Comprehensive-Mini-Notes

For more notes visit

https://rtpnotes.vercel.app

Note

These are just some portions which may be useful
Reference Youtube Playlist: https://www.youtube.com/playlist?list=PLl2mlueB4kymkBHxvZLz4NlHaEHFcq9ry

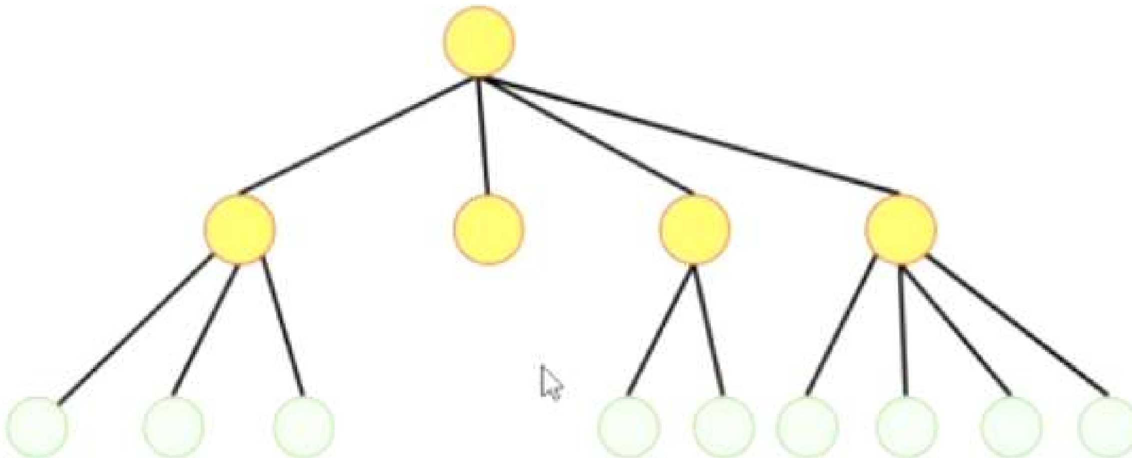
DS

1. Complexity Questions

Table 1: Complexities and stability of some sorting algorithms

Name of the algorithm	*Average case time complexity*	*Worst case time complexity*	*Stable?*
Bubble sort	$\Theta(n^2)$	$O(n^2)$	Yes
Selection sort	$\Theta(n^2)$	$O(n^2)$	No
Insertion sort	$\Theta(n^2)$	$O(n^2)$	Yes
Merge sort	$\Theta(n \log_2 n)$	$O(n \log_2 n)$	Yes
Quick sort	$\Theta(n \log_2 n)$	$O(n^2)$	No
Bucket sort	$\Theta(d(n+k))$	$O(n^2)$	Yes
Heap sort	$\Theta(n \log_2 n)$	$O(n \log_2 n)$	No

2. Number of leaves in a k-ary tree with n internal nodes



$$n(k-1)+1$$

3. Infix, Prefix, Postfix

❖Expression Notations

➢Infix Notation

- It places binary operator in between its two operands

Eg: A+B, C-D, E*F

➢Prefix Notation

- Operator symbol is placed before its own operands

Eg: +AB, *EG, -CD

➢Postfix Notation

- Operator symbol is placed after its two operands

Eg: AB+, EG*, CD-

Infix to postfix conversion video

https://www.youtube.com/watch?v=vXPL6UavUeA

- Make a table of symbol, stack and postfix
  - Add each element to symbol
  - Insert only operators and brackets to stack
  - Letters are added directly to postfix
- Precedence
  - `^` - 3
  - `* /` - 2
  - `+ -` 1
- In the stack the precedence of characters should be in ascending order
- if the order is wrong, add the symbol before it to postfix
- If the brackets are closed in stack, add the contents to postfix

infix to prefix conversion video

https://www.youtube.com/watch?v=xwqQQMVLuXI

- Reverse
- Do postfix
- Reverse again

postfix to prefix conversion video

https://www.youtube.com/watch?v=smQ88h1qzQY

- Insert each item to stack
- if a symbol is found, do operation on last 2 elements of stck
- Operator + operand1 + operand 2

posfix to infix conversion video

https://www.youtube.com/watch?v=UikLQldGV_0

prefix to postfix conversion video

https://www.youtube.com/watch?v=jZxIl0guwUo

prefix to infix conversion video

https://www.youtube.com/watch?v=U7ATLzWGsbE

4. Sequence Containers

Types of Sequence Containers in C++:

1)Vector

2)Deque

3)List

4)Forward List

5)Array

6)String

5. Data structure in BFS and DFS

bfs-queue(bfsq)

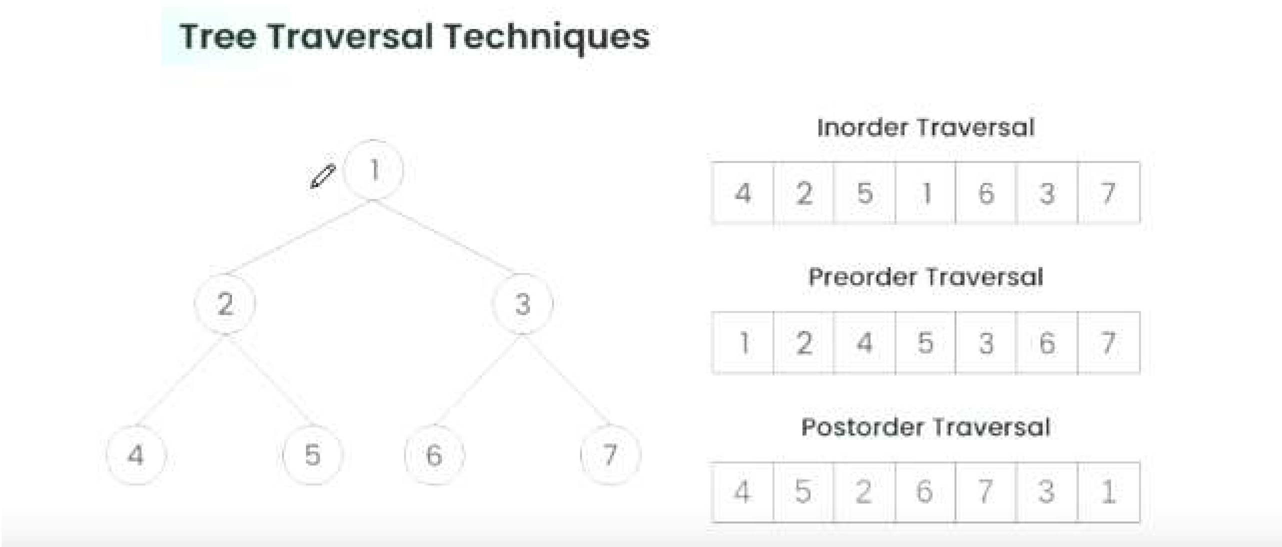
dfs-stack(dfss)

6. Tree traversal techniques

Inorder – Left,Root,Right

Preorder – Root,Left,Right

Postorder – Left,Right,Root



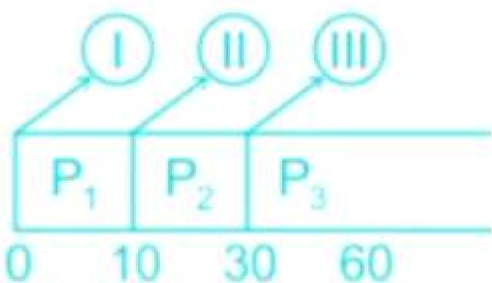
OS

1. Context switching problem

Consider three CPU-intensive processes, which require 10, 20 and 30 time units and arrive at times 0, 2 and 6 respectively. How many context switches are needed if the operating system implements a shortest remaining time first scheduling algorithm? Do not count the context switches at time zero and at the end.

Process table:

Process Id	Arrival Time (AT)	Burst Time (BT)	Completion Time (CT)
1	0	10	10
2	2	20	30
3	6	30	60



Number of context switches = 2

2. Memory space problems

A memory management system has 64 pages with 512 bytes page size. Physical memory consists of 32 page frames Number of bits required in logical and physical address are respectively:

- Number of pages = virtual memory space/page size
- Number of frames = physical memory space/frame size
- Page size = Frame size
- Virtual memory space = Number of pages * page size
    - Virtual memory space = 64 * 512B
    - Virtual memory space = 2^6 * 2^9 = 2^{15}
    - So 15 bits are required for virtual memory space
- Physical memory space = Number of frames * frames size
    - physical memory space = 32 * 512 B

      physical memory space = 2^5 * 2^9B

---

3. FIFO Page replacement Algorithm

Consider the reference string:
0 1 2 3 0 1 4 0 1 2 3 4
If FIFO page replacement algorithm is used, then the number of page faults with three page frames and four page frames are ____ and ___ respectively.

