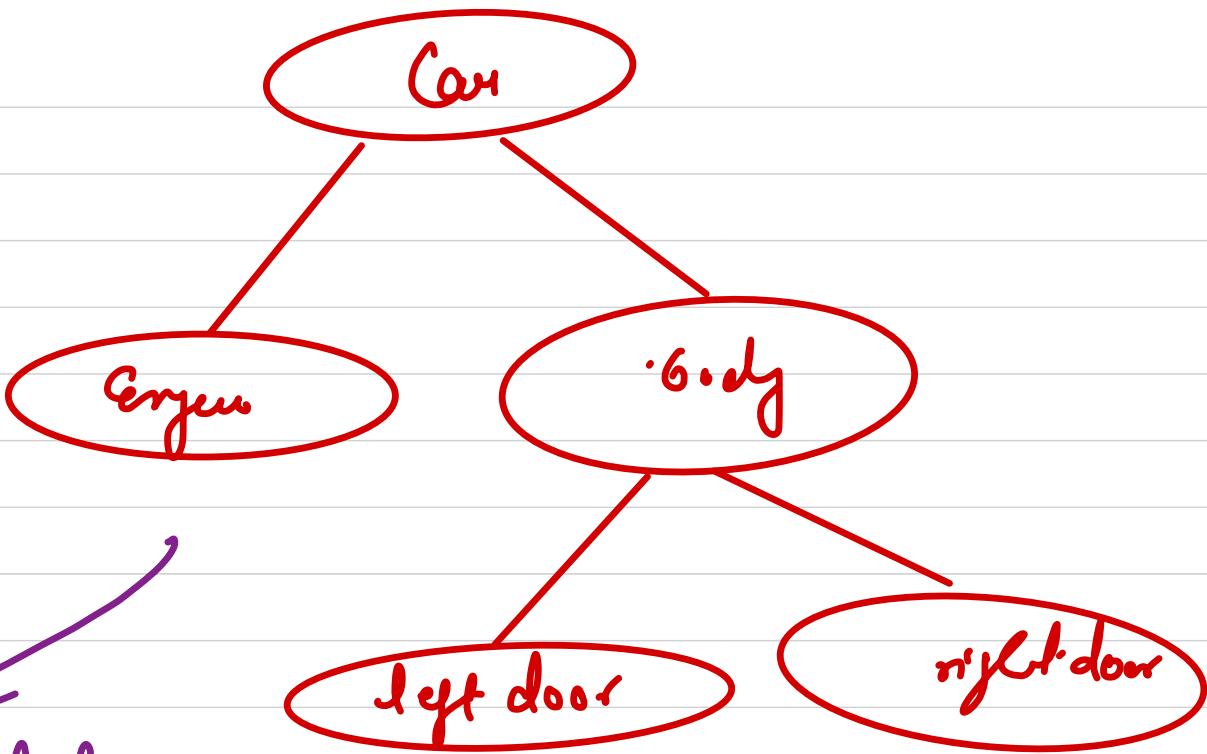
data model is a collection of notations for describing the data, its relationships & constraints

Types of data models →

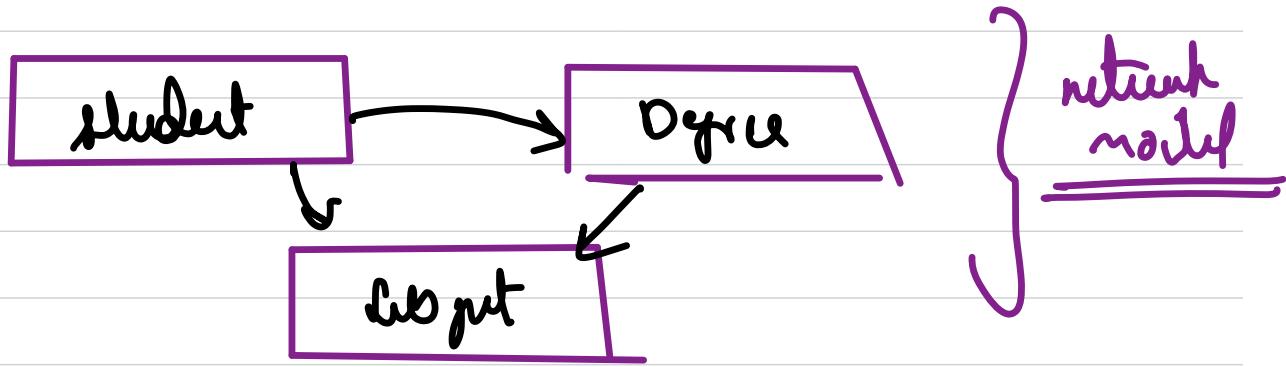
1) High level conceptual data model → We use
ER diagram (Entity relationship diagram)

