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 with 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

1957 → ARPA (advanced research project agency) → 1960's-70's → 1980's → 1990's

1957
↓
Soviet
↓
Sputnik

ARPA
↓
US

1960's-70's
↓
Communication System for ARPA's Computers to talk
↓
1969
ARPANET

1980's
TCP/IP
↓
Internet

1990's
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 requires 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

mongo → 27017

★ 49152 – 65535 → Dynamic ports

Access Networks

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

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

DSL (Digital Subscriber Line)

→ DSL uses the existing telephone groundwork 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 (7 layer)

- Application
- Presentation
- Session
- Transport
- Network
- Data Link
- Physical

TCP/IP (5 layer)

- Application
- Transport
- Network
- Data Link
- Physical

Application layer → email server, chat server, browser

Presentation → Presentation of data, compression encryption

Session → User session managed

Transport → Divides big chunk of data coming from above to small chunks. & mgt these chunks.

Network → how route of packets will be done in the network

Data link layer → error / flow control, multiplexing & demultiplexing, handle address

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 & recovery can also be done.

Where it exists??

↳ End systems

↳ Instant messaging

↳ www

↳ VoIP

↳ email

Client - Server Architecture →

↳ It is a 2- level architecture

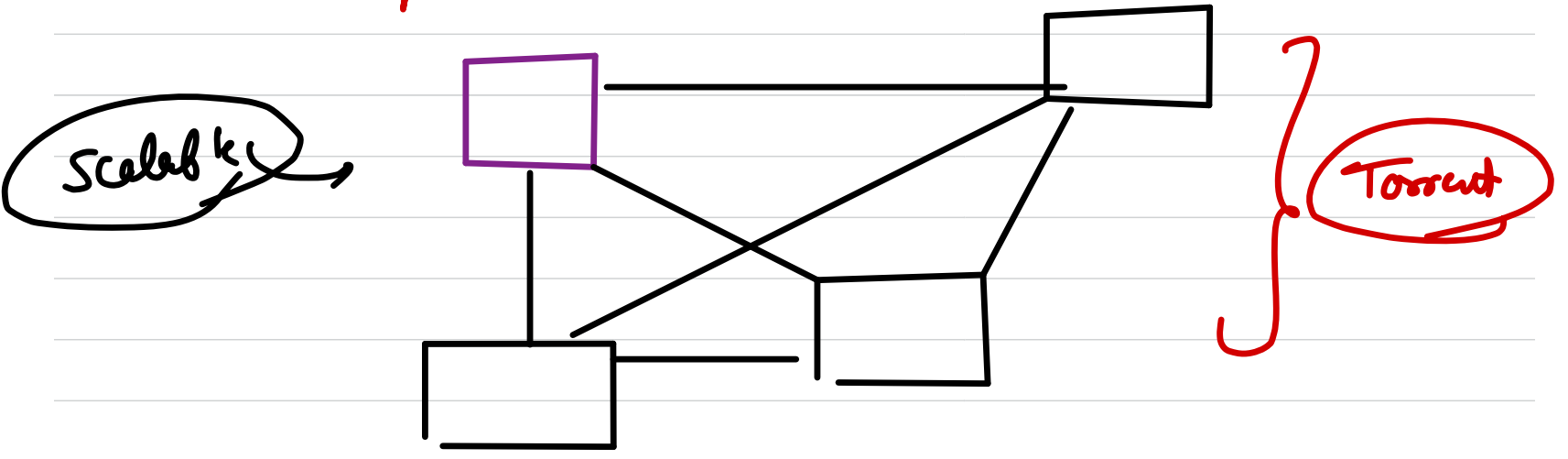
Client-side Server side

Servers → 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

HTTP

HTML

It stands for hyper text transfer protocol

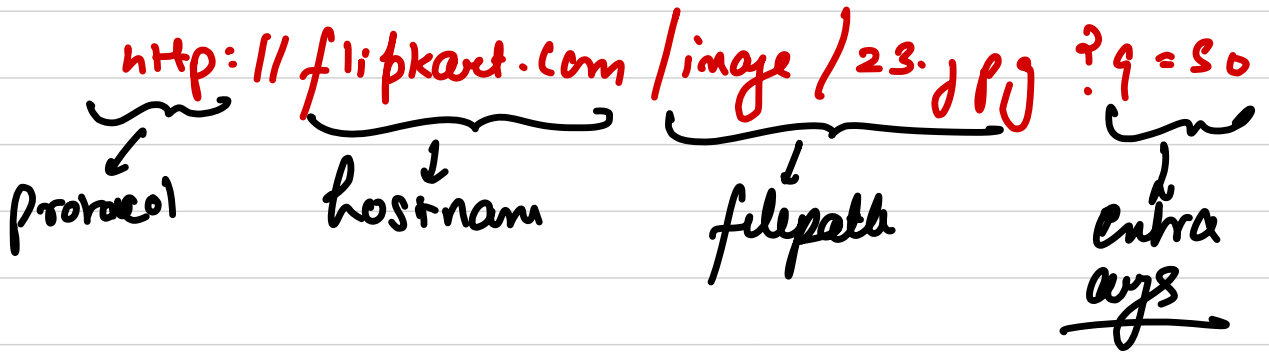
↳ **Objects** → web pages are the main objects that contains other objects.

Some other objects can be mp3 file, png, jpg etc

Every objects has a URL

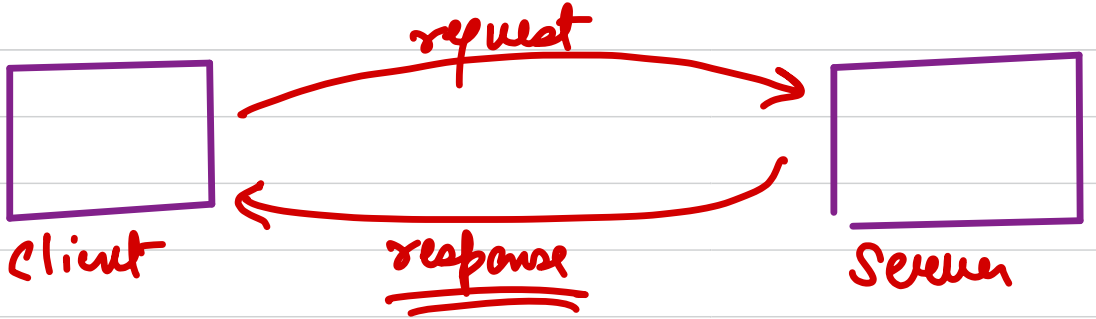
URL (uniform resource locator)

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


http://flipkart.com/image/23.jpg?q=50
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 called as request-response protocols

→ 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 2 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 req and response msgs

HTTP Req Msg →

any http message are plain ASCII text

- host
- method
- status code
- Referer-policy
- ⋮

Method	URL
Get	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 → delete an object at a given url

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

* Accept-Language → specifies the preferred language

* Connection : close

HTTP Status code:

Convention → SOAP

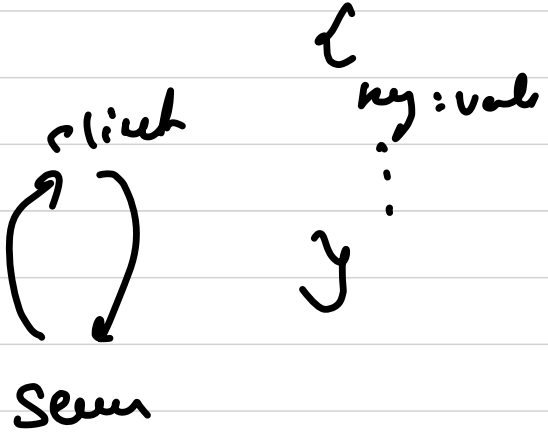
REST

GRPC

XML

JSON

protocol buffers



deal everything as a resource

deal everything as an action

Torrent

DNS

Cookie

SMTP

Cookies

↳ These are mainly concerned towards privacy.

↳ HTTP is a stateless protocol, & a lot of time 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 shared, it is sent along with subsequent http req 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 folder 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.

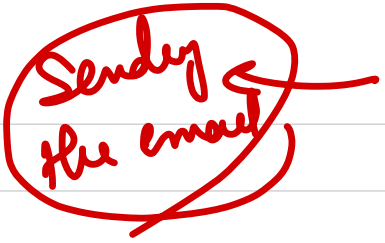
Connection for SMTP is setup on port 25.

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

How SMTP works??

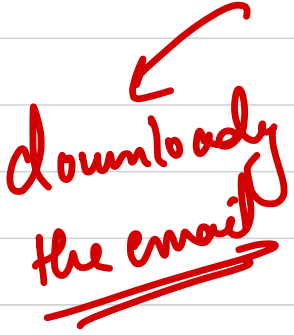
* When an Email is sent, it is sent to the Sender's SMTP server using SMTP protocol (Also the SMTP server is configured in the mail clients)
* The SMTP server places the email on a message queue.

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



SMTP → push protocol

POP3/IMAP → pull protocols



If receipient 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

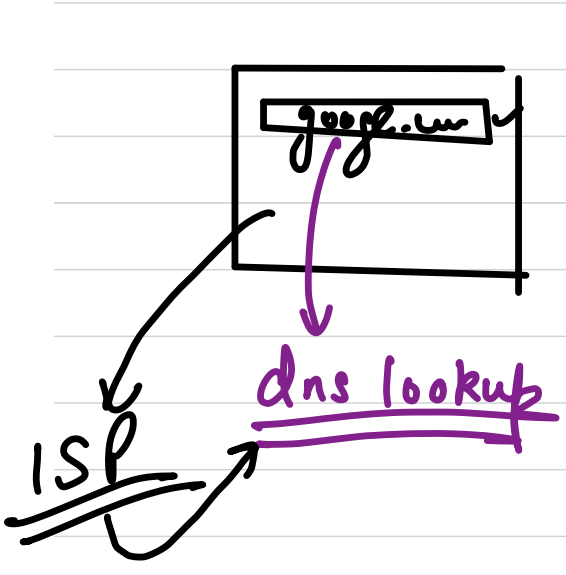
IMAP (Internet message access protocol)

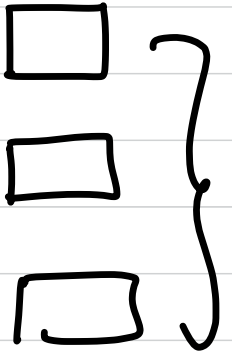
→ Emails are kept on the Server & not deleted

→ local copies of the emails are cached on each device

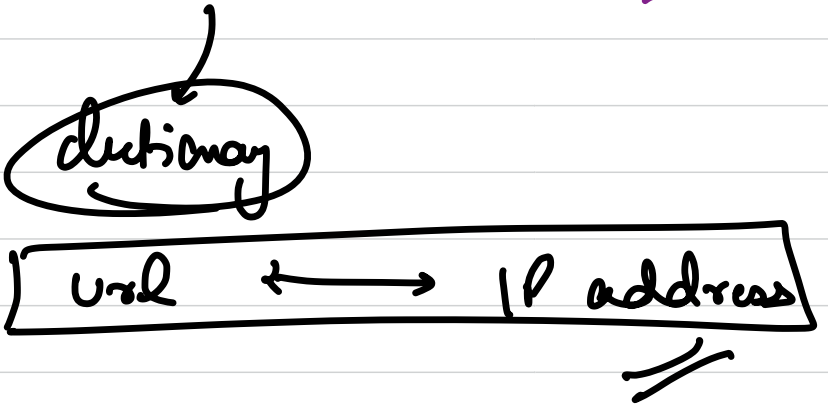
→ If an email is deleted by user manually then only it gets deleted from server.

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





DNS (domain name server)



http :// cs144.scs.stanford.edu / labs / sc.html

application protocal

host → IP address

file / resource

HOSTS.txt (URL - IP)

all the hosts were in a file hosts.txt, maintained by a network information center

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

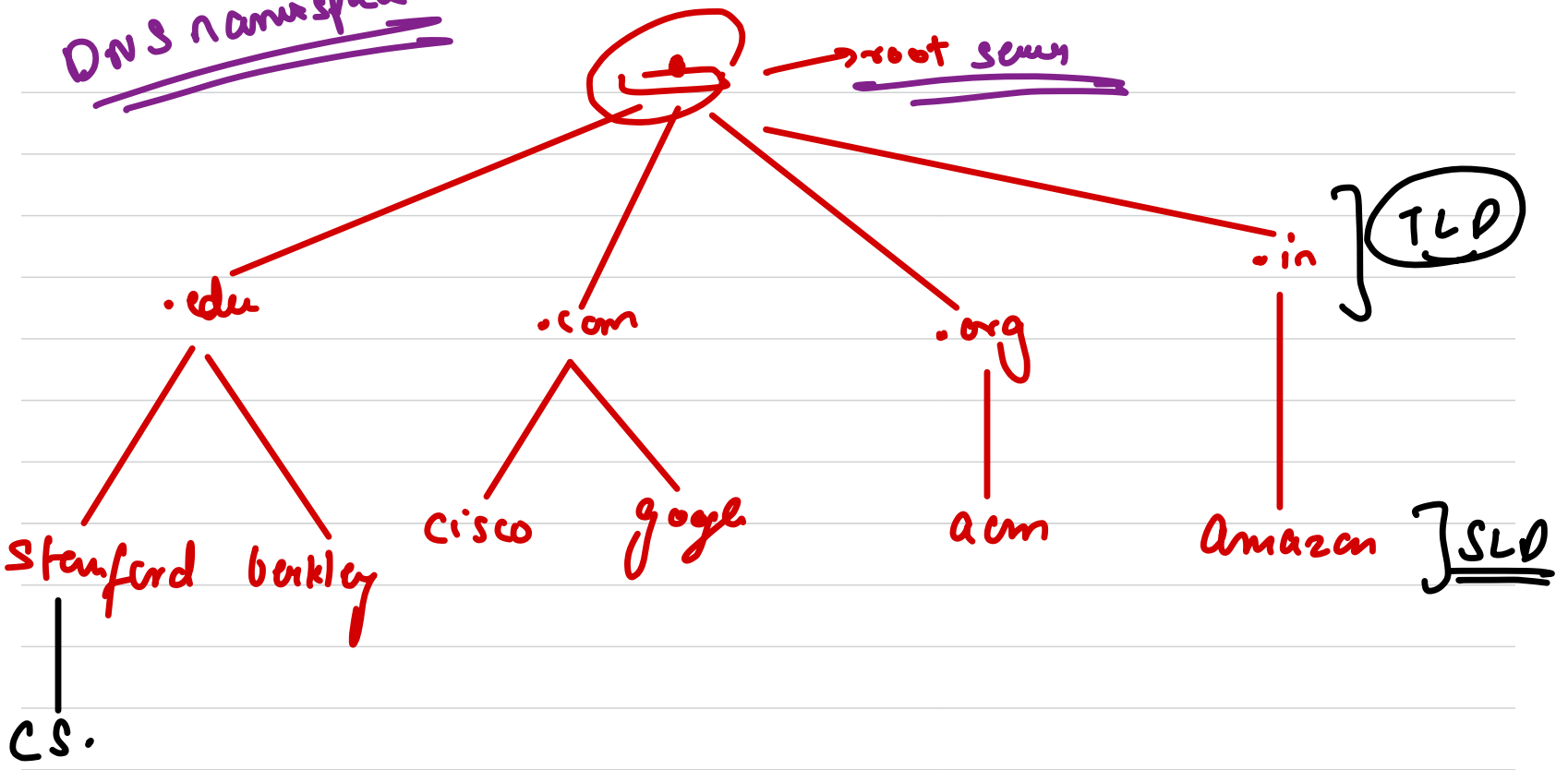
Domain Name System (DNS)

↳ maps names to address

↳ was able to handle huge records.

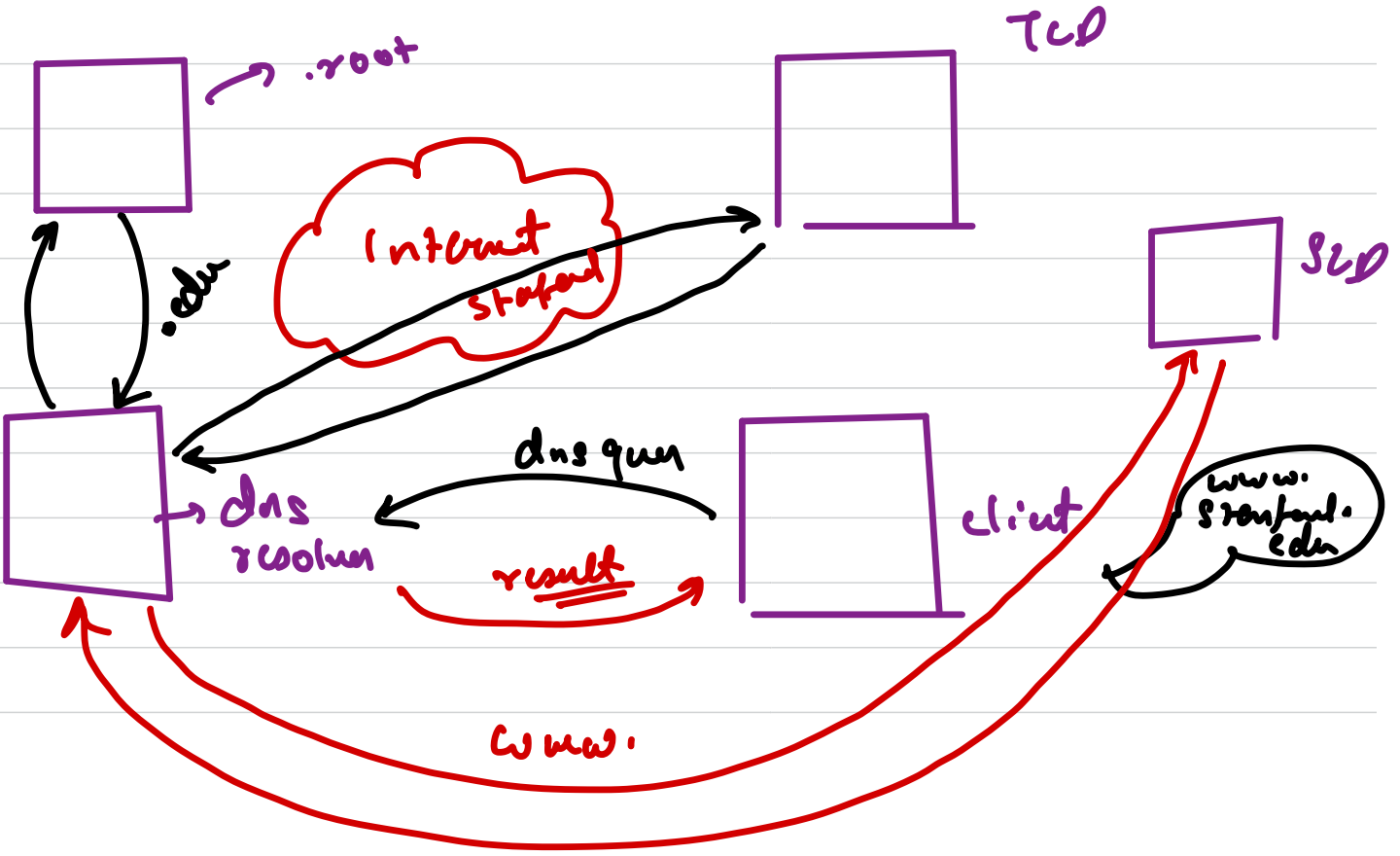
↳ Robust to failures.

It is a tree based structure


DNS namespace
.
root servers
.edu
.com
.org
.in
TLD
stanford
berkley
cisco
gogle
acm
amazon
SLD
CS.

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 share large files.

→ A bit torrent client requests files from multiple clients in parallel.

→ Small chunks of data is called pieces.

→ If a client succesfully 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 downloady a .torrent file

→ Gives info about the file they shared, like how big it is, size of it's 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 trackers (ces torrents. (distributed HT)

Q. What exactly torrent does??

(1) It breaks the file they shared into N pieces.