Finding page faults

0 1 2 3 0 1 4 0 1 2 3 4

FIFO-3 page frames

		2	2	2	1	1	1	1	1	3	3
	1	1	1	0	0	0	0	0	2	2	2
0	0	0	3	3	3	4	4	4	4	4	4
PF	PF	PF	PF	PF	PF	PF	NPF	NPF	PF	PF	NPF

Number of page fault=9

0 1 2 3 0 1 4 0 1 2 3 4

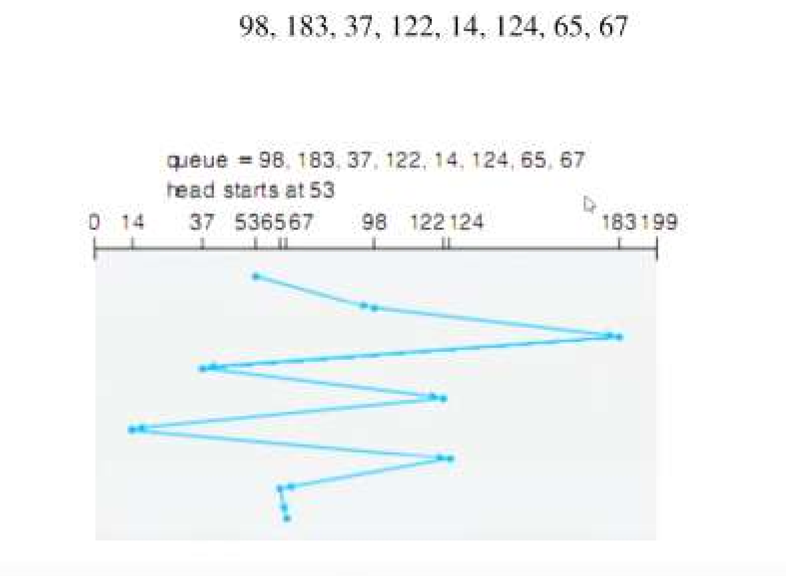
FIFO-4 page frames

			3	3	3	3	3	3	2	2	2		
		2	2	2	2	2	2	1	1	1	1		
	1	1	1	1	1	1	0	0	0	0	4		
0	0	0	0	0	0	4	4	4	4	3	3		
PF	PF	PF	PF	NPF	NPF	PF	PF	PF	PF	PF	PF		

4. Scheduling algorithms

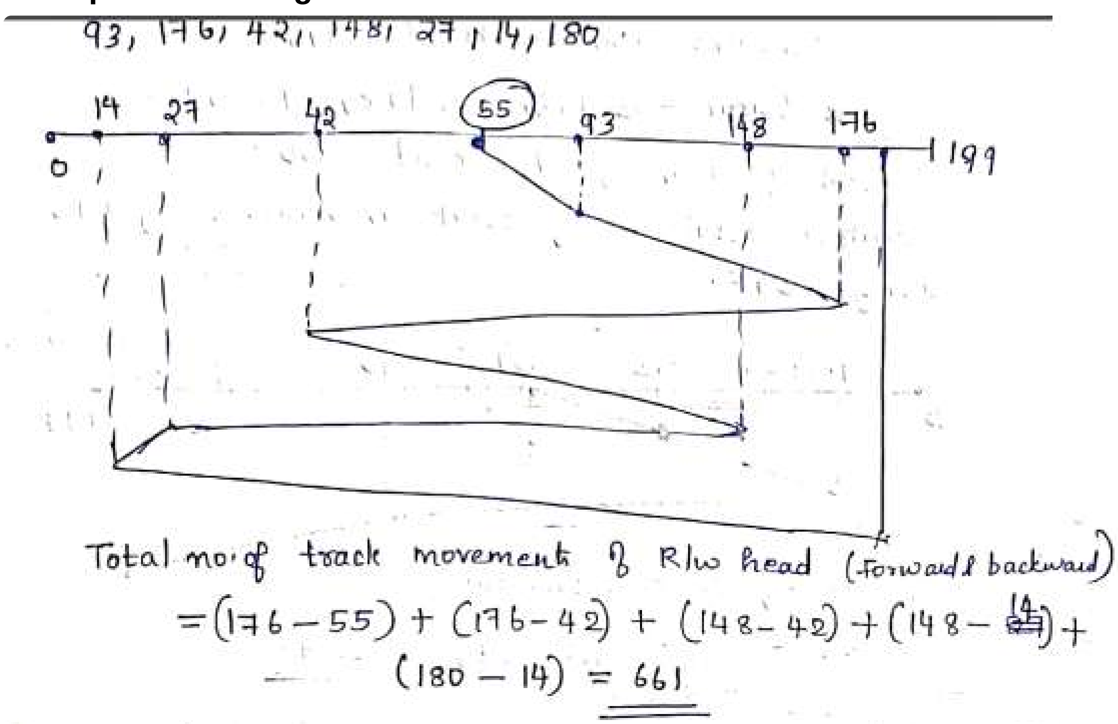
FCFS (First come first serve)