2) Record based data model →

- ↳ Hierarchical
- ↳ Network
- ↳ Relational



Hierarchical
data model



Physical data model

Database Schema → It is the **blueprint** of the database

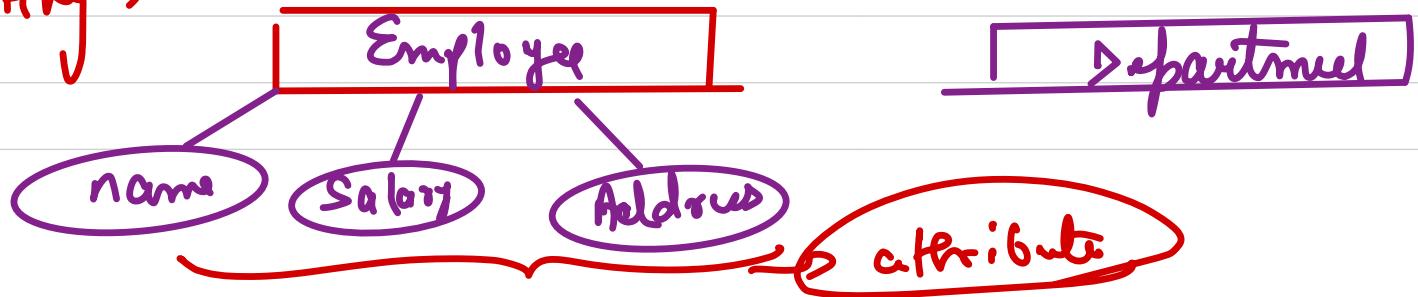
Diff → Data Model | Data Schema

E-R Model

Entity → There are real world objects

Relationship → this is how the entities are
related -

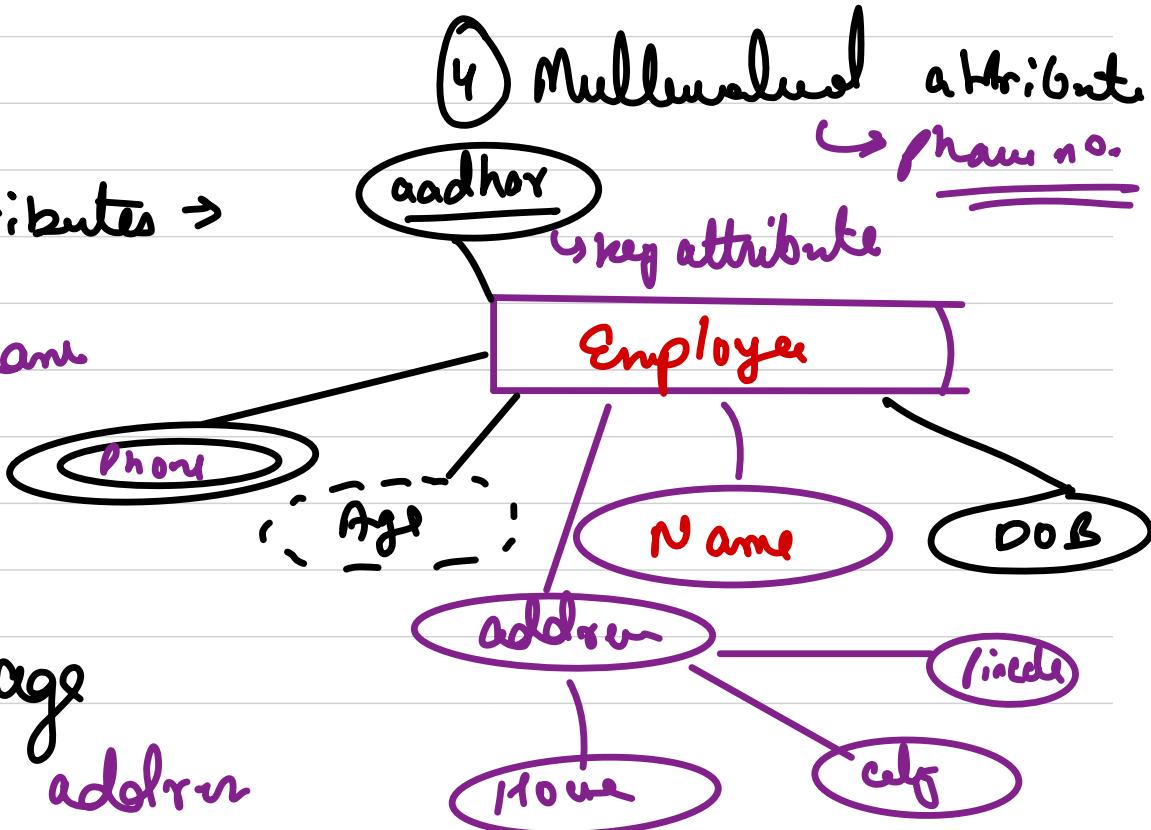
Entity →



* Attributes → These are properties that define the database.

Types of attributes →

① Simple → name



② Derived → age

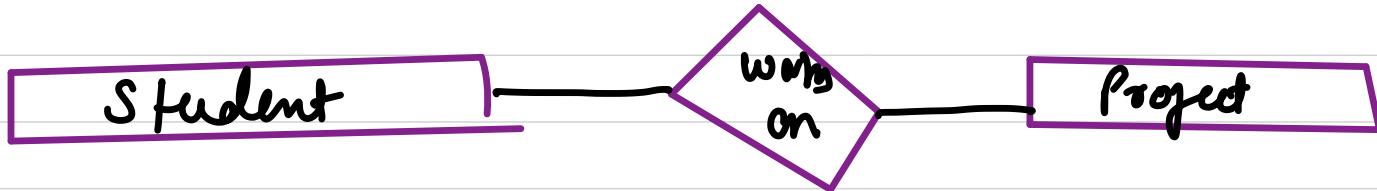
③ Composite → address

keys → In an ER diagram, an imp constraint on the entities is the KEY or uniqueness attribute. This unique attribute is also called as primary key -