→ for better perf → 256 kb – 1MB

→ 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.

Peers 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.

TRASPORT 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 multiplexes & demultiplexes data.

Where it exists

↳ Transport layer also exists on end systems.

In transport layer there are 2 main protocols

1) TCP → Transfer Control protocol
2) UDP → User datagram protocol

TCP

→ While using TCP small pieces of data are called segments

Ensures reliable and in-order delivery of segments

Detects 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 datagrams

Does not ensure reliable delivery of datagrams.

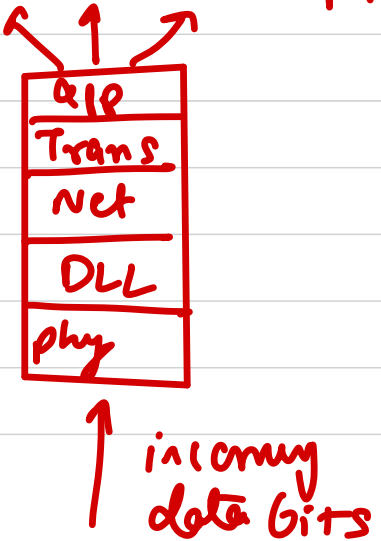
It also detects modifications but does not correct them.

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

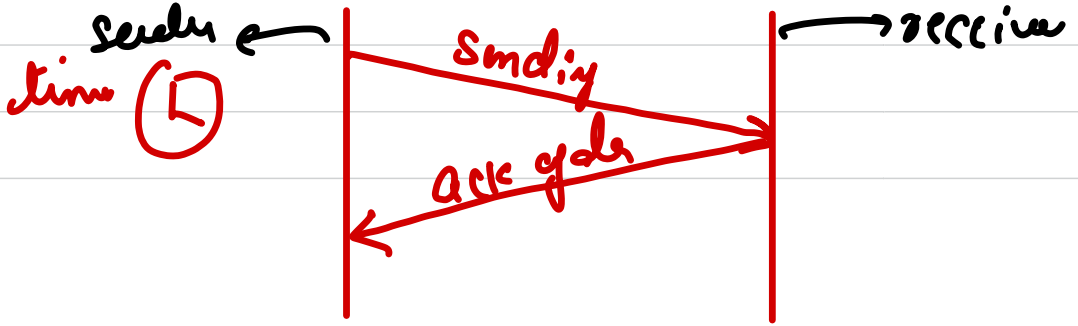
Imperfections in Network Layer.

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

Checksums → 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 →



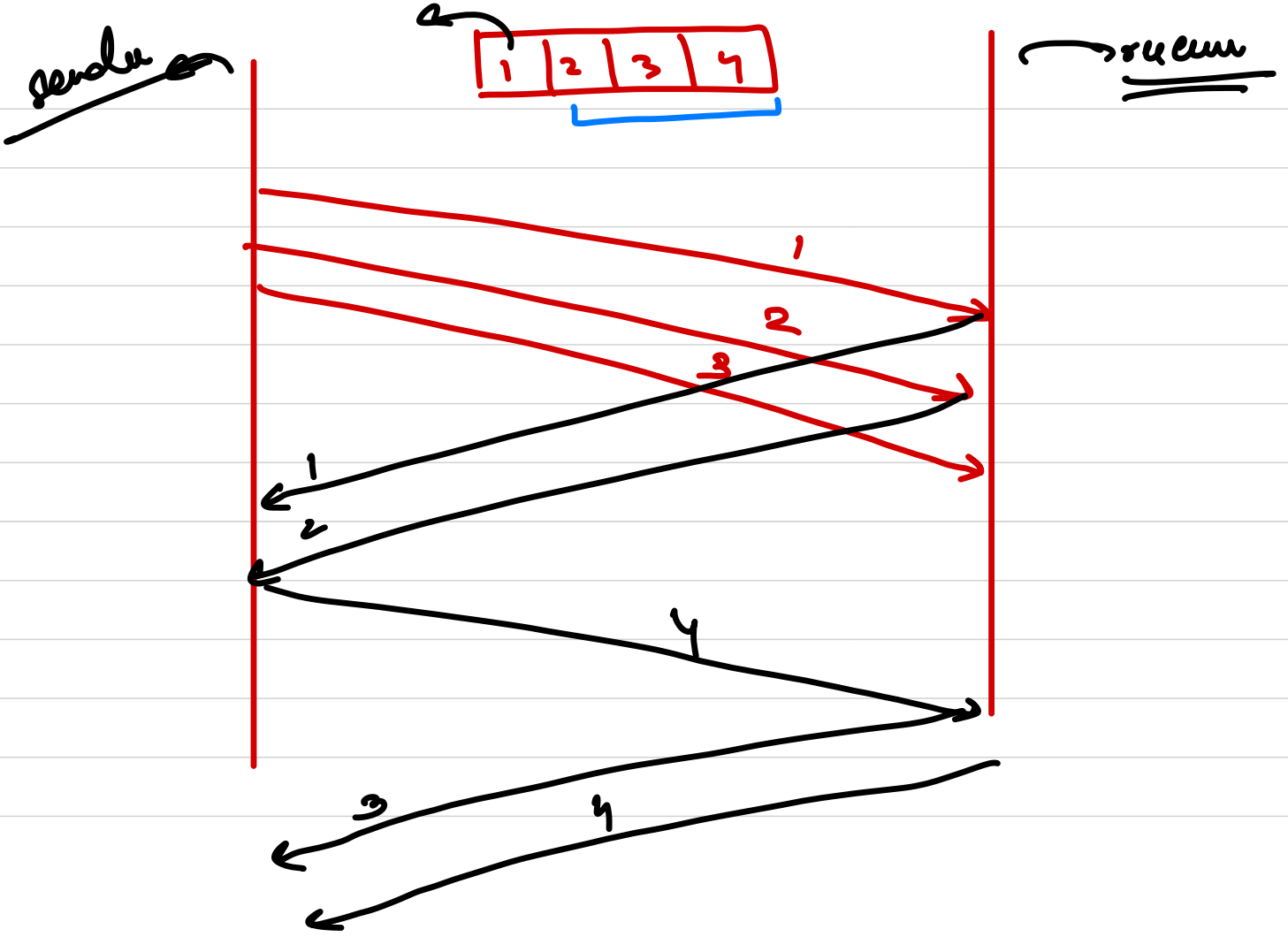
Sequence Number → unique id no for every segment

Pipelining → apps can generate data at a much higher 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.


sender
1
2
3
4
receiver
1
2
3
4
1
2
4
3
4

Detection & re-transmission of packets cannot be handled just by Sliding window protocol.

Go-Back N

It is defined at two ports in a network

1) Sender

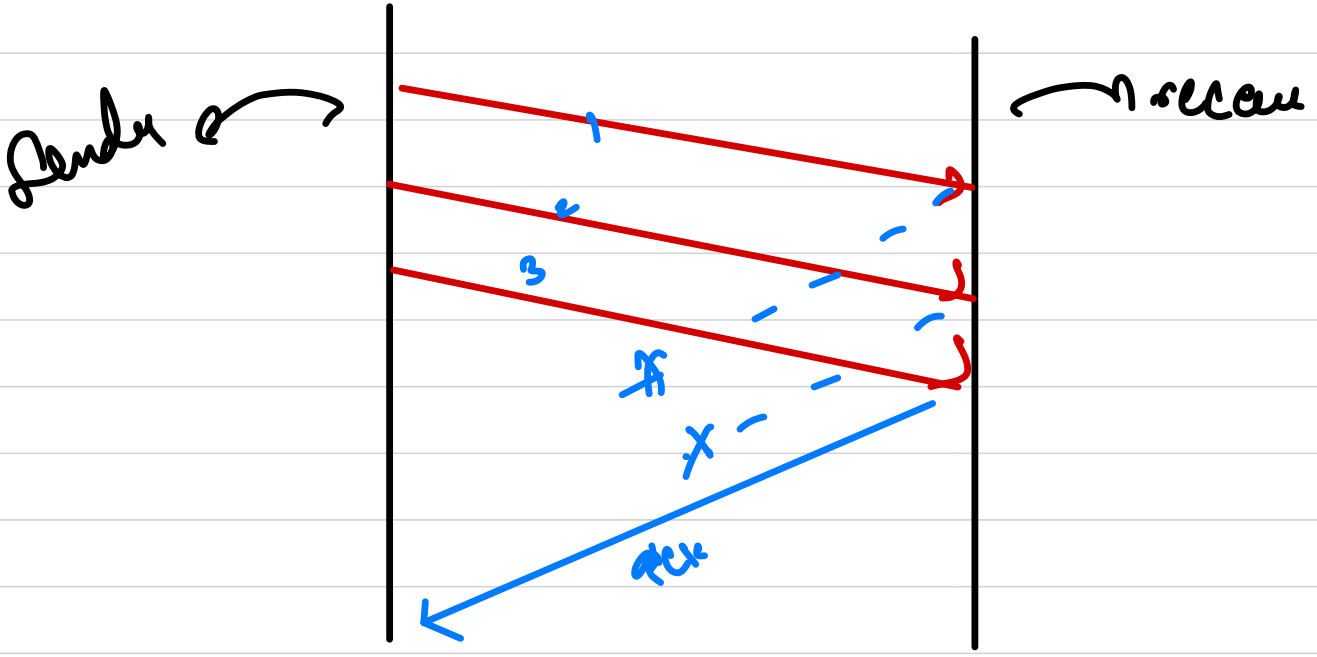
2) Receiver

→ Receiver →

↳ It only accepts that packets that come in seq

↳ any out of sequence packet is discarded.

↳ when it receives a data segment, it sends an ack containg seq no of last in-seq segment.


Sender
Receiver
1
2
3
4
x
4x

→ 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.

Selective Repeat

→ This also uses sliding window like go-back-n.

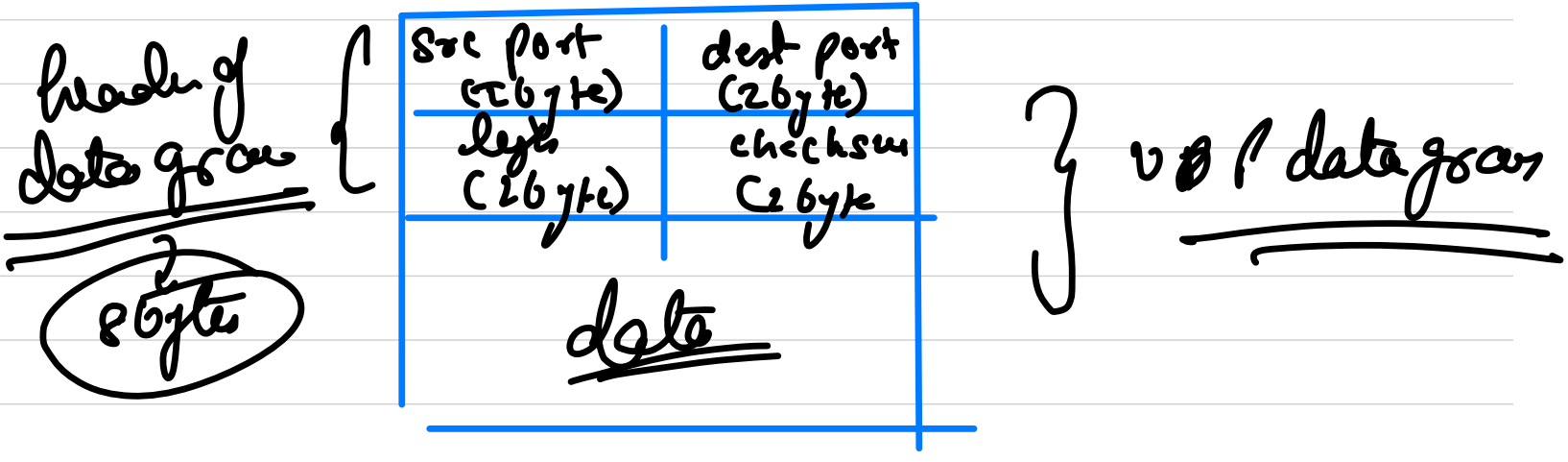
→ Window size should be less than or equal to half of the seq no.

Sender retransmits unack packets after a timeout.

Receiver ack all correct packets.

UDP (user data gram protocal)

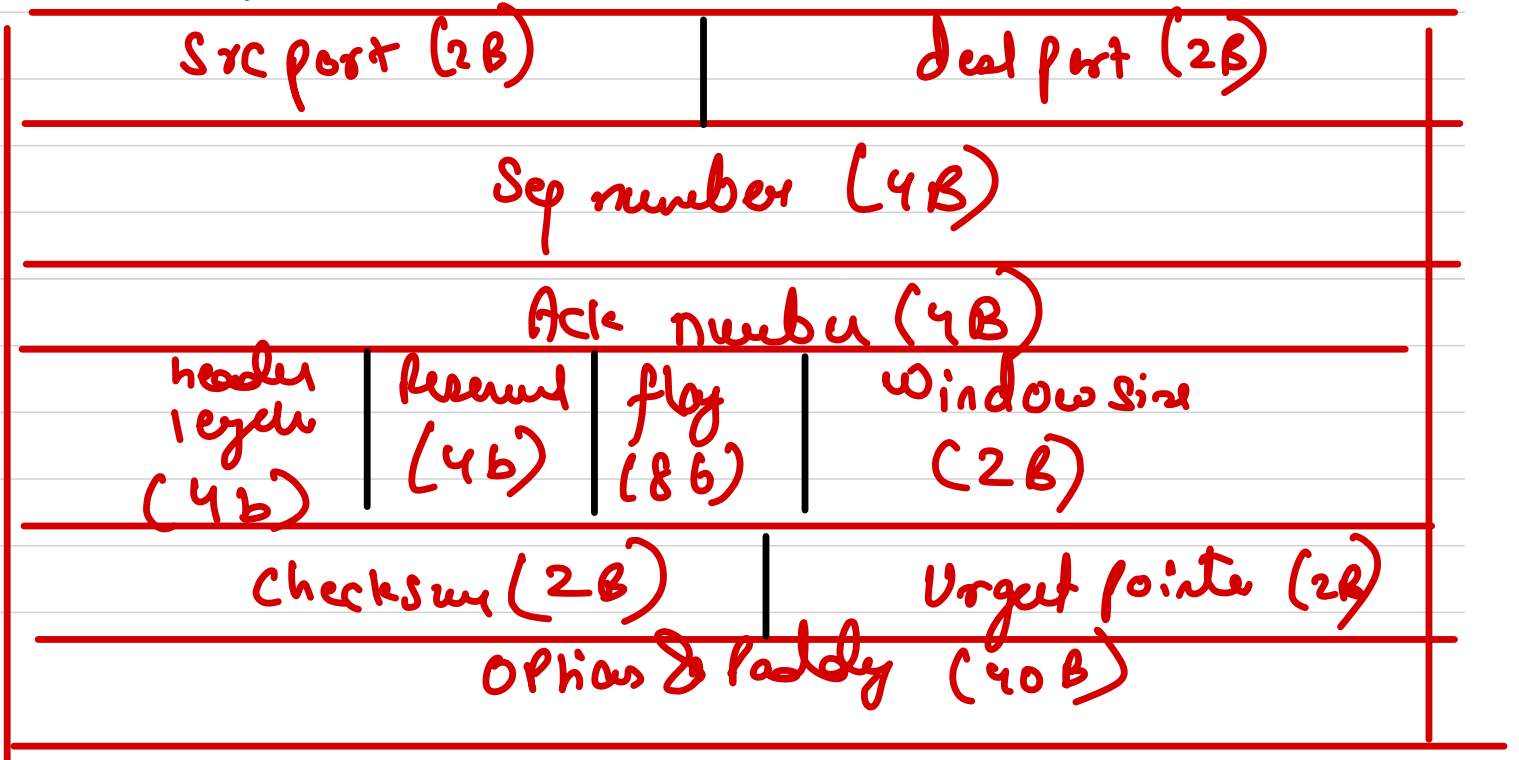
→ Non reliable

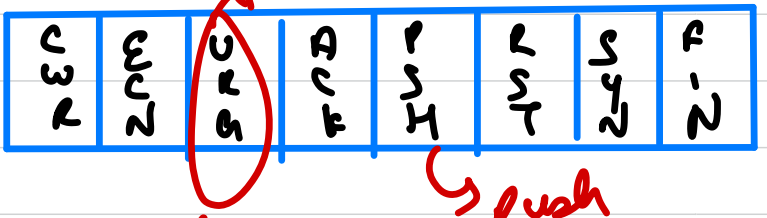


TCP (transmission control protocol)

→ reliable

→ header of TCP





ACK → acknowledgement

RST → reset

SYN → Synchronization

FIN → finish

TCP

Reliable

Connection Oriented

Error Control

flow control

full duplex

How to setup a TCP Connection

Three way handshake →

1) Initiating a Connection →

SYN flag set
Seq 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 → Sender sends to receiver.

↓
SYN is not set
ACK is set

Network Layer

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

Services of Network layer →

1) Unreliable Connectionless service →

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. It does not adapt automatically to changes in network.

1 Dynamic Routing → They have adaptable routing mechanism & routing tables.

There are 2 types of dynamic routing -

1. Link state
2. Distance vector

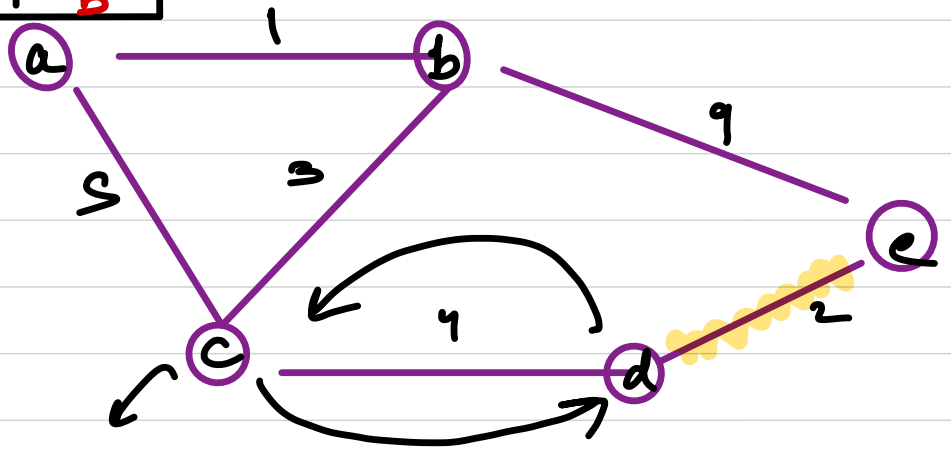

Routing
Static
Dynamic
link state
dist. vector

Distance Vector Routing

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

Bellman ford algo

dest	cost	next
c	5	c
b	1	b



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

Count to infinity problem

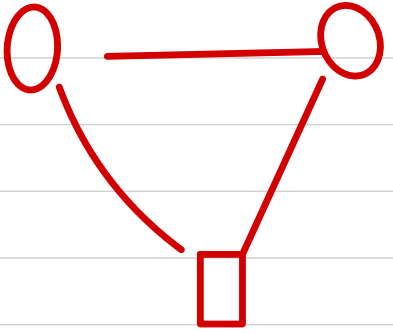
Link State Routing

- reliable flooding
- route calculation

Reliable flooding → Whenever any router boots, it sends a hello msg to the neighbours.

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

LSP → link state packet



It uses flooding to transfer it everywhere

Phase 2 → Route Calc → Dijkstra's

Data Link Layer

Responsibilities of DLL is →

To recieve packets from network layer & dealing with providing hop to hop communication between entities that are connected in physical layer.

Three types of DLL =>

1) Only 2 points of communication are involved. (P2P DLL)

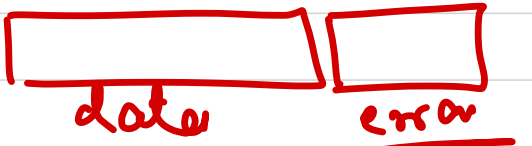
2) Broadcast Multi access => used in LAN.