Example 1



Start from head and go to each element in queue one by one

Example 2: Tracking movement



SSTF (Shortest seek time first)

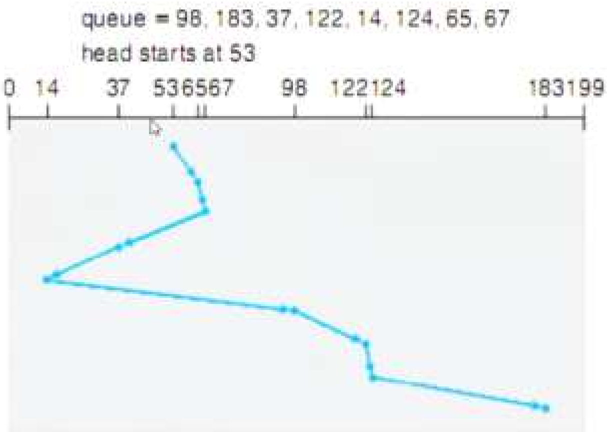
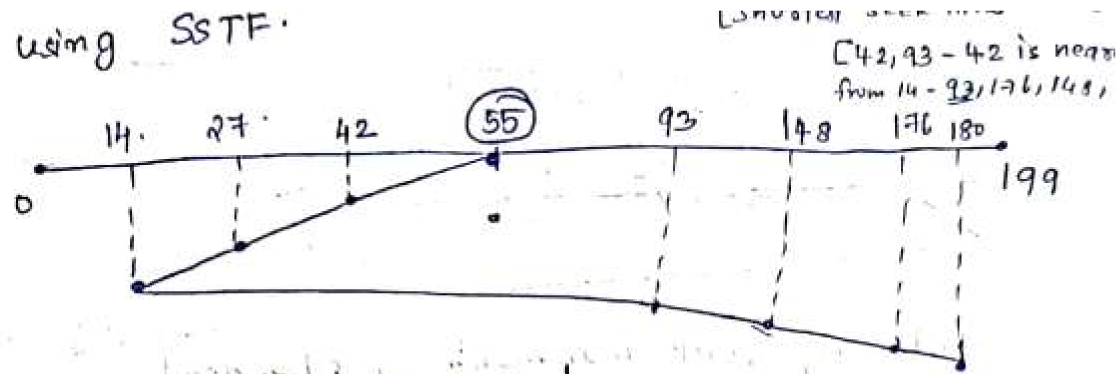


Figure 10.5 SSTF disk scheduling.