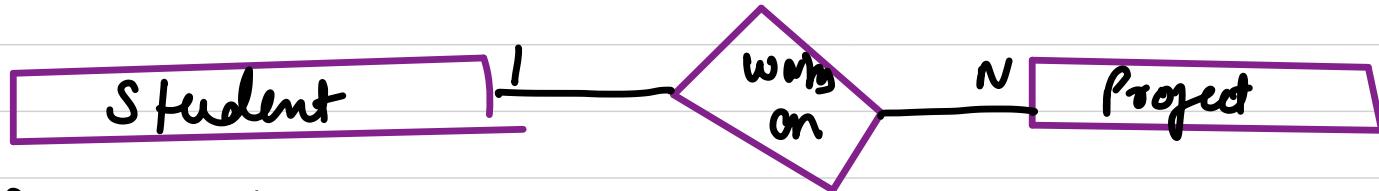
But, sometimes we need 2 or more than 2 attributes to uniquely identify a record. Then these type of attributes are called composite keys.

Relationship

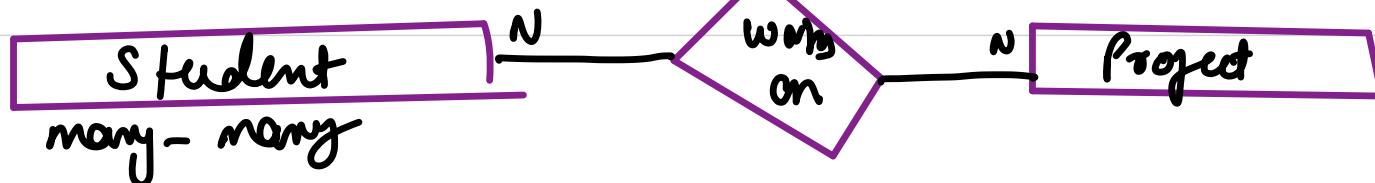
Binary - relationships



One-one relationship