3) Non-Broadcast Multi access.

⇒ Error detection → Receiver

Sender ⟷ receiver

1) The sender adds some redundant info (some r bits) as an error detection code.



parity → Even
→ odd

HTTPS

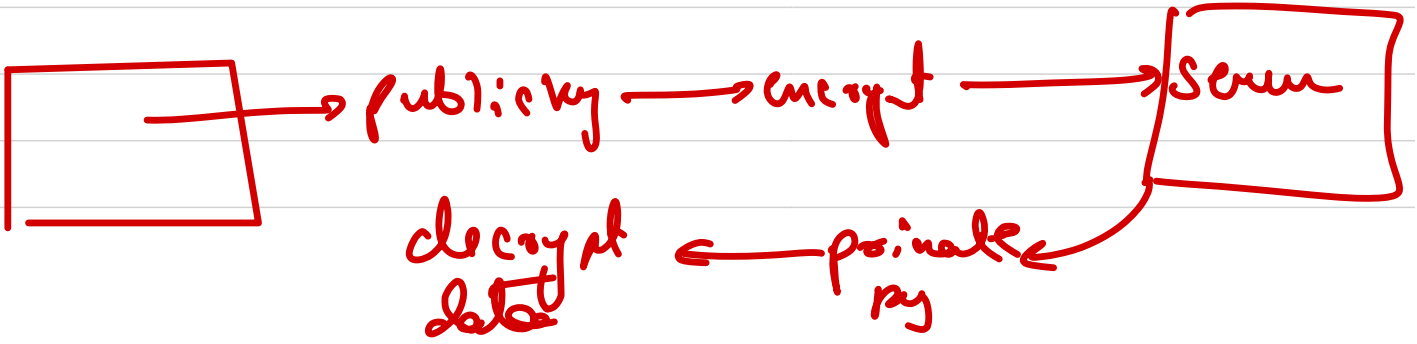
Hypertext transfer protocol Secure.

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 infra for encryption - this security system uses two different keys to encrypt communication between 2 parties.

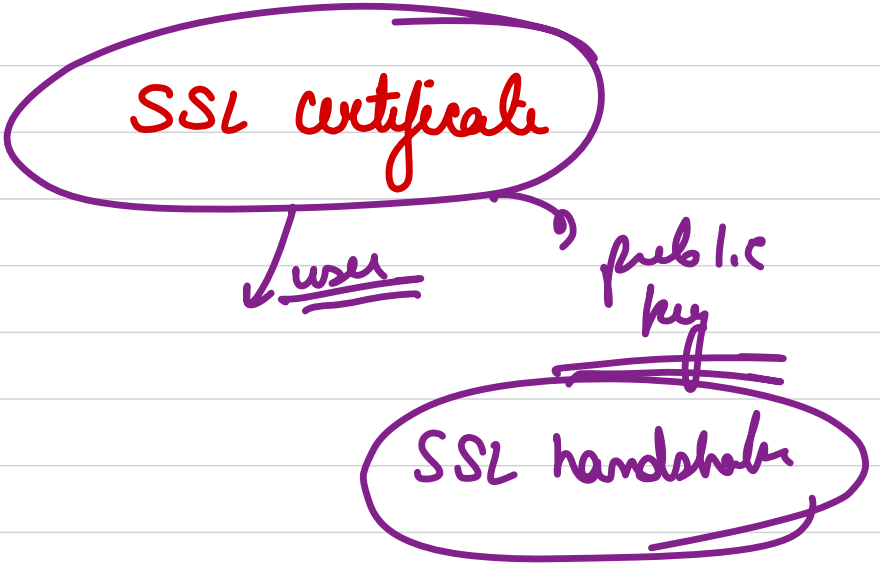
1) Public key → this key is available to everyone who wants to interact with website server in secure way. Info that is encrypted using public key can only get decrypted by private key:



2) Private key → it is controlled by website owner and is kept private from the network. This key lives on the server to decrypt info encrypted by public key.

How https is diff from HTTP

TLS/SSL encryption over http.

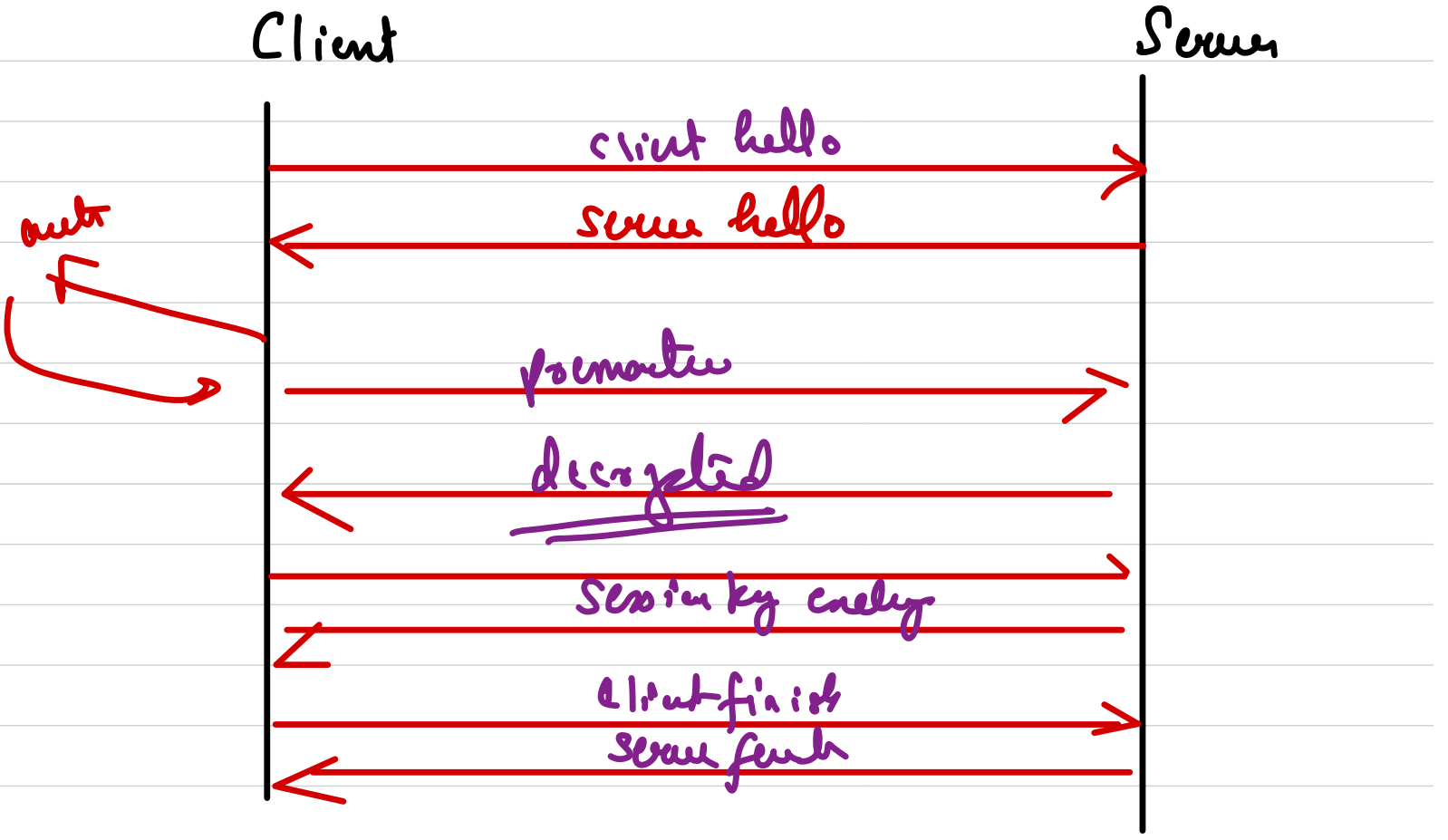


SSL certificate

It contains public key of the & key info.

- Domain name
- org / person to which certi was issued
- Certificate authority
- digital signatures
  - Expiry of certi
  - public key

TLS & SSL handshake


Client
Server
client hello
server hello
auth
premaster
decrypted
session key ready
client finish
server finish

1) Client hello msg → version of tls supported
encryption algo supported
random byte of data → client random

2) A server hello msg → SSL certi
encryption algo
random byte of data → server random

3) Authentication → Client verifies the certi

4) Premaster Secret → Client Sends another set of random bytes encrypted by public key

5) Private key used → Server decrypts the premaster

6) Session key created → Both client & Server generate session keys from client random & Server random & premaster secret.

7) Client is ready → Sends a finished msg that is encrypted with Session key.

8) Server is ready → Server also sends finished msg to client encrypted with session 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

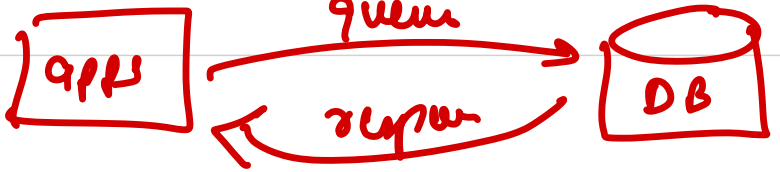
* Databases It is a shared collection of data

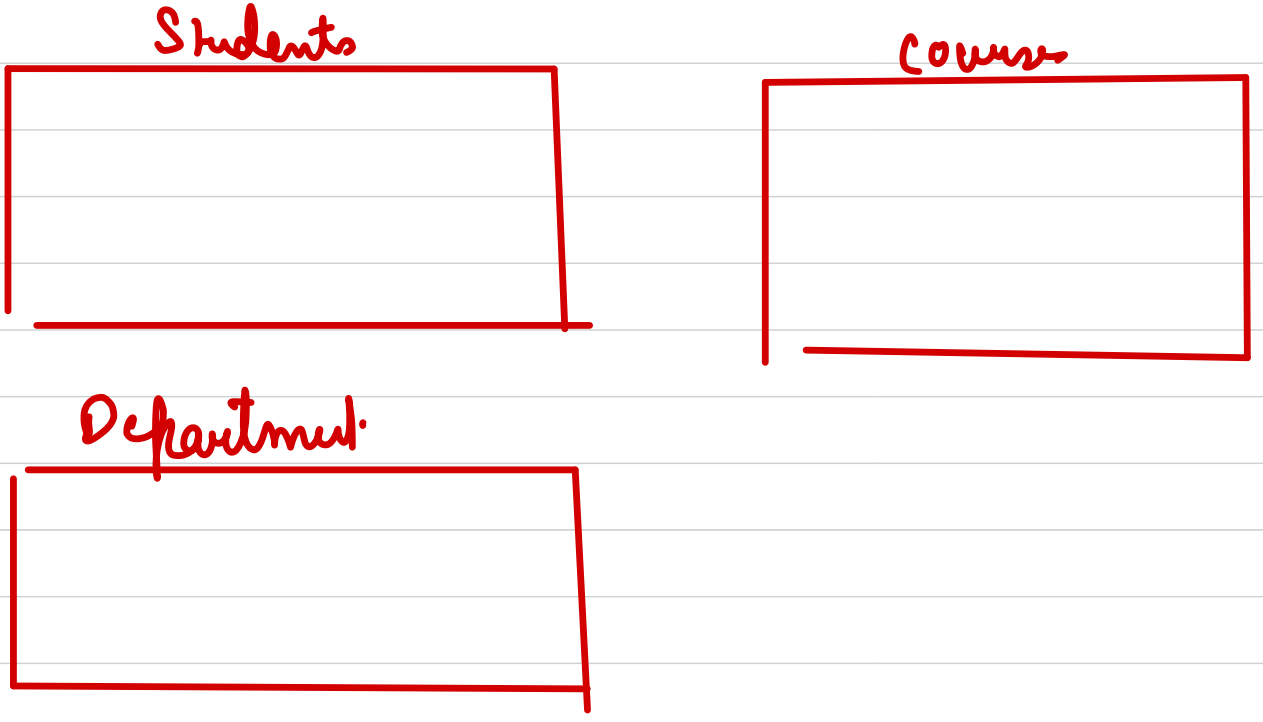
Properties of Databases

DB represents real world entities and relationships, which is also called as universe of discourse
There is some logical coherency in the data
DB are created for specific usecases.

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

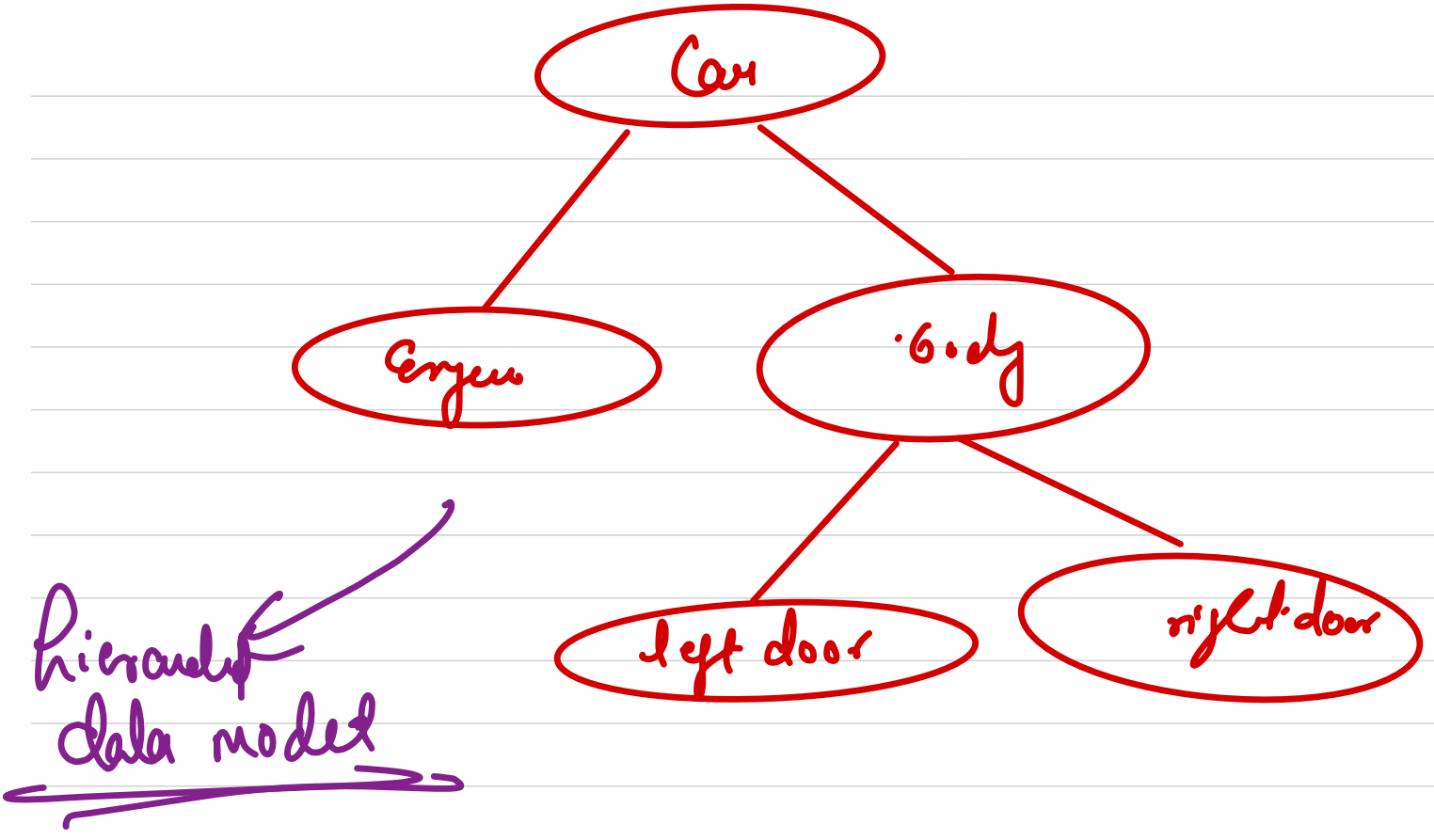
To hide the implementation details of a database we use data models. which helps us in achieving data abstraction.

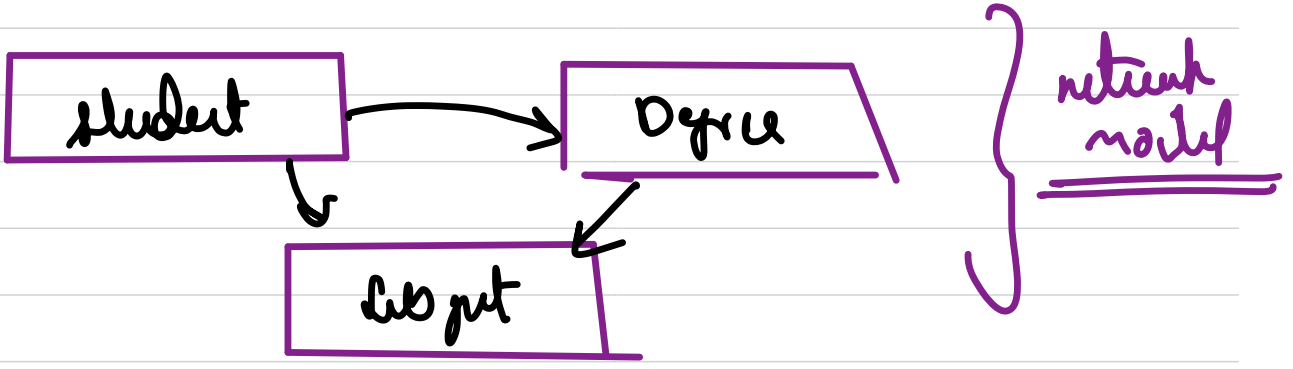
data model is a collection of notations for describing the data, it's relationships & constraints

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


Car
Engine
Body
left door
right door
hierarchy
data model



Physical data model

Database Schema → It is the blueprint of the database

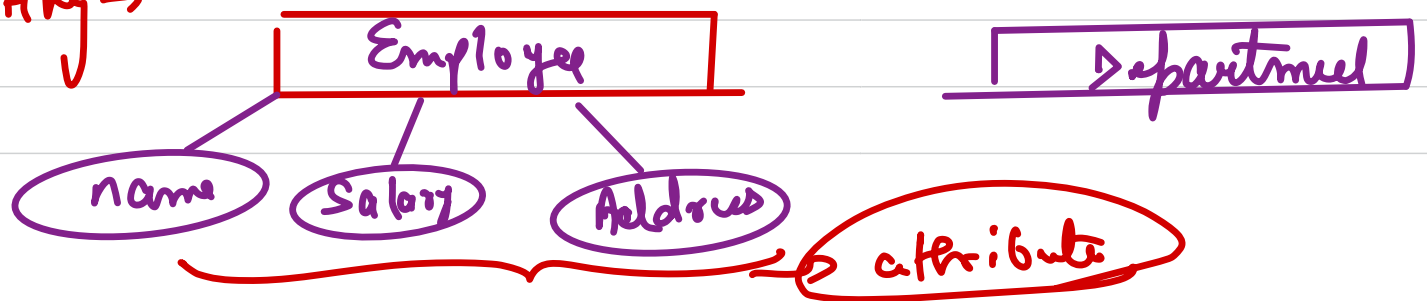
Diff → Data Model | Data Schema

E-R Model

Entity → these are real world objects

Relationship → this is how the entities are related.

Entity →



* Attributes → these are properties that define the database.