Total no. of track movement

$= (55 - 14) + (180 - 14) = 207$

SCAN Scheduling

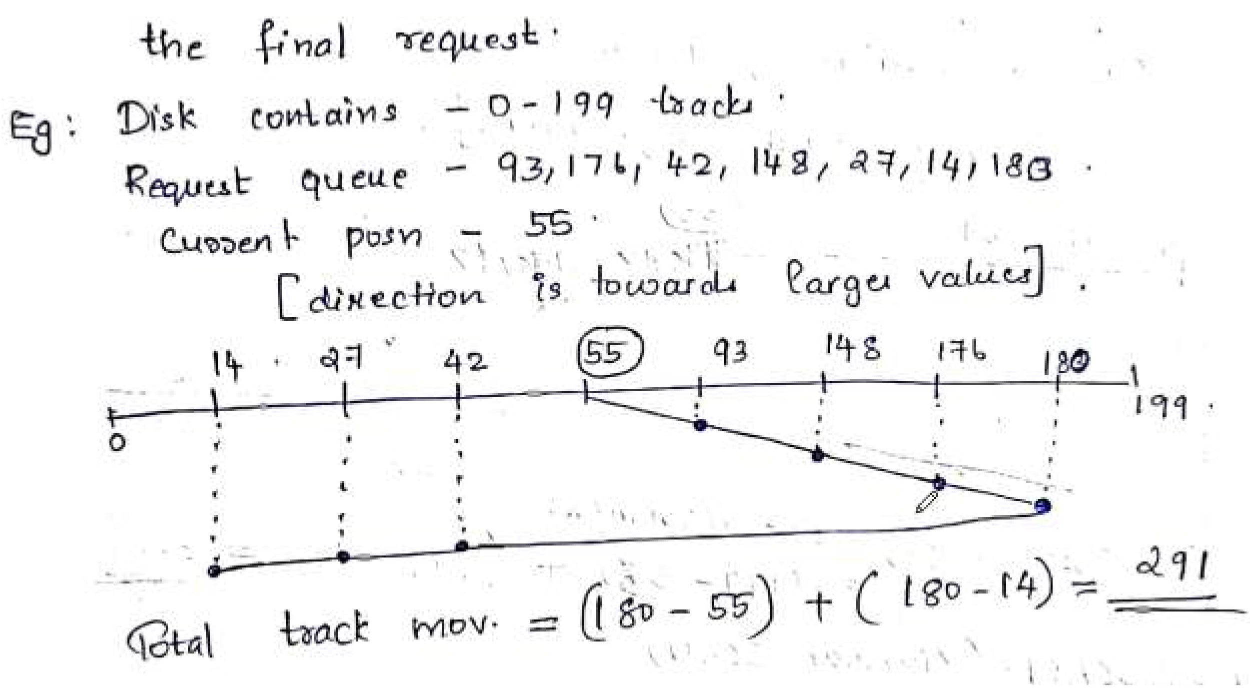
- Before applying SCAN to schedule the requests on cylinders 98, 183, 37, 122, 14, 124, 65, and 67, we need to know the direction of head movement in addition to the head's current position.
- Assuming that the disk arm is moving toward 0 and that the initial head position is again 53, the head will next service 37 and then 14.

CSCAN Scheduling

Figure 10.7 C-SCAN disk scheduling.

LOOK

Same as scan but it doesnt touch the borders, like 0 and 199



CLOOK

Same as CSCAN but doesnt touch the borders

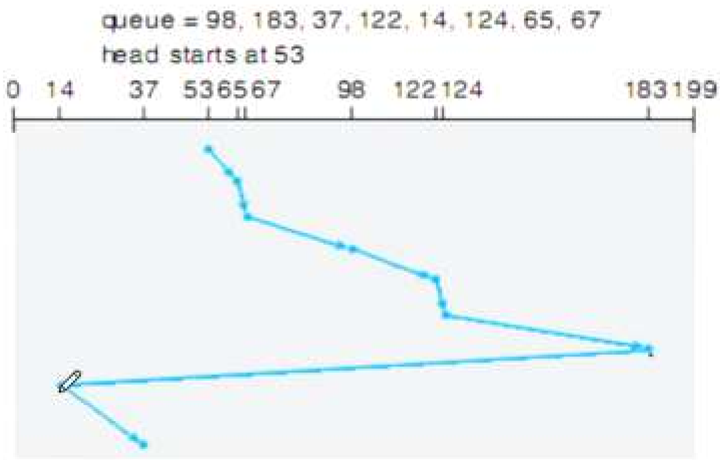
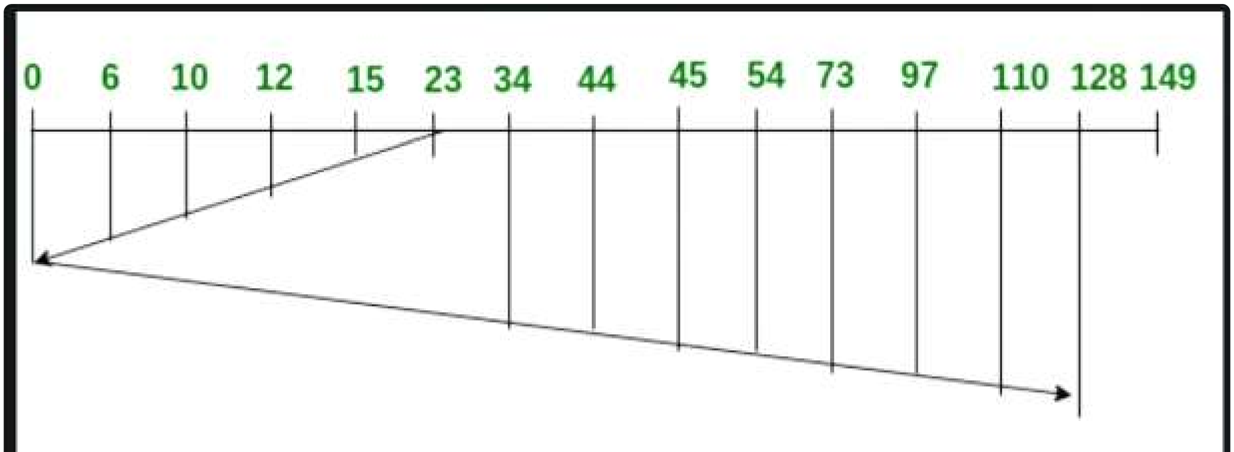