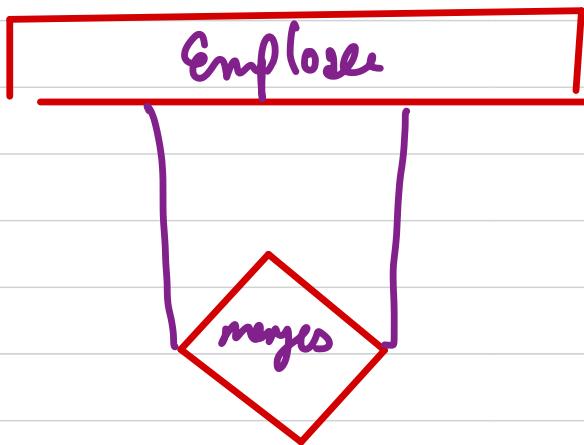


one - many

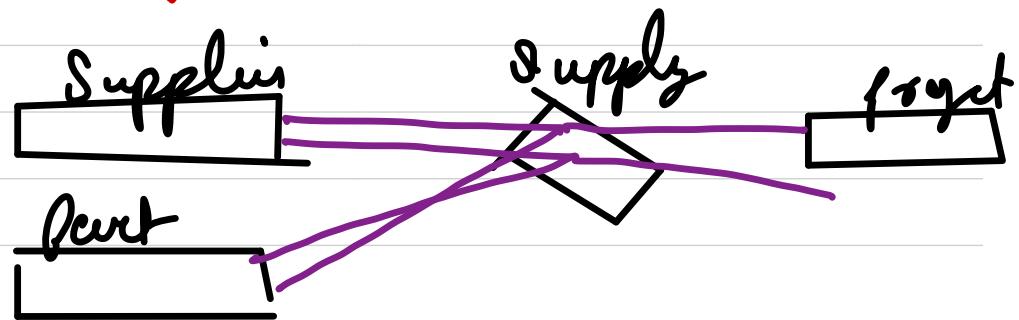


many - many

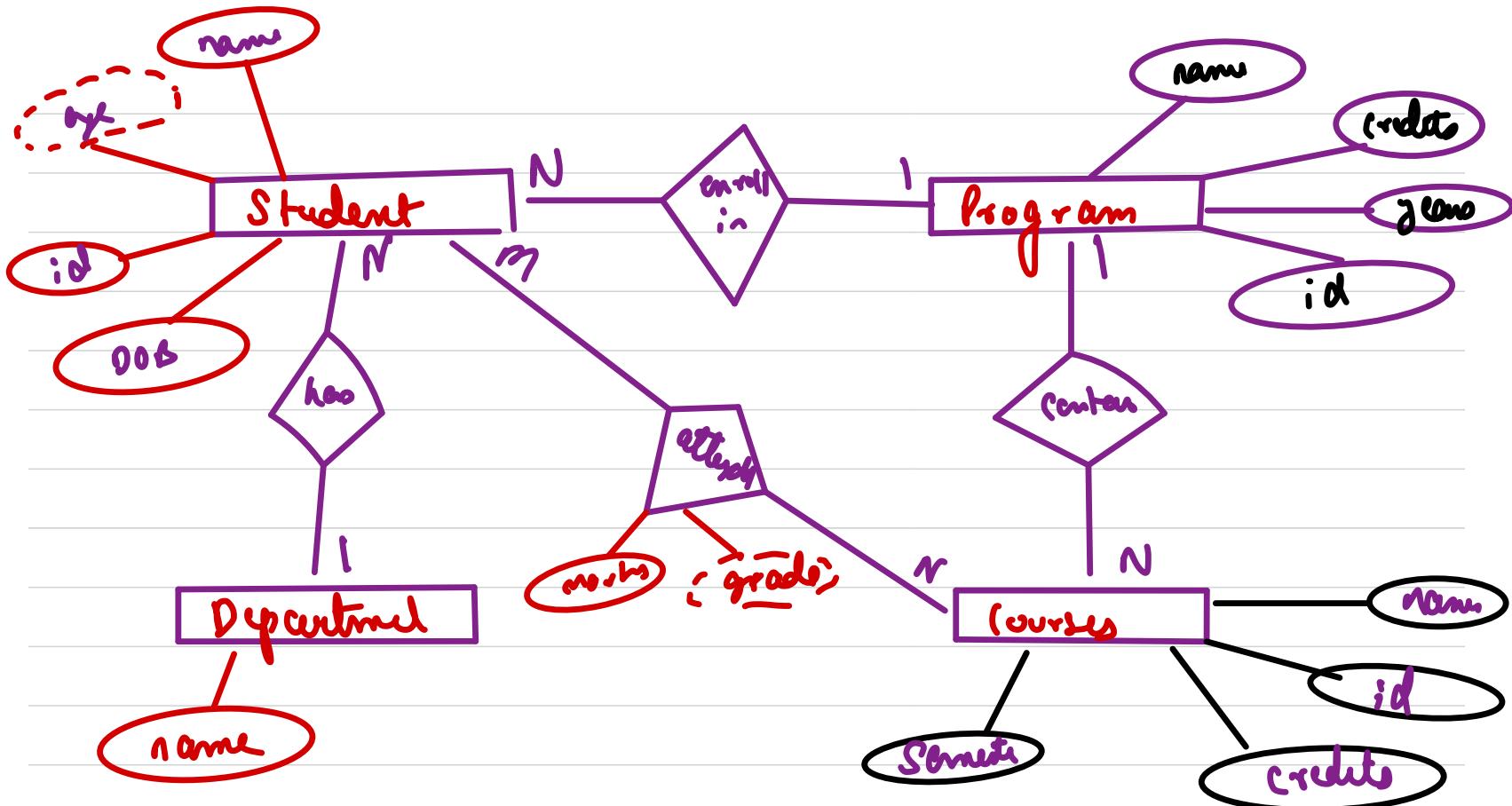
Unary Relationship



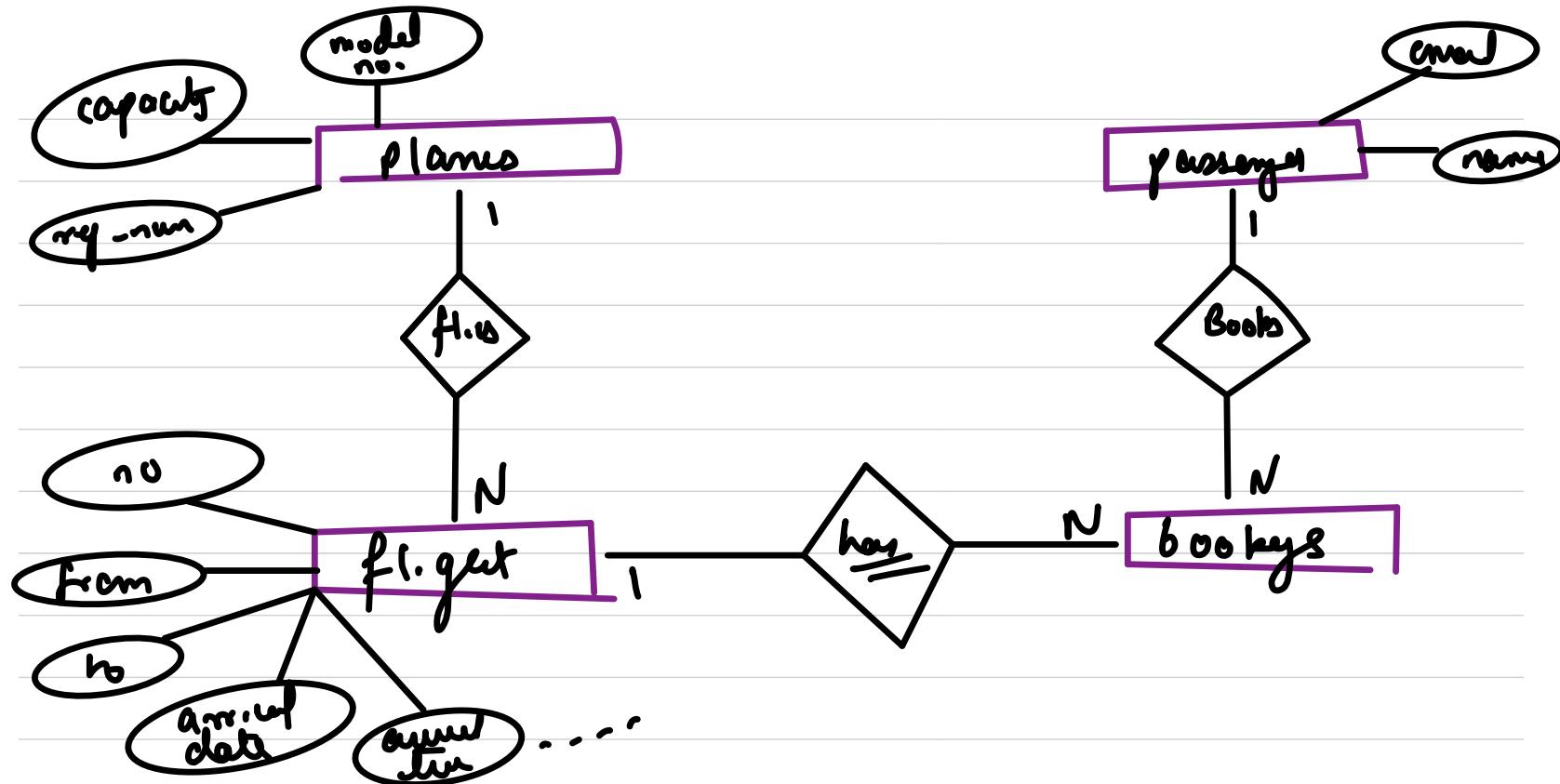
Unary Relationship



try to create ER diagram of university db.