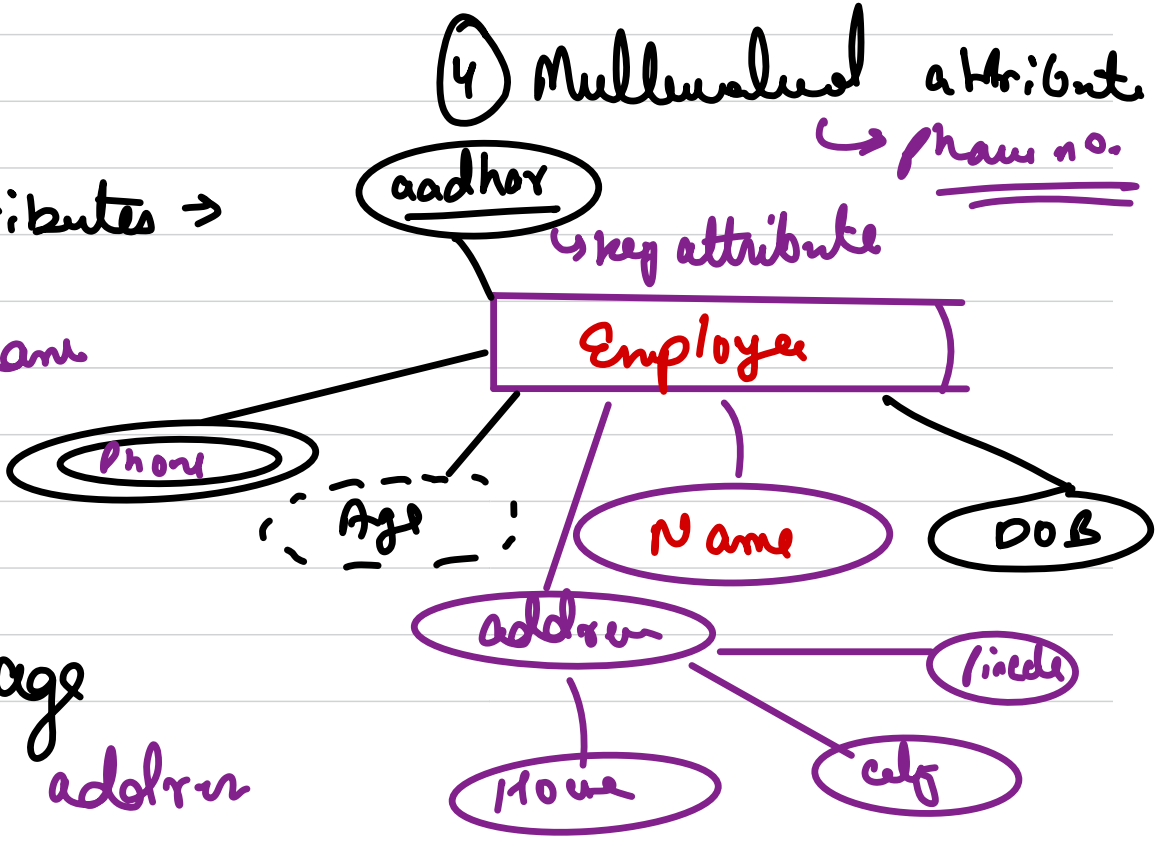
types of attributes →

(1) Simple → name

(2) Derived → age

(3) Composite → address

(4) Multivalued attribute → phone no.

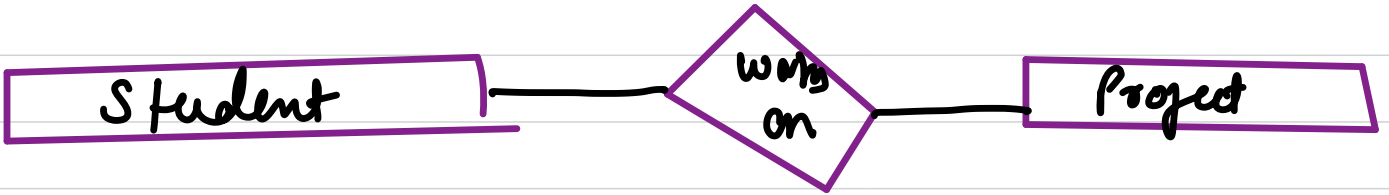


keys → In an ER diagram, an imp constraint on the entities is the KEY or uniqueness attribute. This unique attribute is also called as primary key.

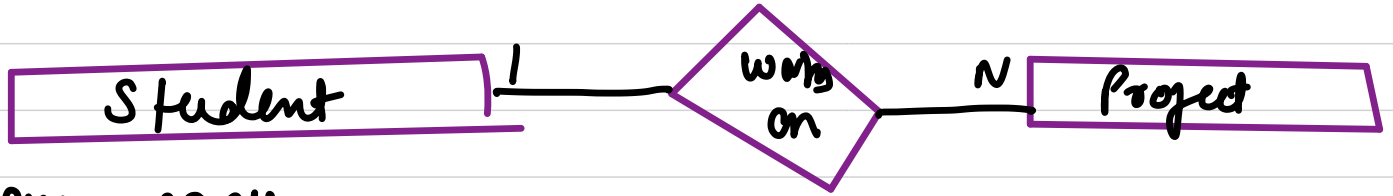
But, sometimes we need 2 or more than 2 attributes to uniquely identify a record. Then these type of attributes are called composite keys.

Relationship

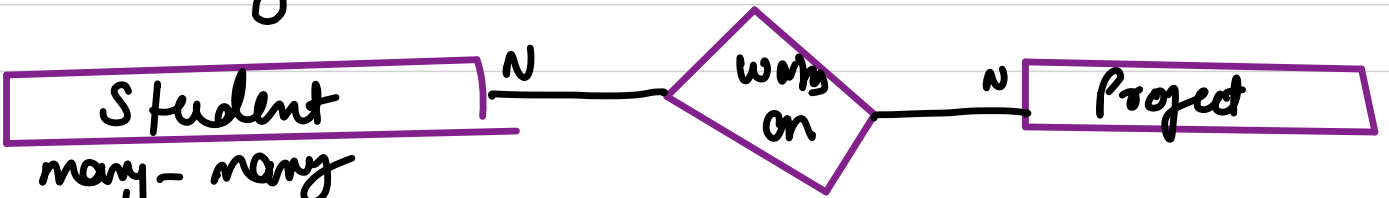
Binary relationships



One-one relation

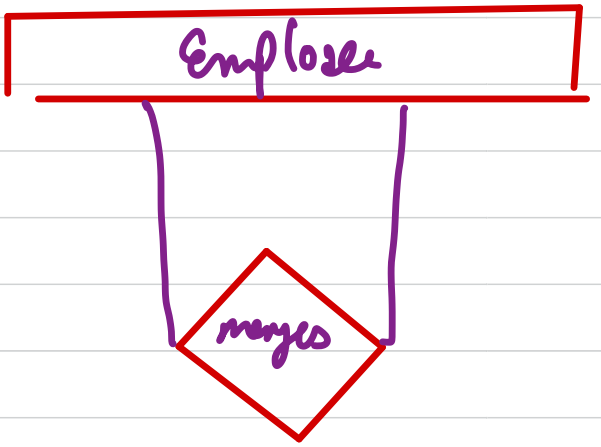


one-many

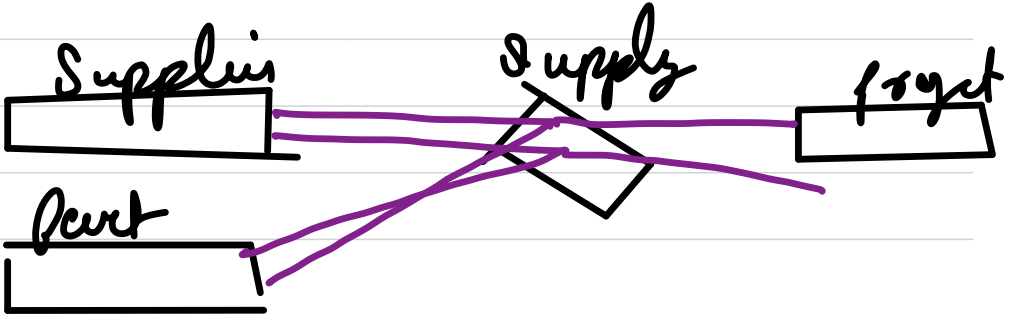


many-many

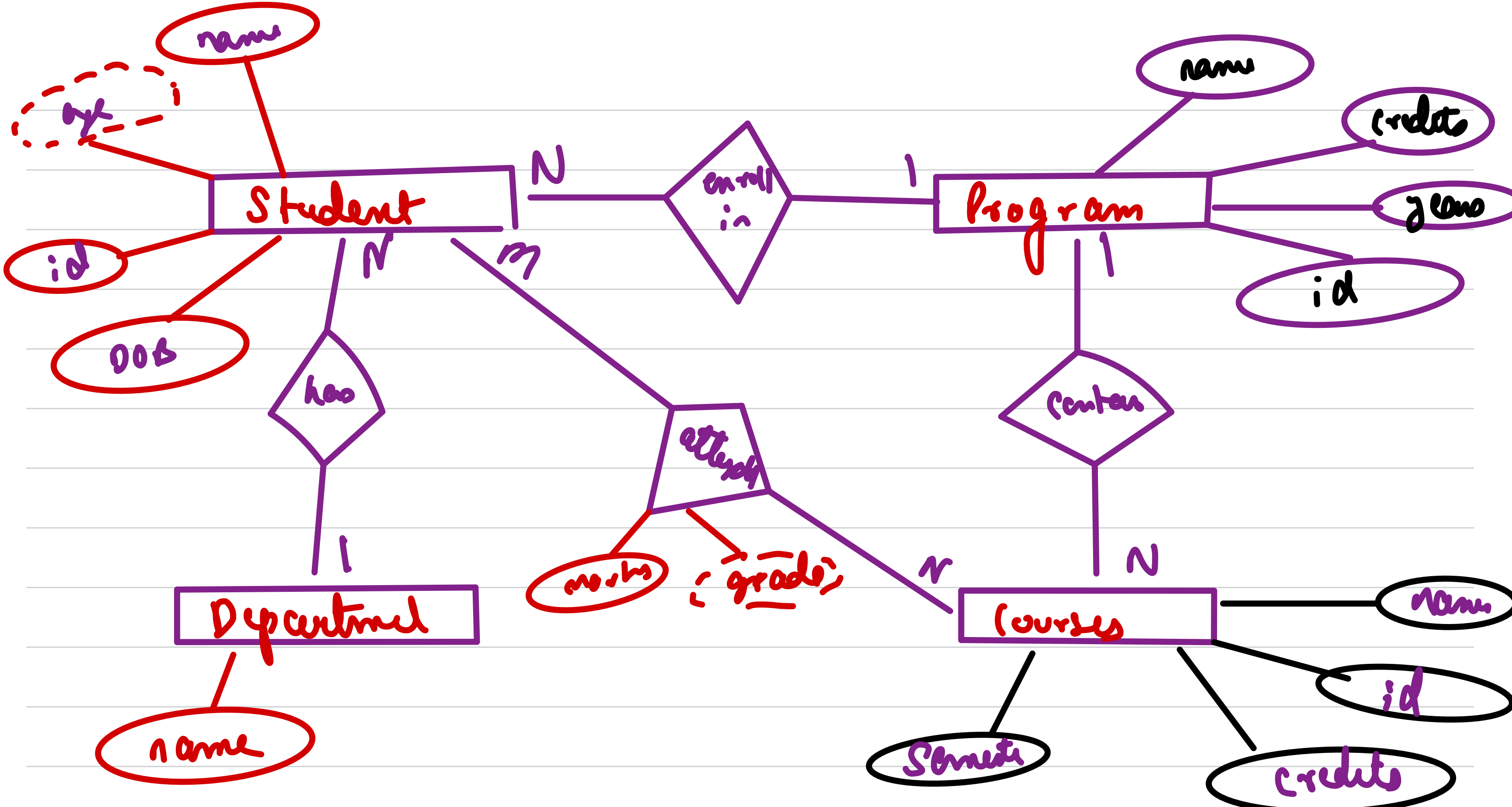
Unary Relationship



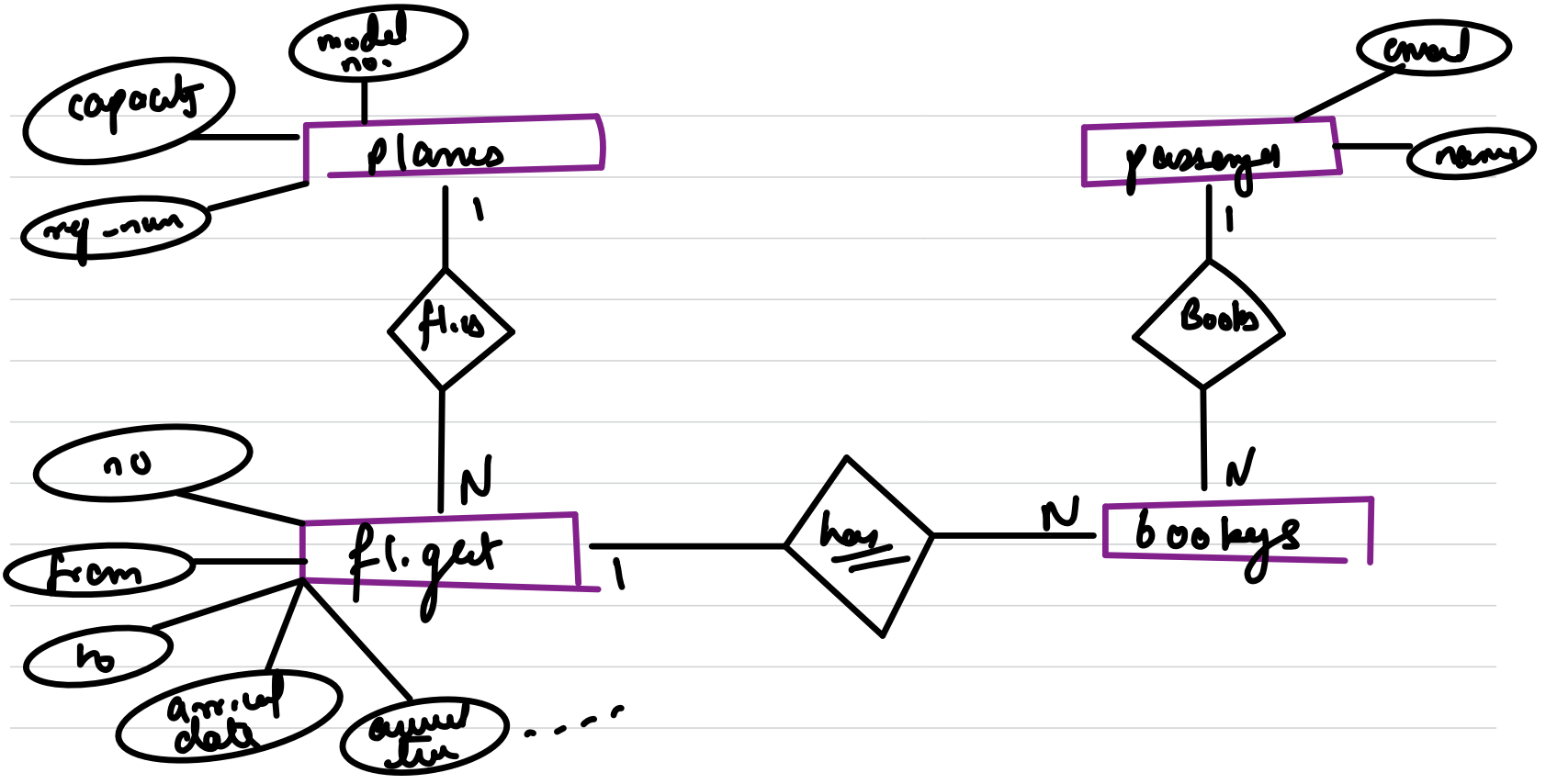
Binary Relationship



Try to create ER diagram of university db.


name
age
Student
N
enroll in
1
Program
name
credits
genre
id
id
DOB
N
M
has
1
Department
name
centers
attends
marks
{grade}
N
N
Courses
name
id
Semester
credits

Draw an ER diagram of flight database →


model no.
capacity
planes
reg_num
1
N
H.is
no
From
fl. get
to
arrival date
1
has
N
N
Books
1
N
passengers
email
name
bookings

DBMS

Relational Data Model

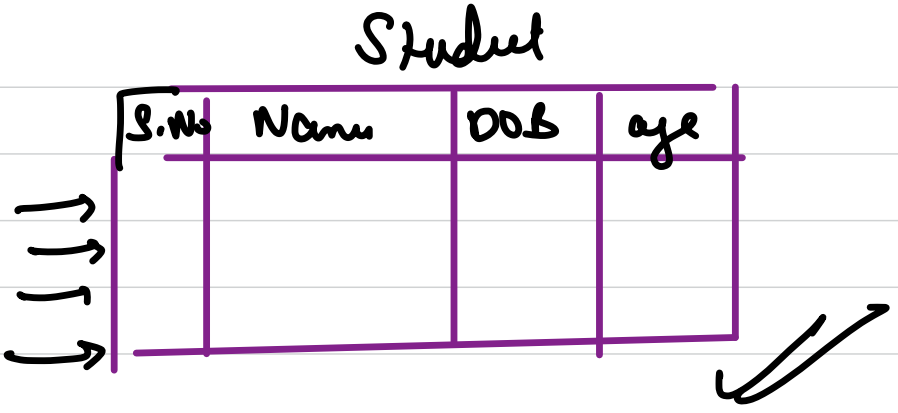
In actual entities are represented as tables & data as values (rows) and attributes of the entities are represented by columns.

RDBMS → MySql, PgSQL, Oracle db etc

columns → attributes
rows → Tuples

properties →

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



→ RDBMS mostly support schemas

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

Composite key →

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

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

alternate key → Candidate key other than primary key.

foreign key → a group of attributes that establishes link btw 2 tables.

Student

S_id	nam	phn
1	a	---
2	b	---

Cours

C-id	name	dept
1	c	1
2	d	2

Student_cours

S-id	C-id
1	1
1	2
2	1

functional dependency

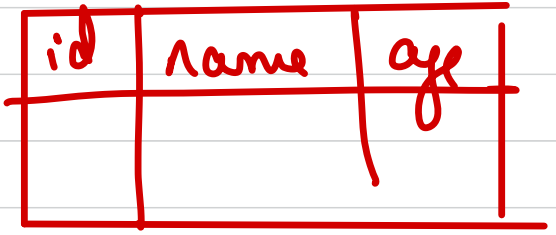
functional dependency is relationship between 2 attributes. Generally we show this relationship between primary key and other non-key attribute

for any relation R, attribute Y is functionally dependent on X if for every valid instance of X, the value of X uniquely identifies Y.

$X \longrightarrow Y$

↳ dependant

determinant



id $\longrightarrow$ age

id $\longrightarrow$ name

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

1) Axiom of reflexivity →

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

$y \subseteq x$ then $x \longrightarrow y$

address → h.no, street, state ....

h.no ⊆ address

Adress → h.no

Axiom of augmentation

It is also called as partial dependency.

if X determines Y, then X2 determines Y2

where X2 is XUZ and Y2 is YUZ

if $X \rightarrow Y$ then $X2 \rightarrow Y2$

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

(S-id, C-id) → composite key → 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 violates 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 can determine

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

$E \to A$, $E \to D$, $A \to C$, $A \to D$, $AE \to F$, $AG \to K$

Find Closure of E

$E \to A$

$E \to AD$

$E \to ADCE$

$E \to AEDCF$

$E \to D$

$A \to C$

$AE \to F$

$E \to E$

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

$B \to C$ $\quad BC \to AD$, $\quad D \to E$, $CF \to B$ · find B^+

$B^+ \to B$

$B^+ \to BC$

$B^+ \to BCAD$

$B^+ \to BCADE$

Normalization

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

- 1NF
- 2NF
- 3NF
- BCNF Boyce Codd 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 a row.