Figure 10.8 C-LOOK disk scheduling.

Question

Consider a disk queue with I/O requests on the following cylinders in their arriving order: 6,10,12,54,97,73,128,15,44,110,34,45. The disk head is assumed to be at cylinder 23 and moving in the direction of decreasing number of cylinders. Total number of cylinders in the disk is 150. The disk head movement using SCAN –scheduling algorithm is:



When it completed its last given sequence from input, it stops.
So, total disk head movement = 23+128 = 151.

5. Average Waiting time Problem

Consider a system with four processes: P1, P2, P3, and P4. The arrival times and burst times for each process are given in the table below:

Process	Arrival Time	Burst Time
P1	0	4
P2	2	6
P3	4	8
P4	6	2

Assuming the scheduling algorithm is First-Come, First-Served (FCFS), what is the average waiting time for these processes?

- Turn around time(TAT) = CT - AT
- Waiting Time(WT) = TAT - BT

CT	TAT	WT
4	4	0
10	8	2
18	14	6
20	14	12

:, First-Served (FCFS), what is the average

Average Waiting time = SUm of waiting time/ no of processes = 20 /4 = 5

7. Semaphore problem

At a particular time of computation, the value of a counting semaphore is 10. Then 12 P operations and x V operations were performed on this semaphore. If the final value of semaphore is 7, x will be

- **Initial Value:** The semaphore starts with a value of 10.
- **P Operations:** Each P operation decrements the value by 1. With 12 P operations, the value decreases by 12 (10 - 12 = -2).
- However, a semaphore value cannot be negative. In such cases, processes attempting a P operation would block until a V operation allows them to proceed.
- **V Operations:** Each V operation increments the value by 1. We need the final value to be 7. So, starting from -2 (after P operations), we need to add 9 (7 - (-2) = 9).
- **Finding x:** Since each V operation adds 1, we need x to be equal to 9.

- Therefore, for the final semaphore value to be 7, x (the number of V operations) must be **9**.

FLAT

1. 5-Tuple finite automata

Explanation: A FA can be represented as FA = (Q, ∑, δ, q0, F) where Q=Finite Set of States, ∑=Finite Input Alphabet, δ=Transition Function, q0=Initial State, F=Final/Acceptance State).

2. Pumping Lemma

- Pumping lemma is used to prove that given grammar is not regular

3. Regular language rule

Explanation: If regular languages are closed under an operation op, then the resultant of the languages over an operation op will also be regular At first stage 1/2 L will be regular and subsequently, all the options will be regular

4. Context free languages

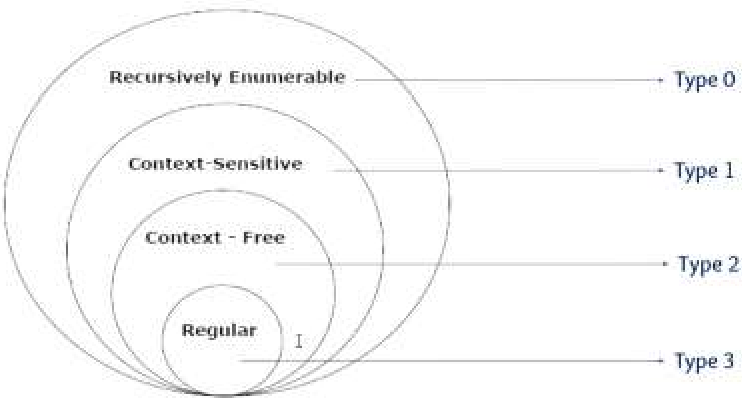
If L1 and L2 are two context free languages, then-

- L1 ∪ L2 is also a context free language.
- L1.L2 is also a context free language.
- L1* and L2* are also context free languages.
- L1 ∩ L2 is not a context free language.
- L1' and L2' are not context free languages.