Draw an ER diagram of flight database →



OBMS

Relational Data Model

So actual entities are represented as tables & data as values (rows) and attributes of the entities are represented by columns.

ROBMS → MySQL, PostgreSQL, OracleDB etc

columns → attributes
rows → Tuples

Properties →

- 1) Each row is unique
- 2) We try to store co atomic value as possible.
- 3) Sequence of columns is not relevant.
- 4) Every col has unique - name

Student

S.No	Name	DOB	age
1			
2			
3			
4			



→ RDBMS mostly support Schemas

→ # Primary key → single attribute to uniquely identify a record

Composite key →

Super key → It's a set of attributes. But has the potential to uniquely identify a record

Candidate key → Minimal set of attributes from super that uniquely identifies data.

alternate key \rightarrow candidate key other than primary key.

foreign key \rightarrow a group of attributes that establishes link b/w 2 tables.

Student		
S-id	name	phn
1	a	---
2	b	---

Course		
C-id	name	dept
1	c	1
2	d	2

Student-Course	
S-id	C-id
1	1
2	2

↗

functional dependency

functional dependency is relationship between 2 attributes. Generally we show this relationship between primary key and other non-key attribute.

for any relation R , attribute Y is functionally dependent on X if for every valued instance of X , the value of X uniquely identifies $\underline{\underline{Y}}$.

$$X \longrightarrow Y$$

determinant

dependent

id	name	age

$$id \longrightarrow age$$
$$id \longrightarrow name$$

Armstrong's axioms These are set of rules
that are applicable on functional dependencies.

i) Axiom of reflexivity \rightarrow

if y is a subset of x , then x determine y

$$y \subseteq x \text{ then } x \rightarrow y$$

address \rightarrow h-no, street, state ...

h-no \subseteq address

Address \rightarrow h-no

Axiom of augmentation

It is also called as partial dependency.

if X determines Y , then X_2 determines Y_2

where X_2 is $X \cup Z$ and Y_2 is $Y \cup Z$

if $X \rightarrow Y$ then $X_2 \rightarrow Y_2$

S-id	C-id	name	address	age	grade	date of comple
------	------	------	---------	-----	-------	-------------------

$(S\text{-id}, C\text{-id}) \rightarrow$ composite key \rightarrow primary key

s-id → name

s-id → address

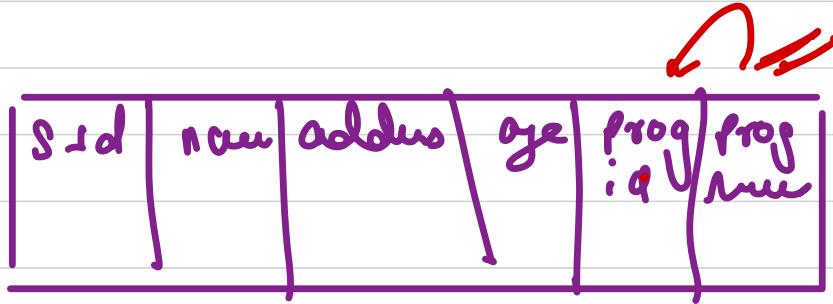
s-id → age

This is not a desirable situation, because every non-key attribute has to be fully dependent on the primary key not just a part.

This violated axiom of augmentable

axiom of transitivity

if $x \rightarrow y$ and $y \rightarrow z$ then $x \rightarrow z$



attribute closure

Closure of an attribute X , is a set of all attributes that are functionally dependent on X .

It is denoted by X^+ which means what all attributes X completely determines

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

$\Sigma \rightarrow A$, $\Sigma \rightarrow D$, $A \rightarrow C$, $A \rightarrow D$, $A\Sigma \rightarrow F$, $Ah \rightarrow K$

Find Closer of E

$E \rightarrow A$

$\Sigma \rightarrow D$

$E \rightarrow AD$

$A \rightarrow C$

$E \rightarrow \Sigma$

$E \rightarrow \underline{ADC}\bar{E}$

$A\Sigma \rightarrow F$

$E \rightarrow \underline{A\Sigma DC}\bar{F}$



$R(A, B, C, D, \varepsilon, f)$

$B \rightarrow C$ $BC \rightarrow AD$, $D \rightarrow \varepsilon$, $CF \rightarrow B$ · find B^+

$B^+ \rightarrow B$

$B^+ \rightarrow BC$

$B^+ \rightarrow BCAD$

$B^+ \rightarrow BCAD\varepsilon$

Normalization

It is the branch of relational theory that provides design insights & help us to determine redundancy in the table.

- ↳ 1NF
- ↳ 2NF
- ↳ 3NF
- ↳ BCNF Biju's odd normal form.