Student

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

1NF →

nam	rollno	course
John	101	DBMS
John	101	OS
Michael	102	CN
Michael	102	DSA

2NF (second normal form) → 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_fee
1	C1	1000
2	C2	1500
1	C4	2000
4	C3	1000
4	C1	1000
2	C3	3000

{S_id, C_id}

→ Partial dependency as C_id → C_fee

2NF →

Student-course

S_id	C_id
1	C_1
2	C_2
1	C_4
4	C_3
2	C_1
	C_5

Course

C_id	fee
C_1	1000
C_2	1500
C_3	1000
C_4	2000
C_5	3000

3NF (third normal form) → The relation should be in 2NF. and dont have transitive dependency:

Score

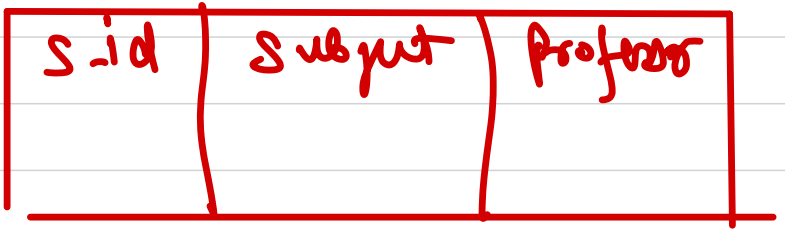
s-id	sub id	marks	exam id

Exam

id	type	total mark

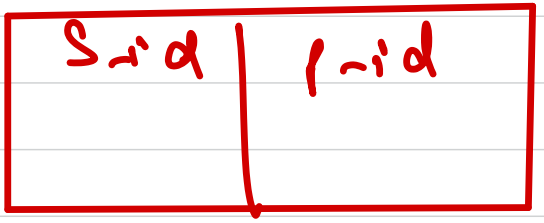
BCNF → relation should be already in 3NF.

And for any dependency A → B, A should be a Super key. (it mean A can never be a non key attributes)



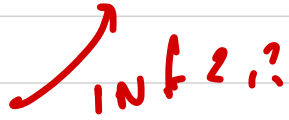
Professor → Subject

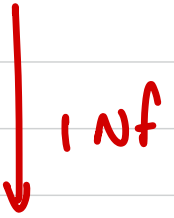
✗ ✗ ✓

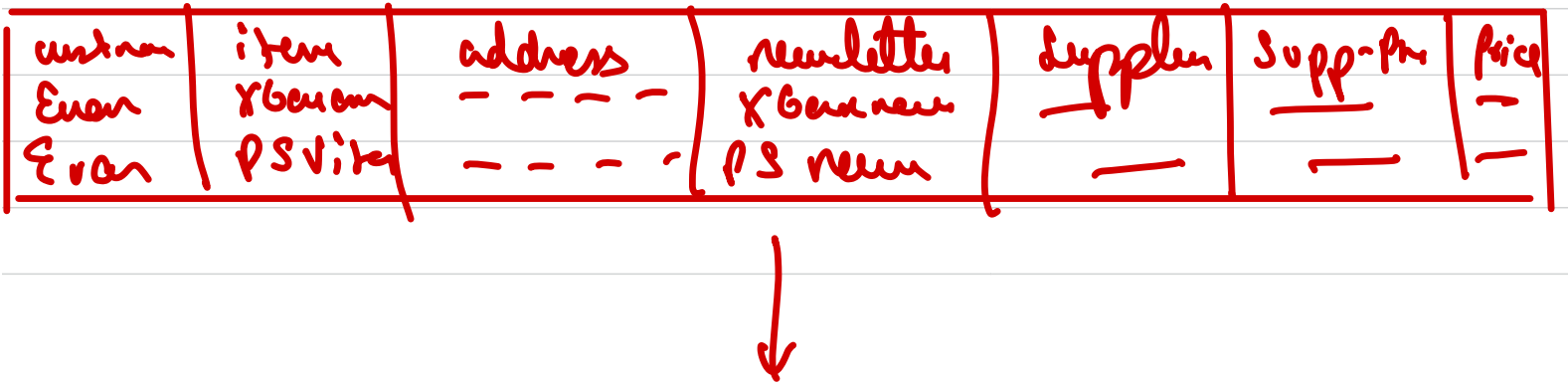

S_id
p_id

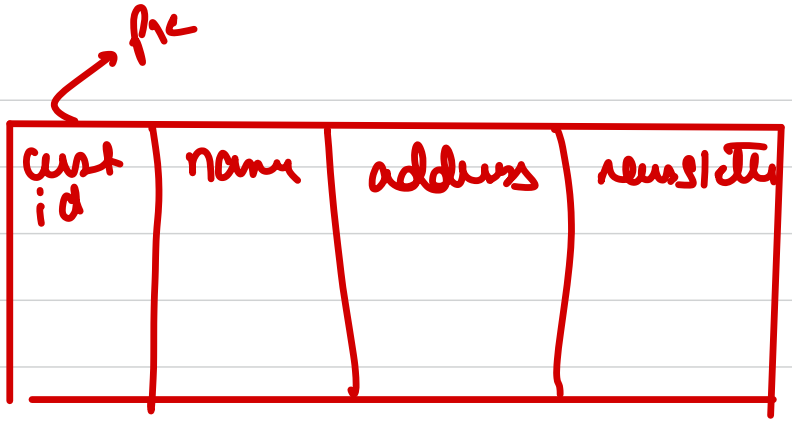

p_id
name
subject

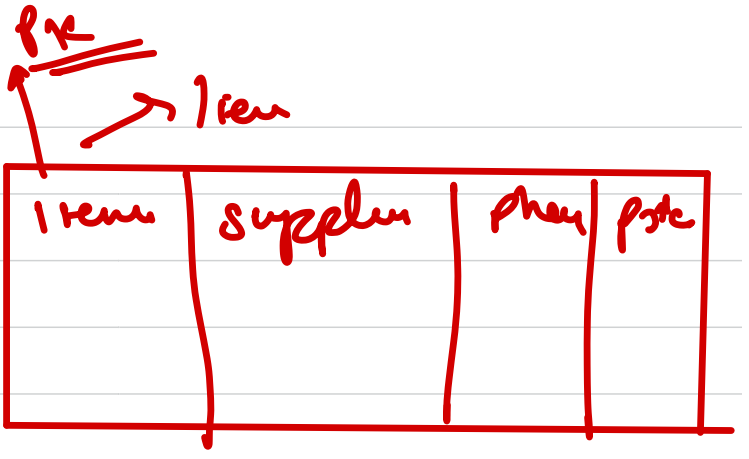
Cust Name	Item	Shipping Address	Newsletter	Supplier	Supplier Phone	Price
Alan Smith	Xbox One	35 Palm St, Miami	Xbox News	Microsoft	(800) BUY-XBOX	250
Roger Banks	PlayStation 4	47 Campus Rd, Boston	PlayStation News	Sony	(800) BUY-SONY	300
Evan Wilson	Xbox One, PS Vita	28 Rock Av, Denver	Xbox News, PlayStation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

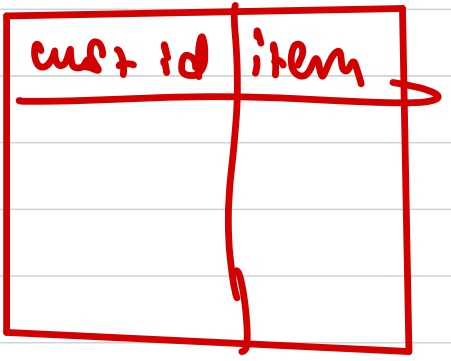






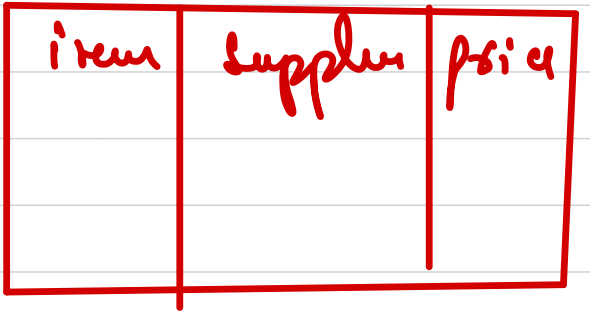


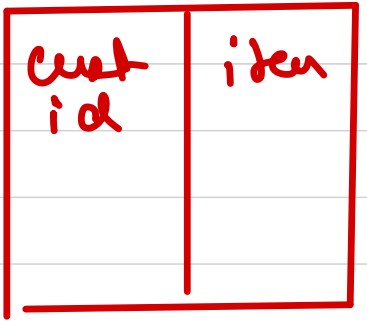




Custom_item_jc


cust id
name
address
news letter


item
Supplier
price


cust id
item


Supplier
phone

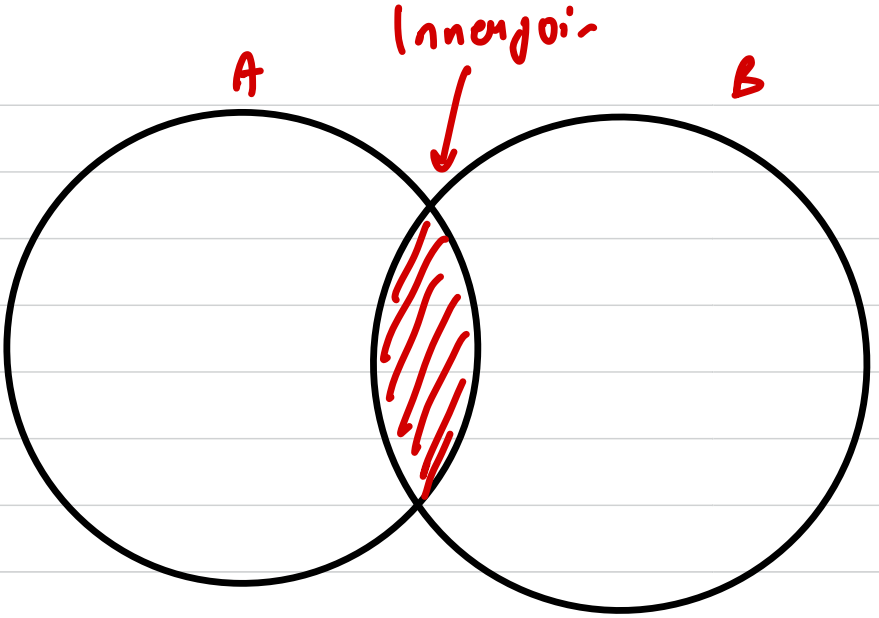
4th normal form → BCNF already &
no multi valued depends

→ create a database → Entertainment

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

Cinema → (name, id, date)

→ add some entries


Innergoi-
A
B

→ Actors

→ ActorMovies (movieId, ActorId)

→ 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

Eg → SELET * from Students WHERE id=34;

In database if we know that 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 & atleast m/2 children.

→ Each node stores a key, a node can store maximum of (m-1) keys & min of $\lceil \frac{m}{2} \rceil - 1$ keys

→ all the leaf nodes are connected as a ll.

Insertion 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

- Breake the node at $m/2^{th}$ pos
- Add $m/2^{th}$ 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 node as well.

1. If there is more than min no. of keys in the node Simply delete the key, from leaf & internal node 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 sibling.

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

ACID properties in RDBMS

A → atomicity

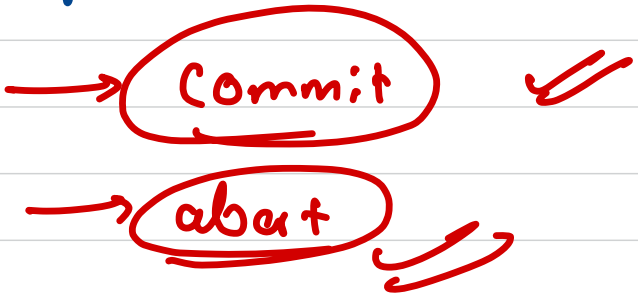
C → consistency

I → isolation

D → durability

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

atomicity 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 data even in cases of failure.

Isolation → RD BMS should manage concurrent transactions

such that no inconsistency of db occurs.

T_1	T_2
Read(x)	READ(x)
x = x + 10	READ(y)
Write(x)	z = x + y
Read(y)	Write(z)
y = y - 5	
write(y)	

Operating System

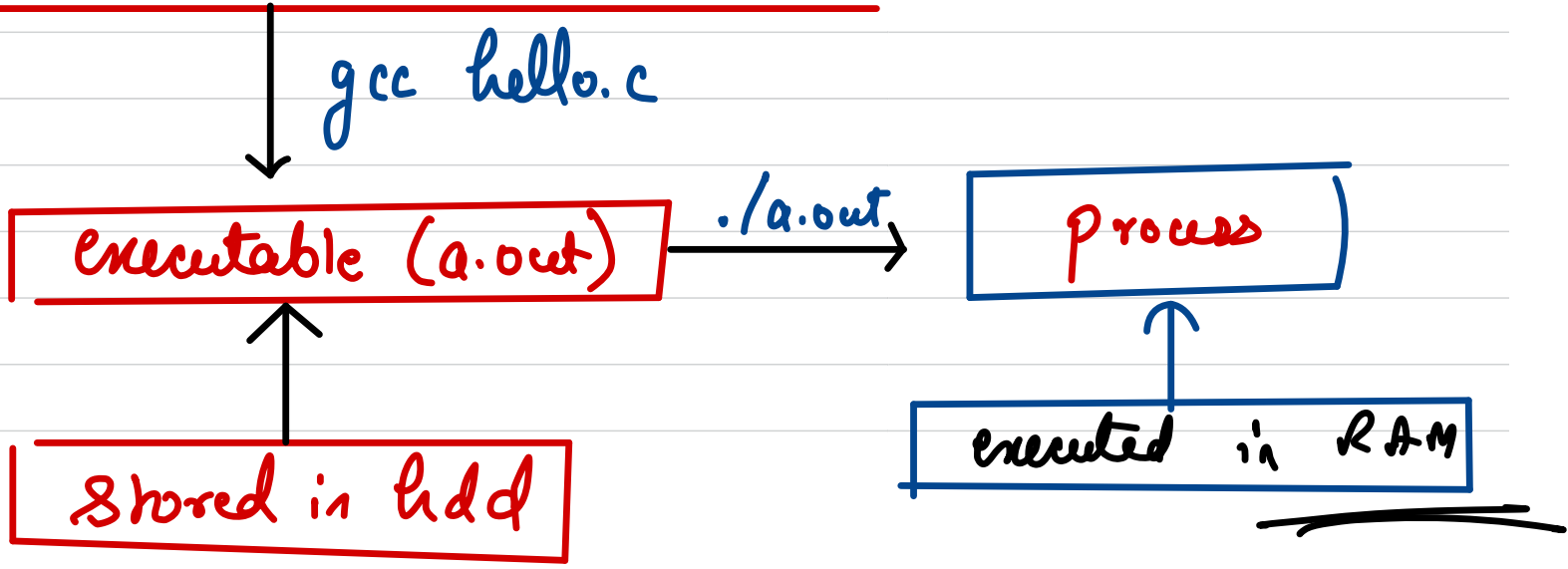
* Memory Management →

Let's say we've a C program

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

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

→ what happens when we compile and run a C program.



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 data etc

all of this is managed by OS

What is an operating system? → software

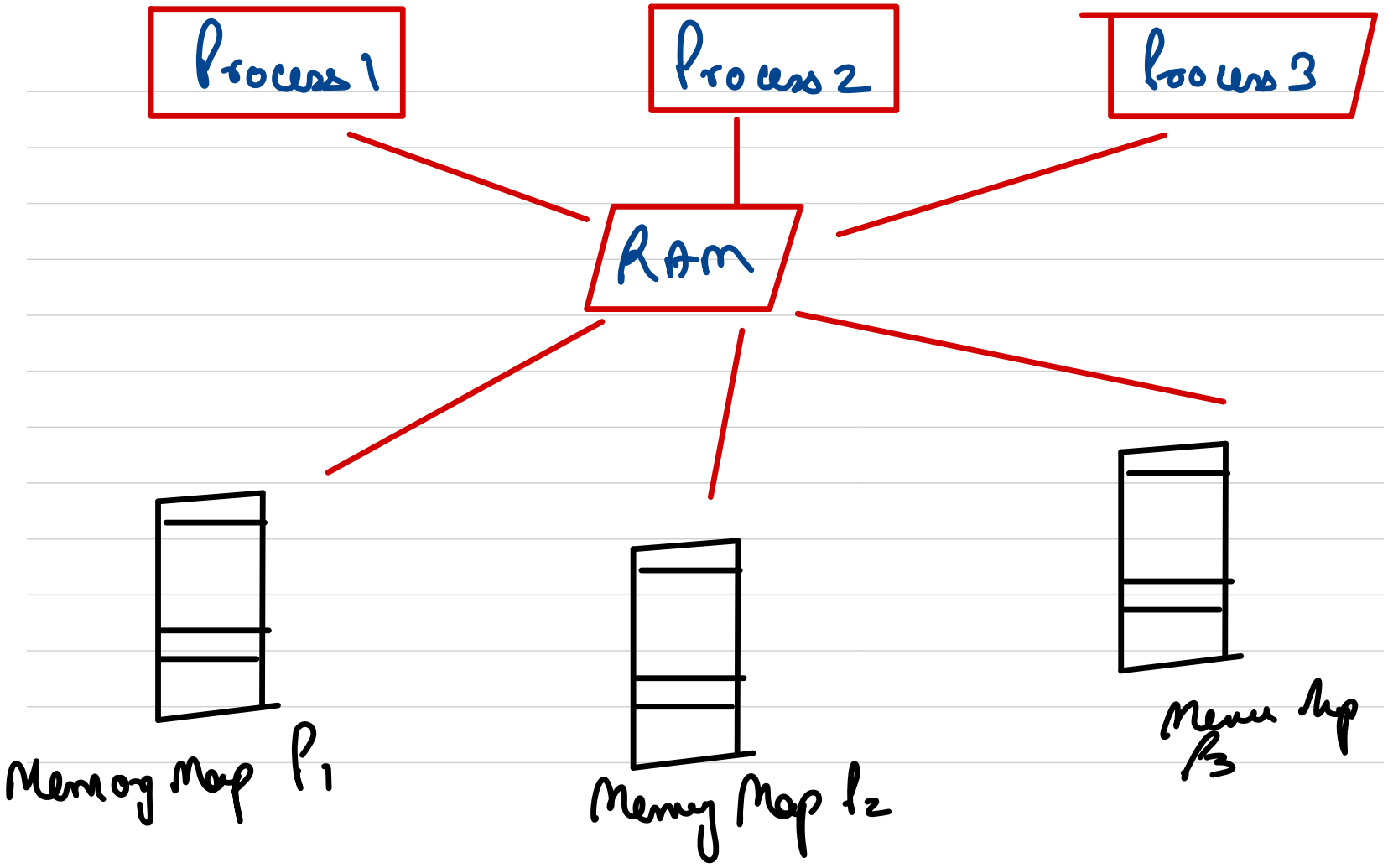
It is an intermediary between end user and computer hardware. It provides suitable environment to create and execute programs.

→ Resource Management
→ Process Management
→ memory & Storage Management
→ Security

How RAM manages process?

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

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


Process 1
Process 2
Process 3
RAM
Memory Map P1
Memory Map P2
Memory Map P3

Let's see how RAM manages multiple process -

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

→ limitation

→ multiple process management is not very good.

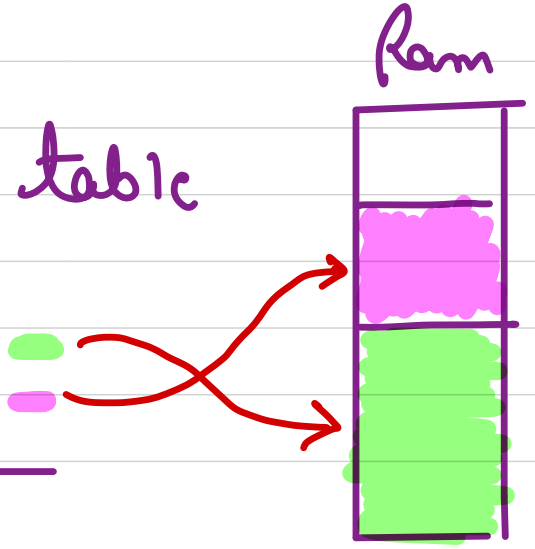
(VVI) → process memory size is restricted by RAM size..