5. Chomsky hierarchy

According to Chomsky Hierarchy, there are four types of grammars – Type 0, Type 1, Type 2, and Type 3.

Grammar Type	Grammar Accepted	Language Accepted	Automaton
Type 0	Unrestricted grammar	Recursively enumerable language	Turing Machine
Type 1	Context-sensitive grammar	Context-sensitive language	Linear-bounded automaton
Type 2	Context-free grammar	Context-free language	Pushdown automaton
Type 3	Regular grammar	Regular language	Finite state automaton



Type 3: Regular Grammar

- Generate regular languages.
- Type-3 grammars must have a single non-terminal on the left-hand side and a right-hand side consisting of a **single terminal** or **single terminal followed by a single non-terminal**.
- The productions must be in the form

  $X \rightarrow a$ or $X \rightarrow aY$
- where **X, Y ∈ V** (Non terminal)
- **a ∈ T** (Terminal)
- The rule **S** → ε is allowed if **S** does not appear on the right side of any rule.

Example :

S□a,A□bB, B□bC, C□b

The Automata that accept the Type-3 languages is **Finite State Automata**

Type 2: Context free grammar

Type 2 – Context Free Grammar

- Generate context-free languages.
- The productions must be in the form $A \rightarrow \beta$
  - Where $A \in V$(Non terminal)
  - $\beta \in (V \cup T)^*$ (String of terminals and non-terminals).

- The left side of the production should contain only one non-terminal
- No restriction on the right side of the production

- **Example :**
  - S□aA , A□b, B□BC, C□aAB

The languages generated by Type-2 grammars are be recognized by a **Pushdown Automaton(PDA).**

Type 1: Context sensitive grammar

Example :