1NF → first normal form → It states that

domain of an attribute must include only atomic values, (simple, undivisible). Also it discourages usage of a set of values in a column of aggregates.

Student

name	roll no	Courses
John	101	DBMS, OS
Michael	102	DSA, CN

1NF

name	roll no	Courses
John	101	DBMS
John	101	OS
Michael	102	CN
Michael	102	DSA

$\mathcal{Q}NF$ (second normal form) \rightarrow A relation must be in $1NF$ and must not contain any partial dependency. (non key attribute are dependent on proper subset of composite key)

S-id	C-id	C-fc
1	C1	1000
2	C2	1500
3	C3	2000
4	C4	1000
5	C5	1000
6	C6	3000

{S-id, C-id}

\rightarrow Partial
dependency as
 $C-id \rightarrow C-fc$

2NF →

Student - course

S-id	C-id
1	C1
2	C2
3	C4
4	C3
5	C1
6	C5

Course

C-id	fee
C1	1000
C2	1500
C3	1000
C4	2000
C5	3000

3NF (Third normal form) \rightarrow The relation

should be in 2NF. and don't have transitive dependency.

dependency:

S-ID	Subj ID	marks	Exam Type	Total marks	
					X

Score

srd	sub id	math	Exam id

Exam

id	type	<u>total mark</u>

BCNF \rightarrow relation should be already in

3NF -

And for any dependency $A \rightarrow B$, A should be a

Super key. (it means A can never be a non
key attribute)

s_id	subject	professor
------	---------	-----------

Professor \rightarrow Subject
 $\times \times \times$

S-icd	P-icd
-------	-------

P-icd	name	subject
-------	------	---------

Cust Name	Item	Shipping Address	Newsletter	Supplier	Supplier Phone	Price
Alan Smith	Xbox One	35 Palm St, Miami	Xbox News	Microsoft	(800) BUY-XBOX	250
Roger Banks	PlayStation 4	47 Campus Rd, Boston	PlayStation News	Sony	(800) BUY-SONY	300
Evan Wilson	Xbox One, PS Vita	28 Rock Av, Denver	Xbox News, PlayStation News	Wholesale	Toll Free	450
Alan Smith	PlayStation 4	47 Campus Rd, Boston	PlayStation News	Sony	(800) BUY-SONY	300

INF 2.?

INF

custname	item	address	newsletter	supplier	Supp-Ph	Price
Evan	XboxOne	---	XBox news	---	—	—
Evan	PSVita	---	PS news	—	—	—

↓

Customer Table

cust_id	name	address	newstatus
1	John Doe	123 Main St	Active

Item Table

item	supplier	open_qty	last_update
1001	Supplier A	50	2023-10-01



Customer-item-join

cust id	name	address	news letter
------------	------	---------	----------------

item	Supplier	price
------	----------	-------

cust id	idea
------------	------

Supplier	phone
----------	-------

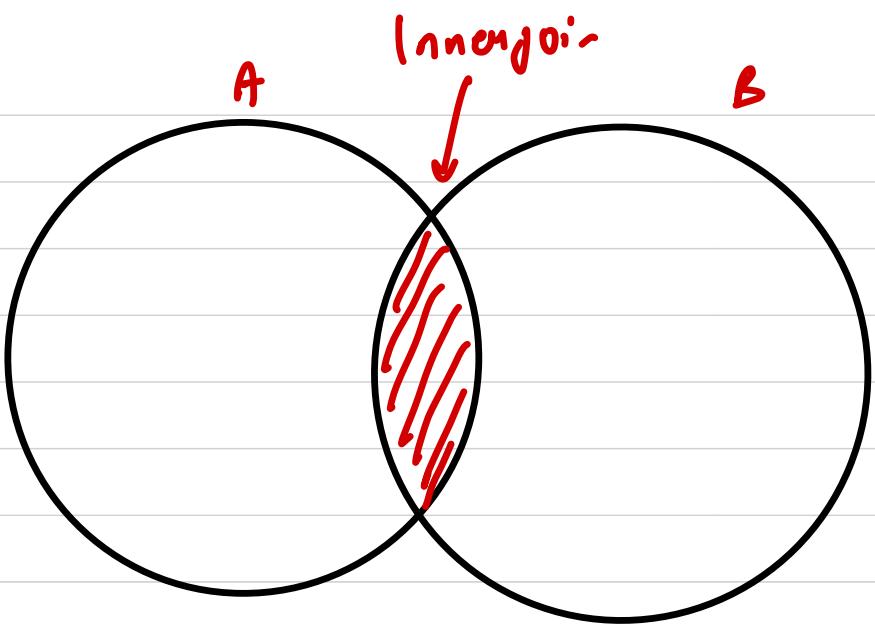
4^{th} normal form \rightarrow BCNF already &
no multi valued depend.

→ create a database → Entertainment

create 2 tables → Movie → (id, name, -)

Cinema → (name, id, date)

→ add some entries



→ Actors :

→ Actor Movies (movieId, Actor Id)

→ display for each actor we display the
movie & the halls

Indexing

A lot of times we have to search inside our database. In the normal orientation the db will kind of do a linear search.

Ex → `SELECT * from Students WHERE id=34;`

In databases if we know flat on what column there will be a lot search ops we can index that column.

Prefix string searches → Trie

improve integer value search) (custom value search

→ B+ Trees

B+ Trees

- It is a self balancing tree.
- Inside B+ tree the final value are present at the leaf node.
- all leaves are at same level.
- Root node has atleast 2 children.