② Partition Model

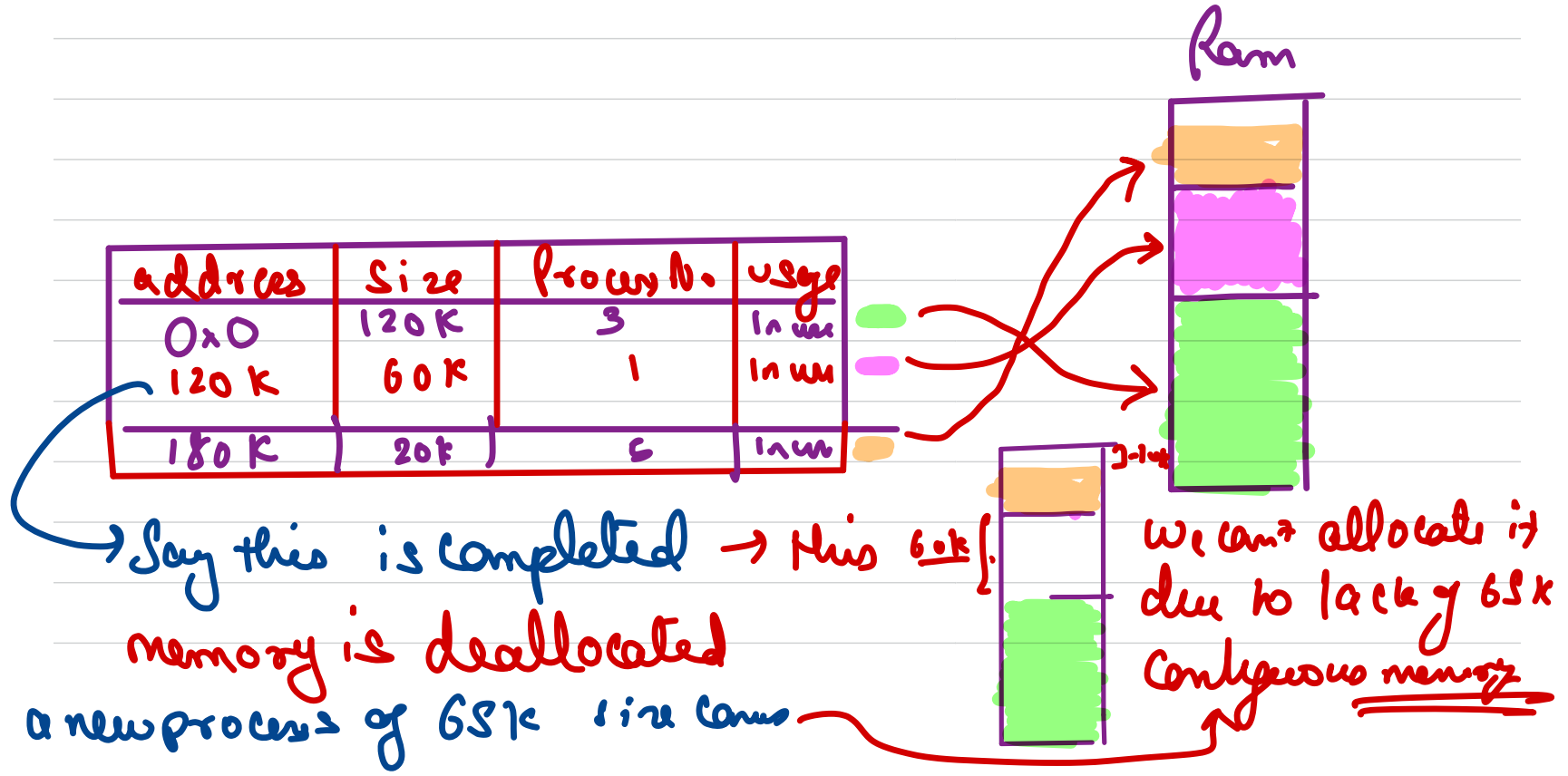
as long as sufficient contiguous space is available new process are allocated in the memory:

It maintains a partition table

address	Size	Process No.	usage
0x0	120K	3	In use
120K	60K	1	In use



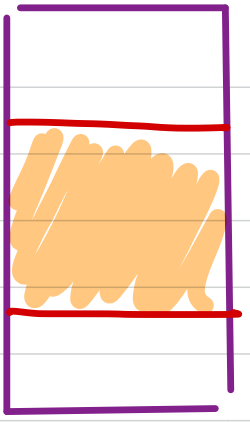
Let's say we get a new process of size 20k



This leads to a phenomenon called as fragmentation

fragmentation is a situation when free block are too small to satisfy an memory request. The small left over memory is getting wasted and if this small chunks inc, then amount of reusable memory decreases.

* Internal
* External



How partition model handles fragmentation ??

① First fit → It may make fragmentation worse.

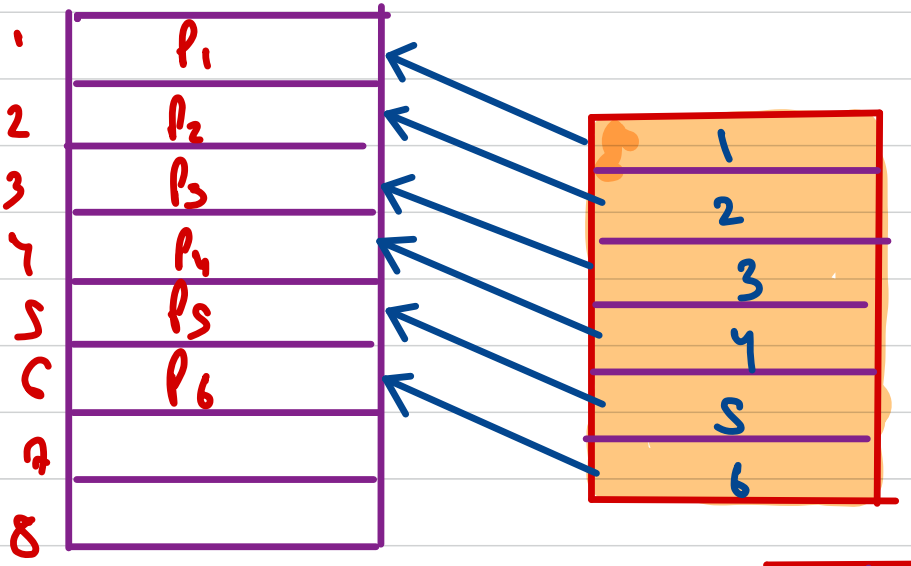
② Best fit → this may affect performance.

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 in abl processors it was typically about 4KB.



RAM

Process also splits into blocks of equal size when Block :: page frame size

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

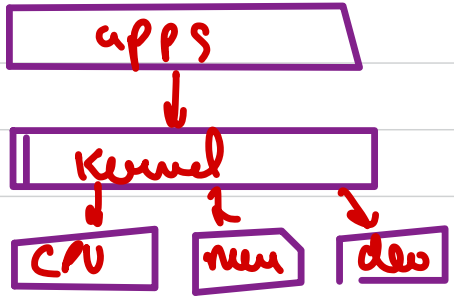
Page table

Then process block is allocated to frames

Because of per process page table, blocks of process 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 partly a cache called as TLB (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. using 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 process.

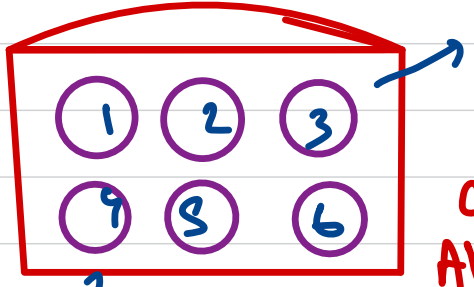
Q) Do we really need to load all blocks into memory before process starts executing ??

No

Not all parts of the program are accessed simultaneously. Infact 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, a space is allocated (swap space) All blocks of mem is present On demand blocks will be loaded in the ram.

block	frame	present bit
1	14	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 block 3 & 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, that 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??

Page replacement policies

Note → Page table's for the processes under consideration are updated during page replacement

Few policies are-

1. First In First Out
2. Least recently used
3. Least frequently used

Based on the policy implemented in the OS, the OS swaps out a block from memory with a new block to be added in RAM

Process of loading a block in RAM is called Swap In

Process of pulling out a block from RAM is called Swap Out

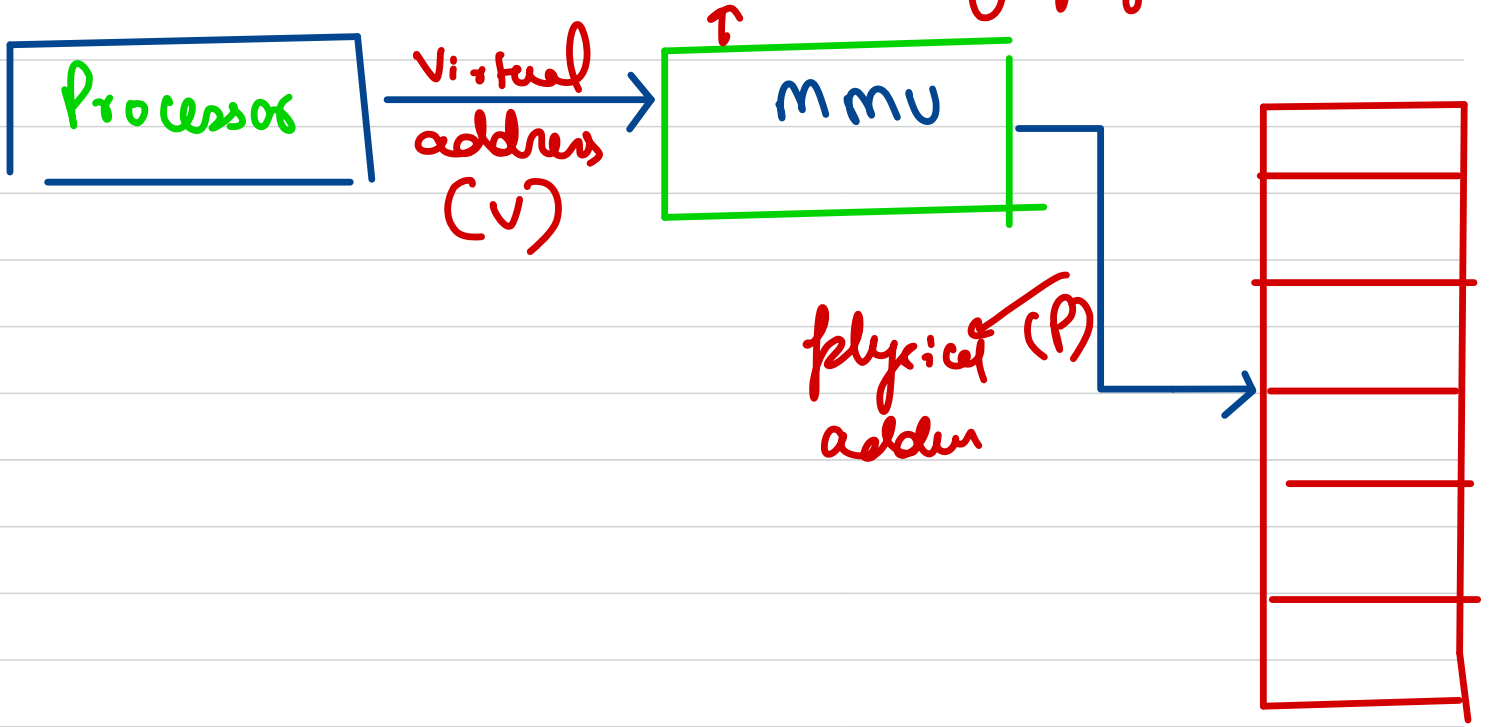
* Swap out process → During the swap out process the changes in the content of block to be swapped from RAM is copied to the disk so the disk has latest piece of changes. But if no changes has been done then no need to do a copy. To maintain this a dirty bit is used.

Virtual address space of a process.

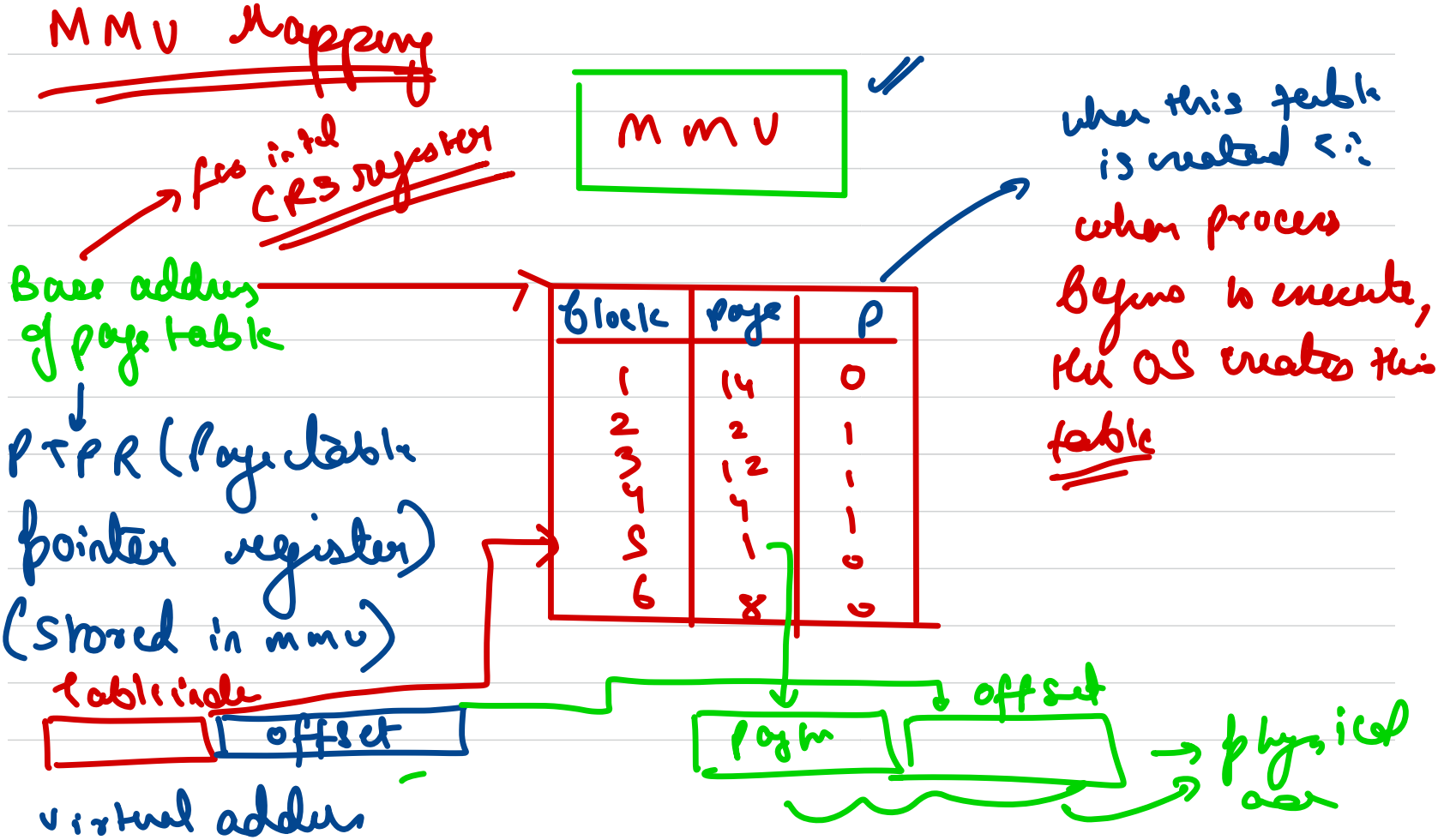
↳ is not the actual physical address

To access process memory the processor generates a corresponding virtual address.

mmu maps virtual address to the corresponding physical addres



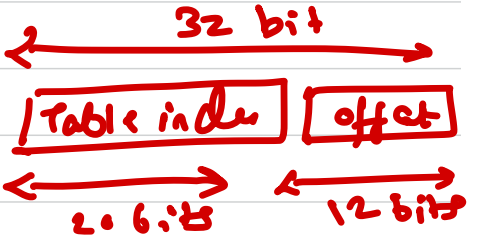
MMU Mapping


MMU
Base address of page table
PTPR (Page table pointer register) (stored in mmu)
fed in the CR3 register
when this table is created ?:
when process begins to execute, the OS creates this table
Block
Page
P
1 14 0
2 2 1
3 12 1
4 4 1
5 1 0
6 8 0
Table index
offset
virtual address
page
offset
physical

MMU mapping for 32 bit System

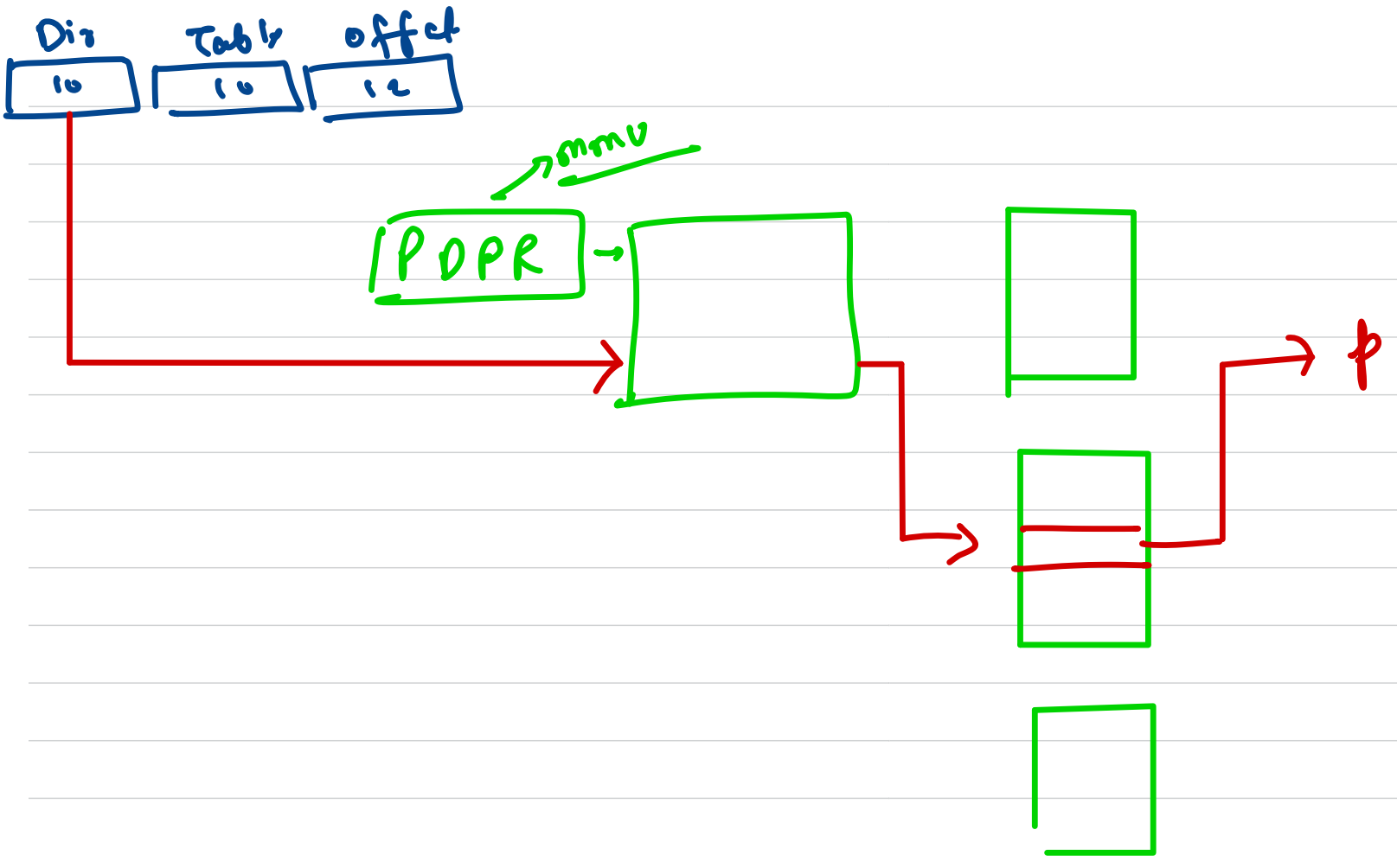
V address → 32 bit

Process size → $2^{32} \approx$ 4gb



if table index is of 20 bits then page table can have 2^{20} entries ≈ 4Mb (& this 4mb is reqd in contiguous order)

Enter 2 level page translation

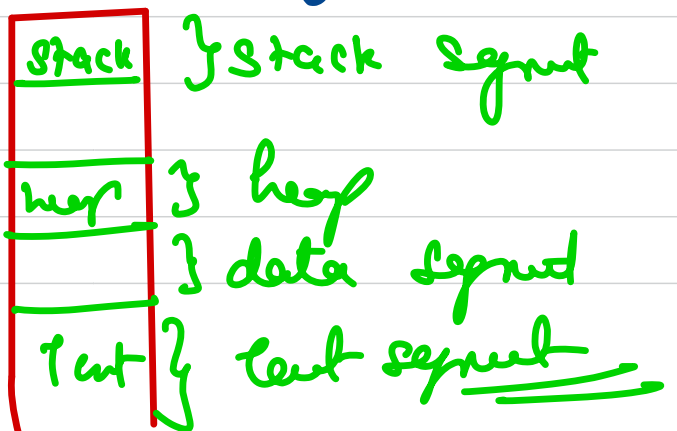

Dir
Table
offset
10
10
12
mmu
PDPR
p

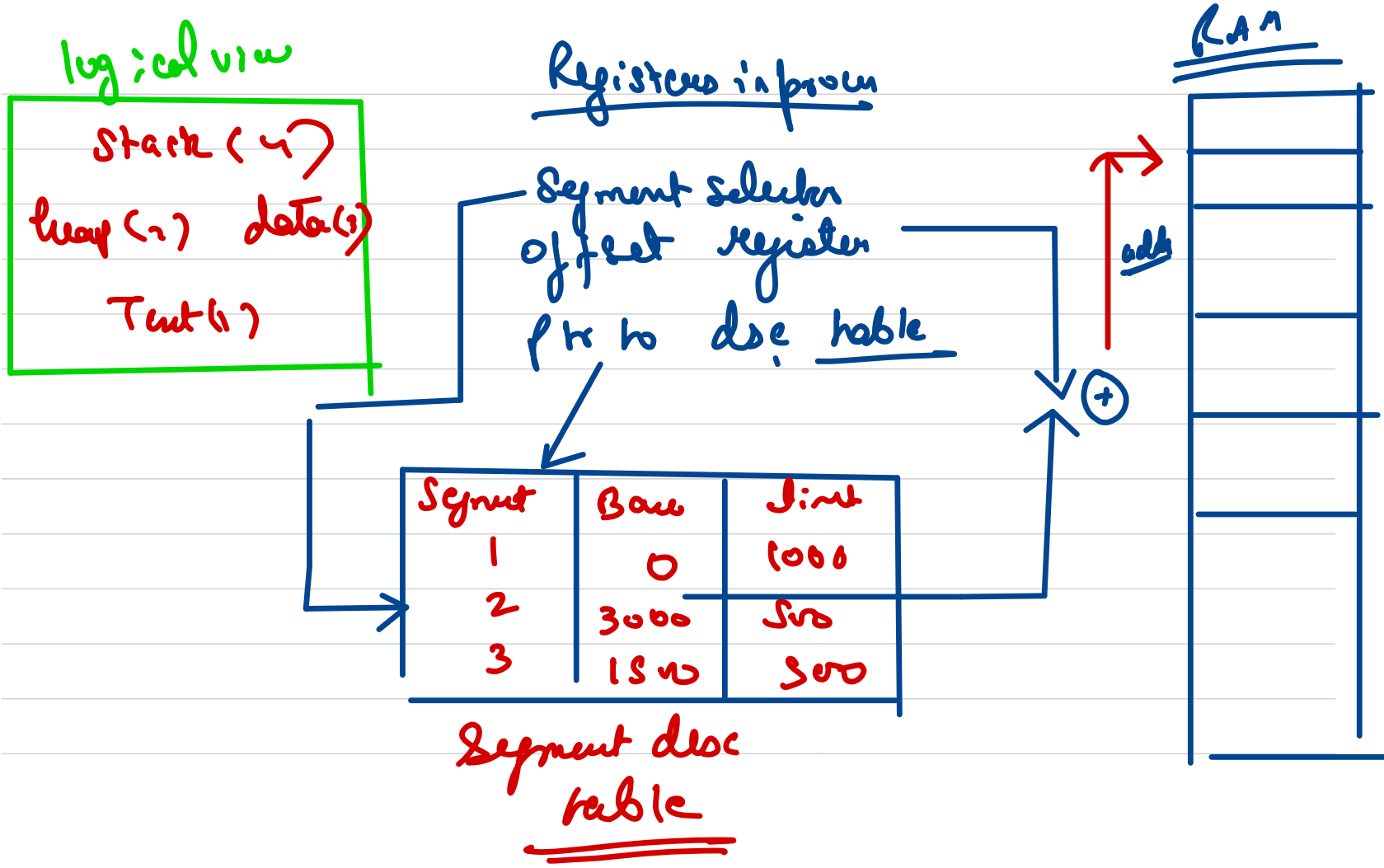
Segmentation

Programs are collection of logical modules.

logical modules → global data, func°, class, namespace

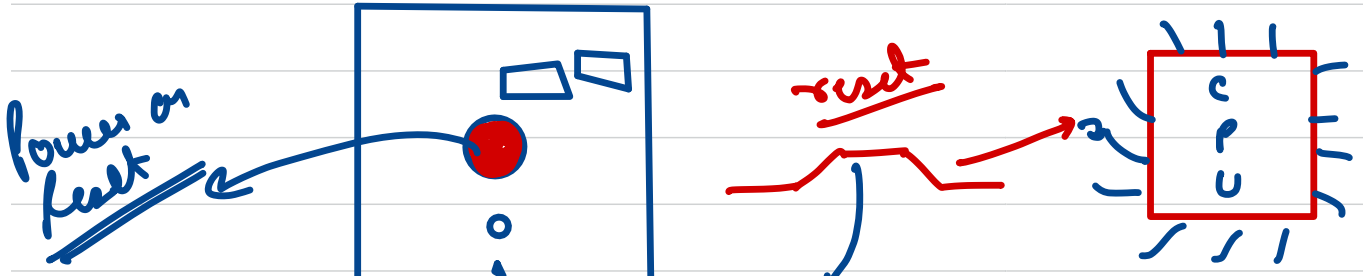
Segmentation splits programs into segment that are more logical




logical view
Stack (4)
heap (2) data (3)
Text (1)
Registers in proc
Segment selector
offset register
ptr to desc table
RAM
addr
+
Segment | Base | Limit
1 | 0 | 1000
2 | 3000 | 500
3 | 1500 | 300
Segment desc table

Q. what happens when the PC boots ??

Intel processors maintain good backward compatibility.



→ this is sent to a special pin of CPU called as reset pin

The moment CPU receives the signal, CPU starts the process of turning the computer on (booting)

What happens during the booting step??

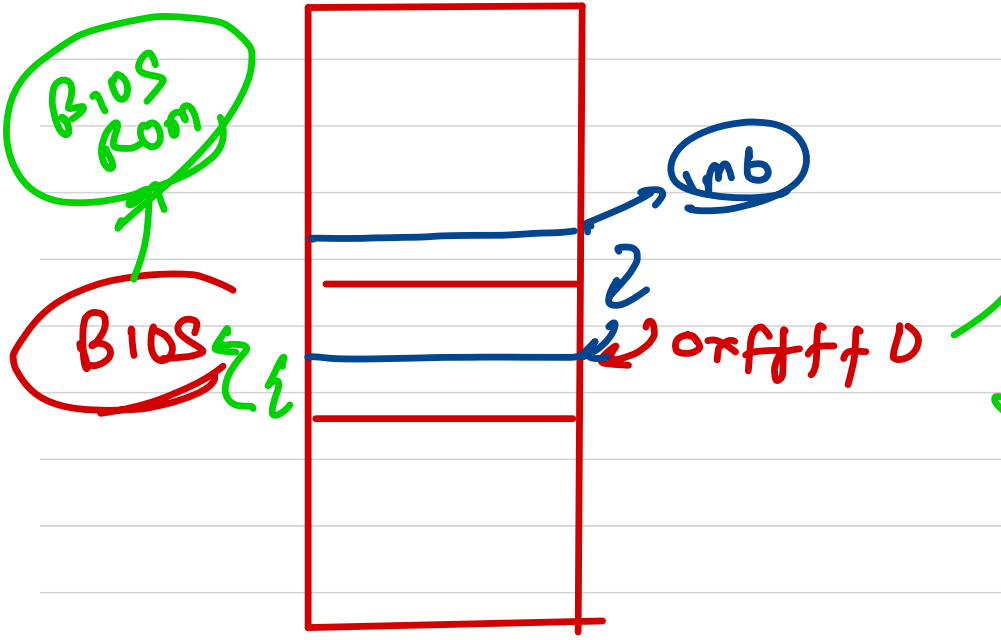
after power on reset, all the registers of the CPU are initialized with 0. except

CS → 0xf000

IP → 0xfff0

Calc the address of the first instruction to be execute

→ (CS << 4) + IP → 0xffffff0



What instruction is present here in

↳ this instruction makes us jump to the executable start of bios

Processor comes in REAL MODE (Backward compatible)

BIOS → Basic Input Output System

↳ It is a small chip connected to processor (ROM/flash)

What Bios do ??

1) Power on self test
2) Initialize the video card & other devices
3) Display BIOS Screen
4) Perform memory test
5) Configure plug & play devices
6) Identify the boot device, read the sector 0 & copy it in 0x7C00, & jumps to 0x7C00

Sector 0 in the disk is called Master boot record.

↳ Total 512 bytes

↳ 446 bytes Bootable code

↳ 64 bytes is disk partition info

↳ 2 bytes Signature

Bootloader

↓

OS loads

What happens inside bootloader ??

May give option to the user to select the OS to load (windows / linux)

→ Disable interrupts

→ Real Mode → Protected Mode ↗ we unlock the accessible memory beyond 1 MB.

→ Load the OS from disk

What happens when we power up OS??

Setup the virtual memory

Initiate
- to mouse, timer, HDD, FS etc

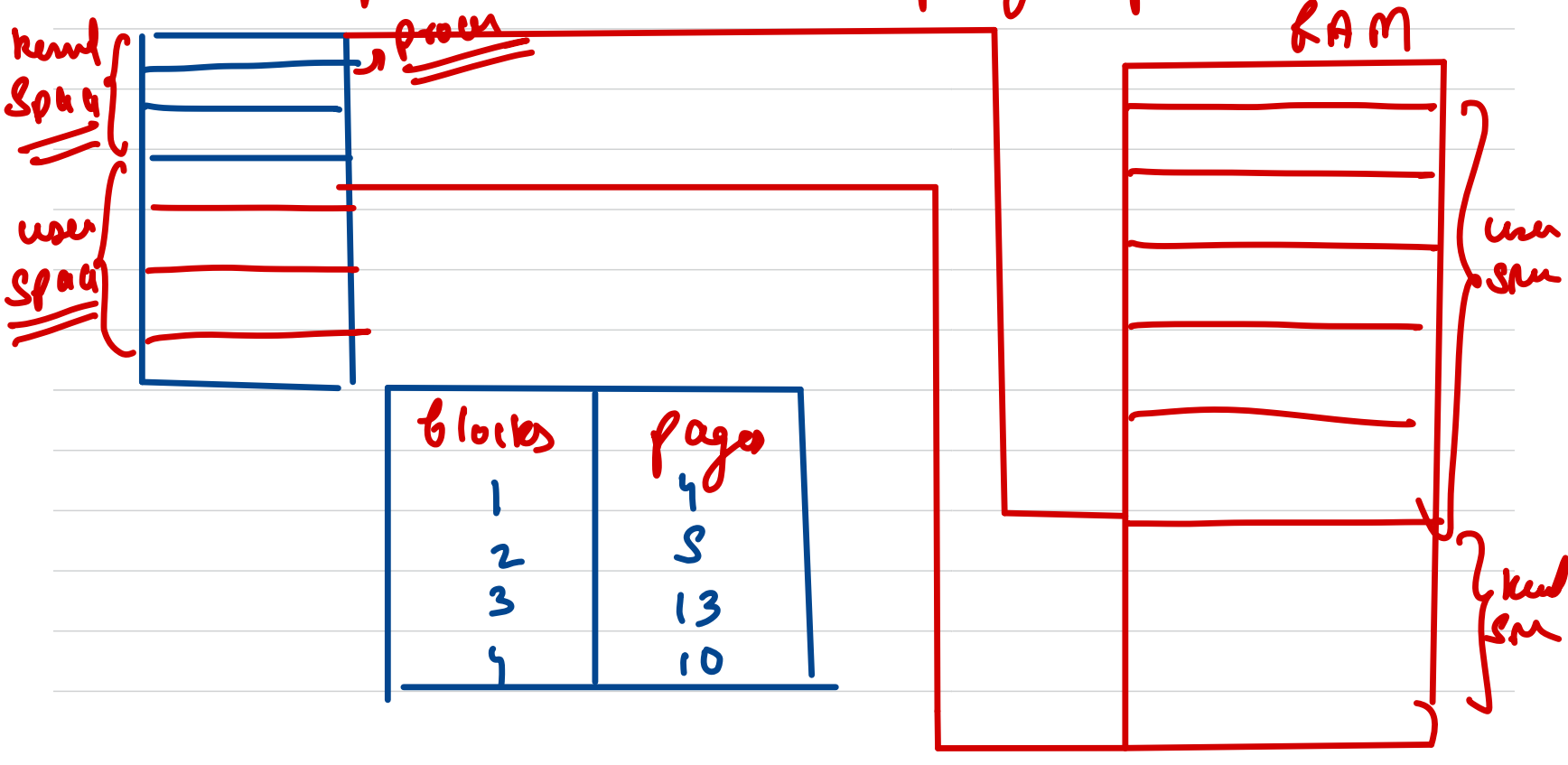
Starts user process

For multiprocessor booting, one processor is designated as "Boot Processor" (this info is stored in BIOS) all other processors are called app processors.

BIOS → Boot processor

↳ triggers booting of other app processors

How is process created step by step ??



Kernel data about a process

corresponding to each process, the kernel keeps some metadata

- PCB (process control block)
- kernel stack for each process to store context
- page tables for user process

Process stacks

Every process has got two stacks:

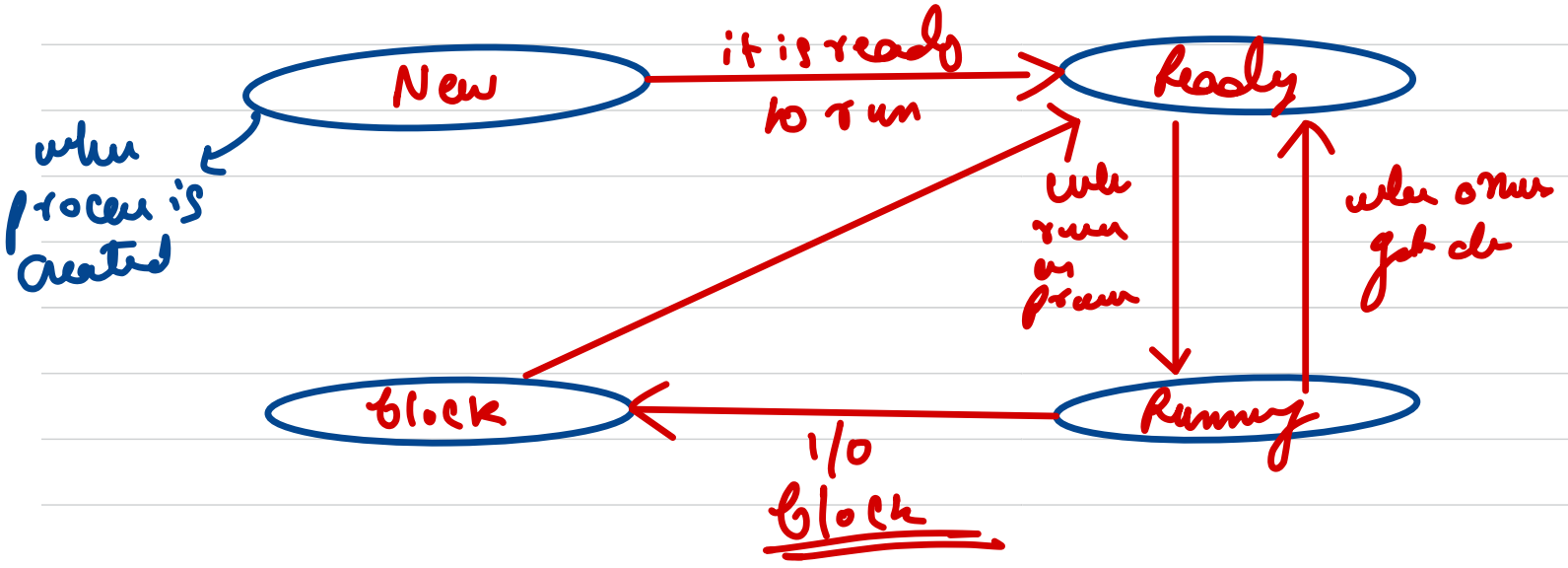
1. User stack → virtual call stack → what executing user code.
2. kernel stack
   ↳ Stores the content of a process