- Each node except the root, can have a max of m children & at least $\frac{m}{2}$ children.
- Each node stores a key, a node can store maximum of $(m-1)$ keys & min of $\left[\frac{m}{2}\right] - 1$ keys
- all the leaf nodes are connected as a ll.

Inserction in B+ trees

before inserting an element we need to take care that

- * root has atleast 2 children.
- * each node except root has max m & min $m/2$ children.
- * each node can contain max $(m-1)$ keys & min $\lceil m/2 \rceil - 1$ Keys.

Case 1

if the leaf is not full , insert the key into the leaf in inc order.

Case 2

if the leaf is full, insert the key into leaf node
in inc order & rebalance the tree

— Break the node at $\frac{m}{2}$ th pos

— Add $\frac{m}{2}$ th key to parent as well

— if parent is also full follow above steps :-

Deletion in B+ trees

Case-1

The key to be deleted is present only at the leaf node not in the internal nodes.

1) There is more than the min no. of keys in the node, delete the entry.

2) There is exactly min no. of keys in the node. Delete the key and borrow a key from immediate sibling. Add the median key of sibling node to the parent.

Case II

The key to be deleted is present in the internal nodes as well.

1. If there is more than min no. of keys in the node simply delete the key, from leaf & internal nodes as well. Fill the empty space with inorder successor.

2. If there is exact min no. of keys then delete the key & borrow from the ~~left~~ child.

3. If empty space is generated above the immediate parent, fill it with inorder successor.

ACID properties in RDBMS

A → atomicity

C → consistency

I → isolation

D → durability

transaction It is a single logical unit of task which reads & sometimes even modifies content of the database.

Atomically says that either the whole transaction
take place at once or doesn't happen at all.



Consistency → Integrity constraint must be maintained
in the db before & after transaction.

Durability → When one transaction is completed

then it's changes persist in the db & the disk even
in cases of failure.

Isolation

→ RD Bms should concurrent manage transactions

such that no inconsistency of db occurs.

T₁

Read(x)

$x = x + 10$

Write(x)

Read(y)

$y = y - 5$

Write(y)

T₂

READ(x)

READ(y)

$z = x + y$

Write(z)

Operating System

* Memory Management →

Let's say we've a C program

```
#include <stdio.h>
int main () {
    char str[] = "Hello";
    printf ("%s", str);
```

}

```
#include <stdio.h>
int main () {
    char str[] = "Hello";
    printf ("%s", str);
}
```

what happens when we compile and run a C program.

gcc hello.c

executable (a.out)

./a.out

process

Stored in HDD

executed in RAM

Process →

- A program in execution is called process.
- Process is present inside RAM
- few components of process are -

* Executable instructions

* Call stack

* Heap

* State of the process in OS

State : → list of open files, related process, registers etc

all of this is managed by OS

What is an operating system?

software

It is an intermediary between end user and computer hardware. It provides suitable environment to create and execute programs.

- Resource Management
- Process Management
- Memory & Storage Management
- Security

How RAM manages process?

We can have multiple processes running simultaneously and sharing RAM. But Ram is a limited process.

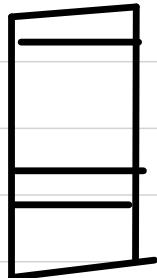
Now, irrespective of the fact on how multiple processes are managed by OS, their memory map should always be present in RAM.

Process 1

Process 2

Process 3

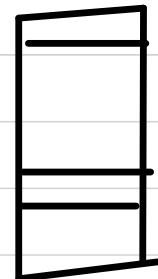
RAM



Memory Map P_1



Memory Map P_2



Memory Map P_3

Let's see how RAM manages multiple processes -

There are multiple modes of RAM management

① Single Continuous model -

→ No sharing

→ at one time only one process occupies RAM

→ when one process is completed then
only other process starts

→ limitations

→ multiple process management is not very good.



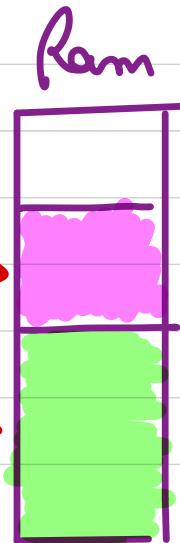
→ process memory size is restricted by RAM size..

② Partition Model

as long as sufficient contiguous space is available new process are allocated in the memory.

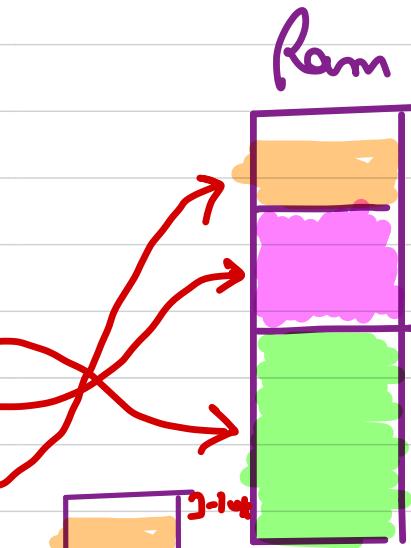
It maintains a partition table

addresses	Size	Proc No.	Usage
0x0	120 K	3	In use
120 K	60 K	1	In use



Let's say we get a new process of Size 20k

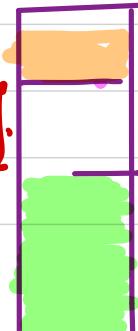
addresses	Size	Proc N.	Usage
0x0 120 K	120 K	3	In use
180 K	20 K	4	In use



→ Say this is completed → This 60k []

memory is deallocated

a new process of 68k size comes



We can't allocate it
due to lack of 68k
Coniguous memor

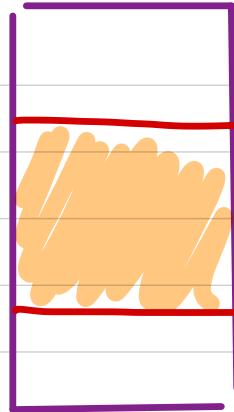
This leads to a phenomenon called as fragmentation

fragmentation is a situation when free block are too small to satisfy an memory request.

few small left over memory is getting wasted and if this small chunk inc, then amount of reusable memory decreases.