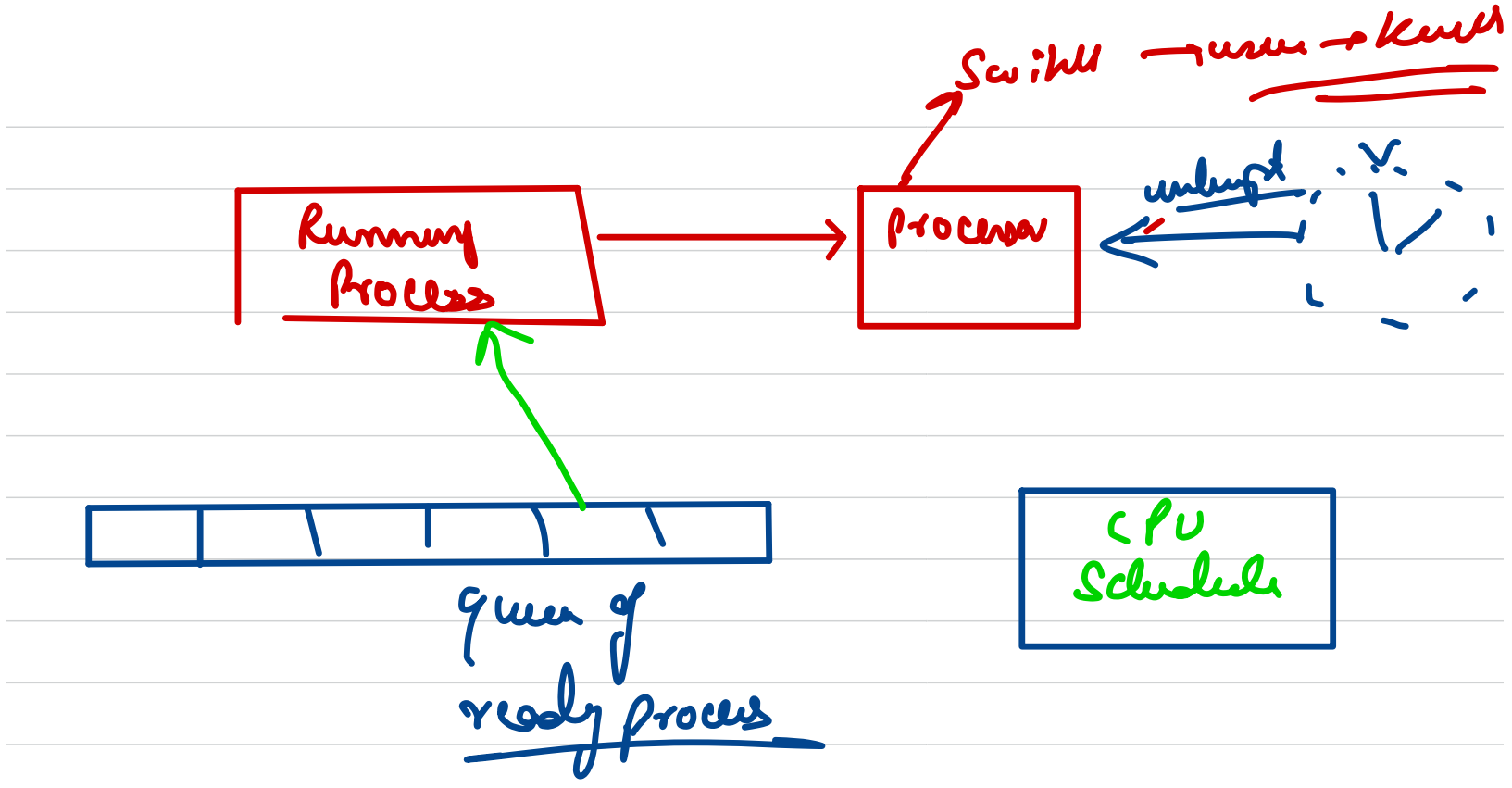
PCB

Size of process memory
List of files opened
current working directory
Kernel stack pointer
process id
page directory
...

pid → unique process identifier
generally sequential

States of a process




Switch → user → kernel
Running Process
Processor
interrupt
CPU Scheduler
Queue of ready process

cpu Scheduler triggered to run when time / io / ome interrupt occurs or when runny process is blocked on: I/O

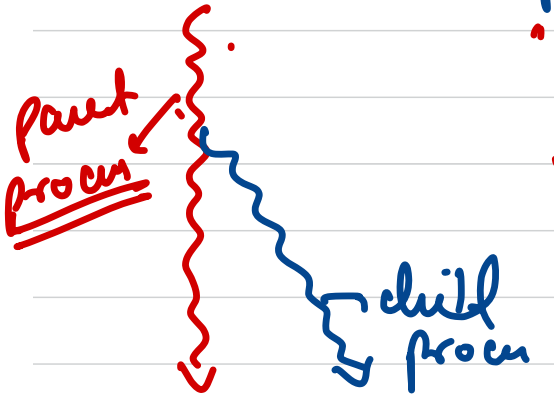
Scheduler picks another process from ready queue.

Performs a context switch.

How actually process is created.

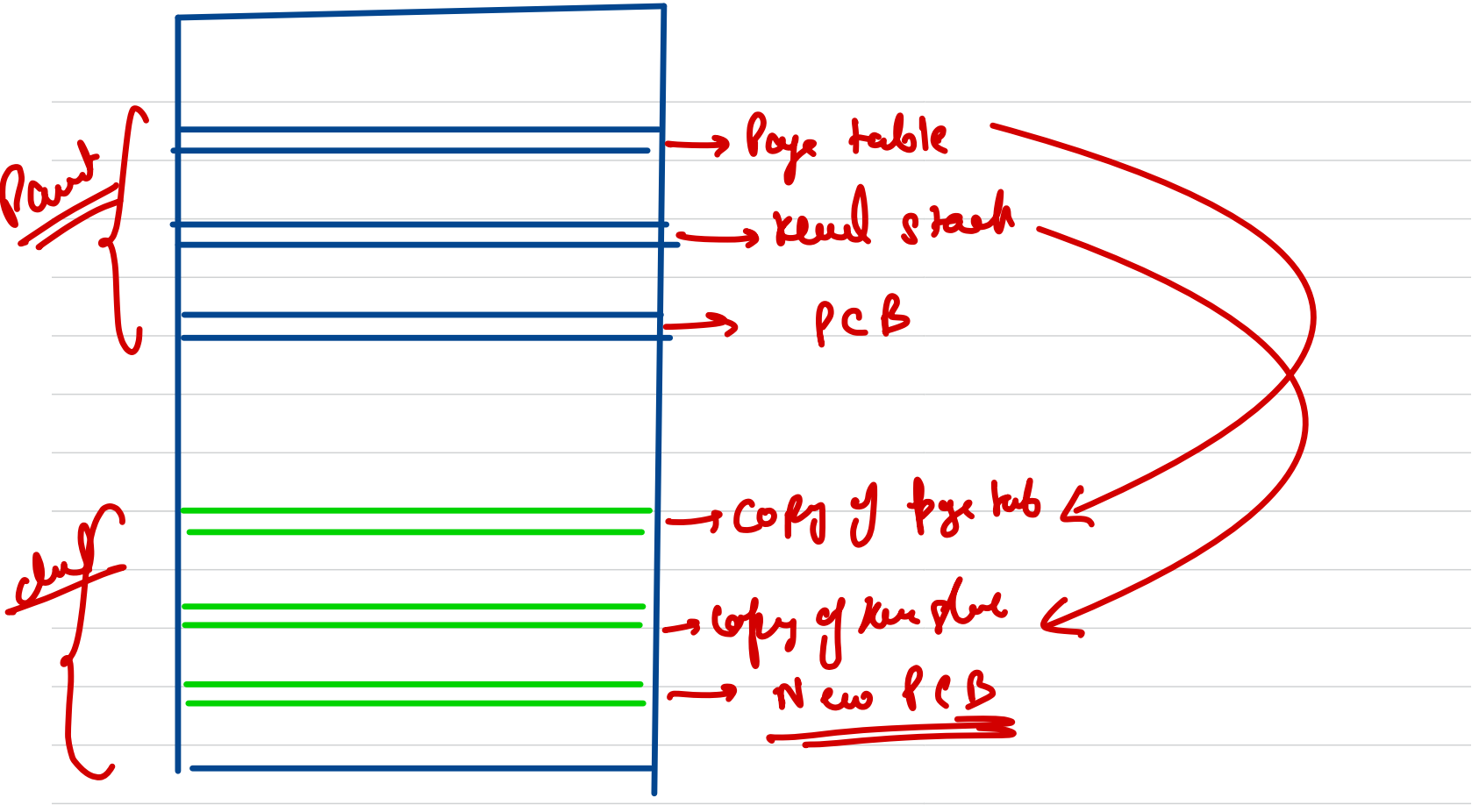
1) Cloning → child process is exact replica of parent process.

"fork" System call is executed



Note: → For accessing any kernel feature or resource, a OS process has to make a System call.


Process 1
system call
fork
Kernel
(wants fork)
child
Process 2


Parent
child
Page table
kernel stack
PCB
copy of page tab
copy of kernel
New PCB

→ find an unused PID

→ set the state of child process to new.

→ set pointers in the newly fork

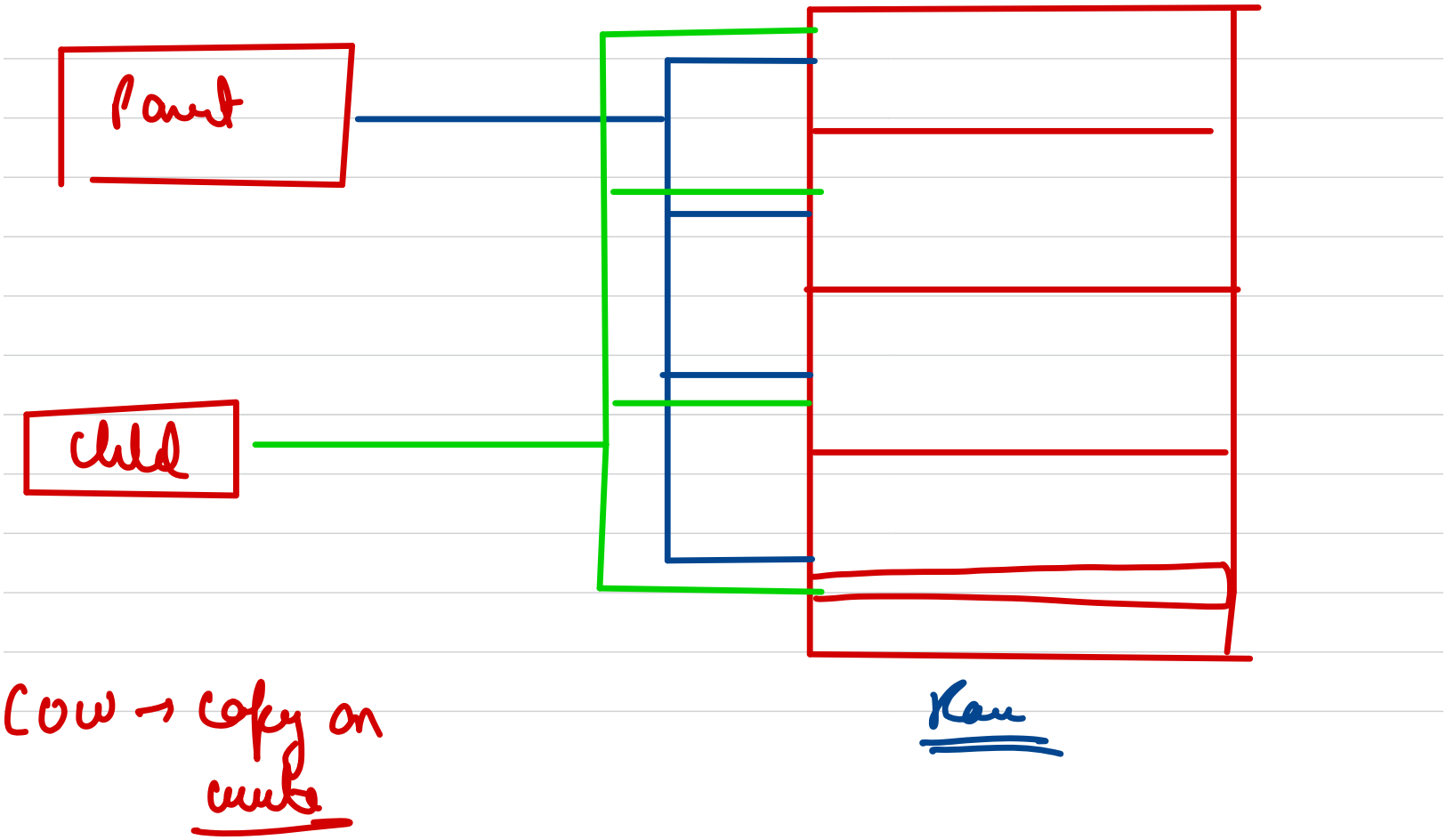
→ pagetable

→ mm_struct

→ Trapframe, set callee

→ copy info like env, files opened, etc from parent

→ set state to Ready before return

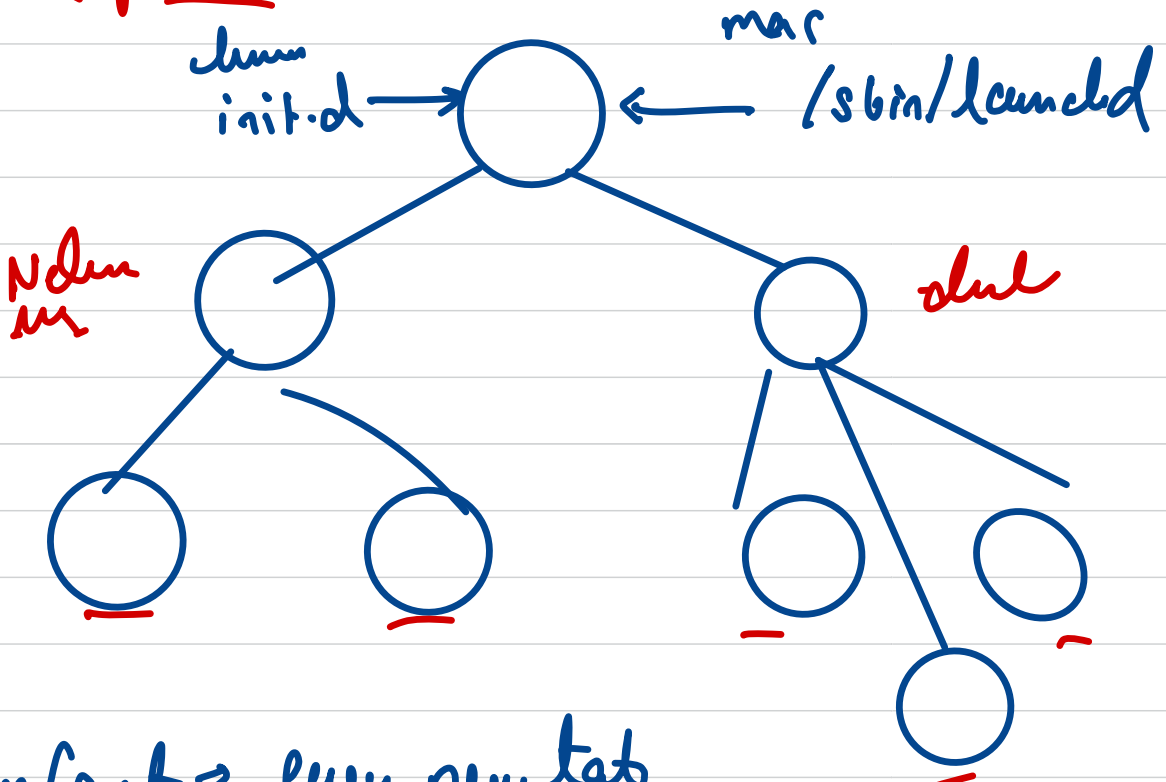

Parent
child
COW → copy on write
Ram

exec system call → It is used to start execution of a process.

→ find or add the location of the executable

→ load on demand the page required to execute it.

The process tree



#funfact → every new tab in chrome is a new process

Exit System call :→

called generally inside a child process for voluntary termination

Terminates the process

The return status is 0 which returned to parent

Kill system call →

this is a signal sent by other process or OS.

It also terminates the process.

What are zombie process ??

the moment a process is terminated at that moment it is not completely wiped from computer. So this is called zombie process

- PCB in OS still exists even though program is no longer executing.
  This is done so that parent process can read status of the killed child

when parent successfully reads the state the zombie is removed from OS using a process called as Reaper.

Orphan process

When parent process terminates before it's child is adopted by root.

There are 2 type of orphan -

1. Unintentional orphan → when parent crashes.
2. Intentional orphan → Background process detach themselves from parent.

Wait System call.

↳ this is enabled in kernel.

Interrupt

OS and events → OS is event driven → it will come into the picture when an interrupt occurs.

Events / Interrupt:

→ (1) hardware interrupt

→ (2) Traps (Software interrupt)

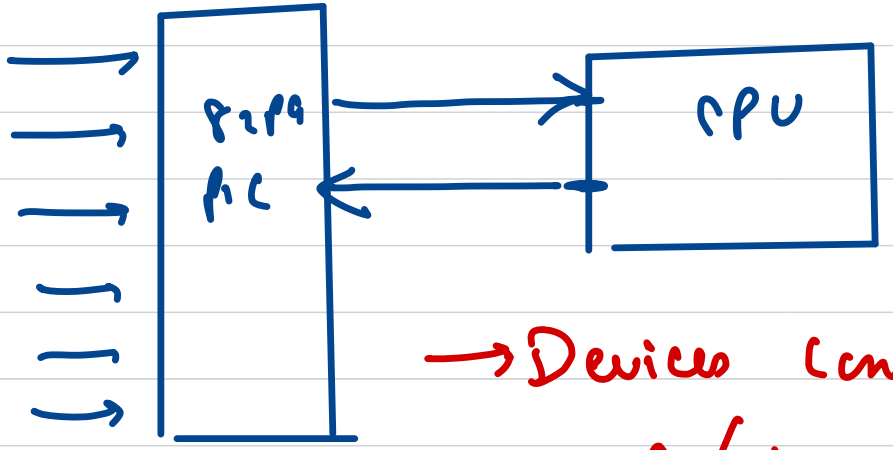
→ (3) Exceptions → faults → recoverable errors
→ aborts → difficult to recover

hardware interrupt

Now a days every processor has a dedicated pin called as interrupt pin. Any hardware like keyboard, will be connected to this pin & whenever you press a key an interrupt occurs.

as soon as, the interrupt occurs the processor executes the interrupt routine handler. (ISR).

PIC (Programmable Interrupt Controller)



It relays upto 8 interrupt to CPU

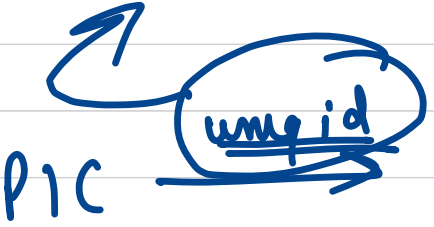
→ Devices connected raise a IRQ (interrupt req)

→ CPU acknowledges & requests 8259 to determine which device created interrupt

→ We can execute interrupts based on priority

Q. How to decide what Interrupt routin handler we need to execute for a hardware interrupt ??

→ IDTR (Interrupt Descriptor table)



CPU context Switching

Let's consider we have a Single CPU in the system which is shared among multiple processes. So, this processor does not executes every thing parallel. OS by a feature called as Multi tasking enables that this CPU is fairly shared among processes.

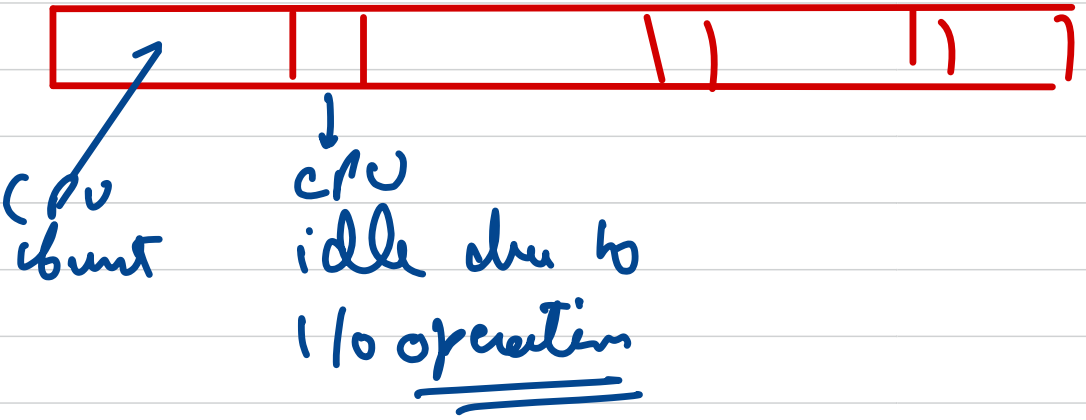
So, in a multi tasking env, or in a multi tasking enabled OS, the OS would allow one process to execute for some time and then there is a context switch. So during this context switch process 1 would stop running, & other process say process 2 starts execution.

The time delta for which a process run is called time slice.

Context switch overhead

CPU scheduling

which process the scheduler should choose??



Based on this we can say, we divide processing types -

1) I/O Bound → has small cpu burst & more I/O
2) CPU Bound → 3d rendering

#scheduling Criteria

1) Max CPU utilisation → CPU should not be idle.
2) fairness → Given each process a fair share of CPU unless they are exception.
3) Max Throughput → Complete as many process as possible per unit time.
4) Arrival time → when process enters ready queue

5) Min turnaround time → for a process total time from start to end.

6) Min waiting time

7) Min response time