* Internal

* External



How partition model handles fragmentations ??

① first fit → It may make fragmentation worse.

② Best fit → this may affect performance-

during deallocation of process we can merge free partitions , but it will take time .

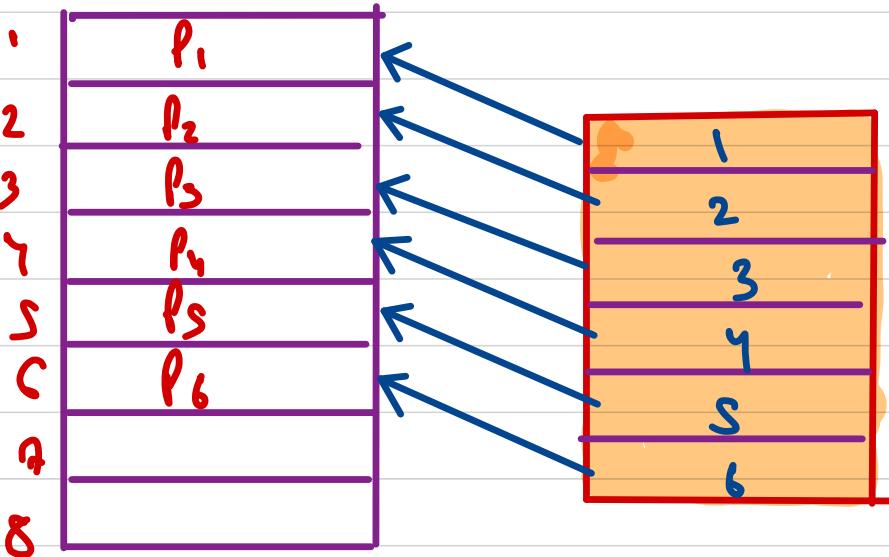
Current OS does better job in memory management
using virtual memory & segmentation

Virtual Memory

Ram is divided / split into fixed size partitions

called as Page frames. and multiple processes

it was typically about 4KB.



RAM

thus process block
is allocated to frames

process also splits
into blocks of
equal size when
Block = page
size

block	page frame
1	1
2	2
3	3
4	4
5	5

Page table =

Because of per process page table, blocks of processes need to be allocated in continuous frames. The page frames can be identified by page table.

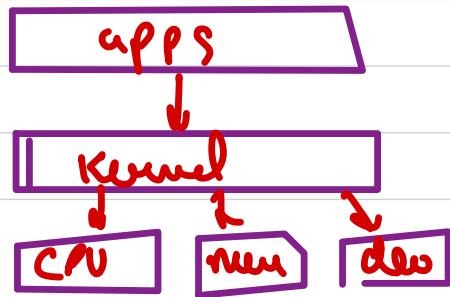
So every memory access has an additional overhead of the lookup in the page table. This can be eliminated by putting a cache called as TCB (Translational lookaside buffer).

* KERNEL → It is a central component of OS

that manages operations of computer ~~hardware~~

It is a bridge between applications & data

processing performed at hardware. very inter-process communication calls & system calls.



*** Note →** Memory associated with the process is in the user region of memory whereas the process page table is in the kernel region.

Depending on active process, the active page table will vary. So a process won't be able to access page frames of other processes.

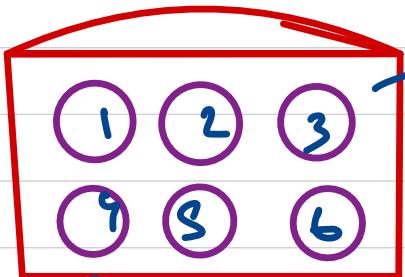
~~Q2)~~ Do we really need to load all blocks into memory before process starts executing ??

No

Not all parts of the program are accessed simultaneously. In fact some code may never be executed.

So this concept is called as Demand Paging.

Virtual memory is a feature of OS, that enables computer to be able to compensate shortage of physical mem. by transferring pages of data from RAM to Disk storage.



Swap Space on Disk

In our secondary storage, as page is allocated (Swap space)
All blocks of exec is present
On demand blocks will be loaded in the ram.

block	frame	present bit
1	1	1
2	-	0
3	-	0
4	-	0
5	-	0
6	3	1

- * Pages are loaded from disk to ram , only when needed.
- * A present bit in table represents if block is in RAM or not.

Say, block 6 of executing process wants to access block³ & block 3 is not loaded in RAM , Then it will check if block 3 is loaded or not using present bit . If the bit is 0, processor issues a page fault interrupt, then triggers the OS to load the page.

Situation → Say OS wants to load a block in
a page frame, but no page frames are free for
new block to be allocated. Then what happens? =