S□aTb | ab

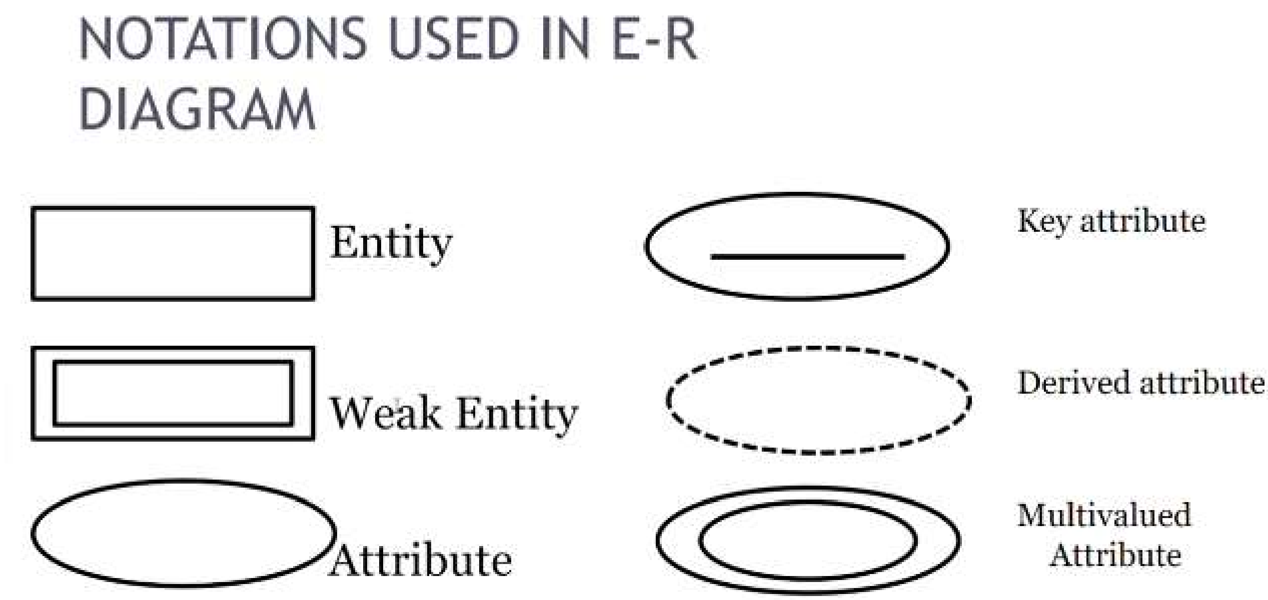
aT□aaTb | ac

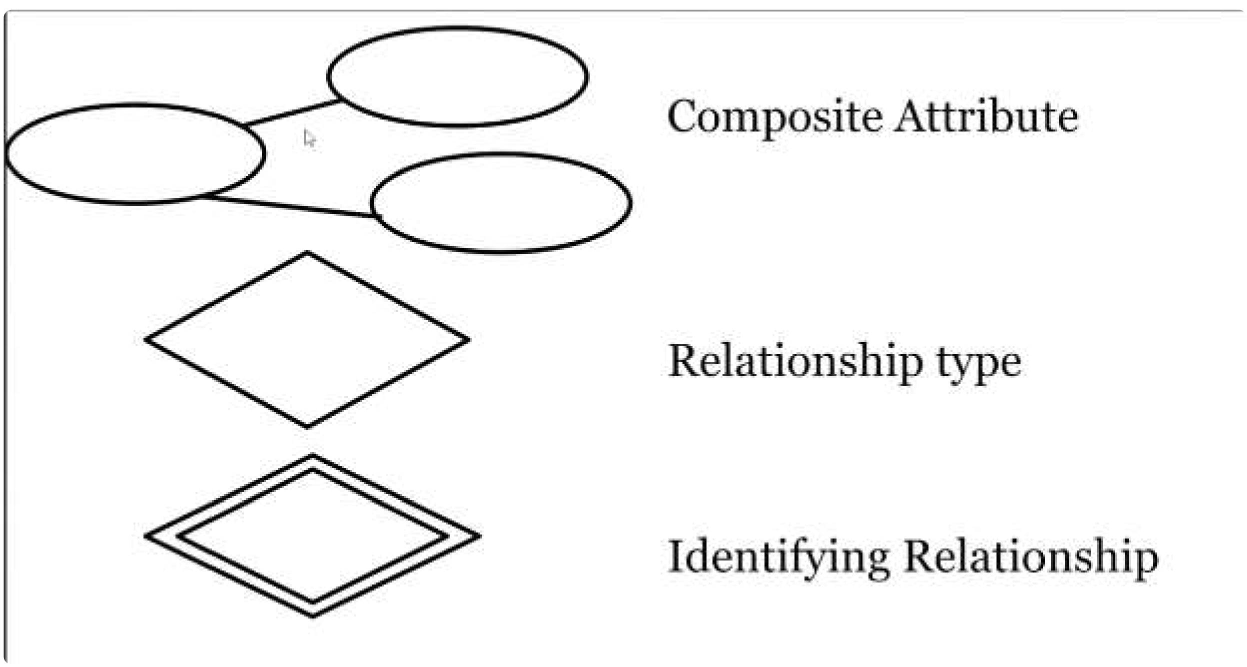
The rule $S \rightarrow \varepsilon$ is allowed if S does not appear on the right side of any rule.

The languages generated by these grammars are recognized by a Linear Bounded Automaton(LBA).

DBMS

1. Notations





2. Cardinality Ratio

Cardinality Ratio

- The cardinality ratio for a binary relationship specifies the maximum number of relationship instances to which an entity can take part in it
- It also specifies number of entities to which other entity can be related by a relationship
- Types
  - One-to-one (1:1)
  - One-to-many (1: N)
  - Many-to-one (N: 1)
  - Many-to-many (M: N)

Cardinality ratio, in simple terms, refers to how many "things" from one group can be connected to how many "things" from another group in a database. Imagine you have a database for a library:

- One group could be **Books** (entity set).
- Another group could be **Authors** (entity set).

The cardinality ratio tells you how many **Books** can be written by one **Author** and vice versa. There are three main types of cardinality ratios:

1. **One-to-One (1:1):** This means one book can only be written by one author, and one author can only write one book (not very realistic in a library!).
2. **One-to-Many (1:N):** This is more common. One author can write many books (N), but a book can only have one author (1).
3. **Many-to-Many (N:M):** This is also common. Many books can have many authors (a book can be written by multiple people, and an author can write multiple books).

3. Natural Join

Consider table R (a, b) as ;

a	b
1	3
1	5

Relation S(a, c) as :

a	c
1	2
1	4
1	6

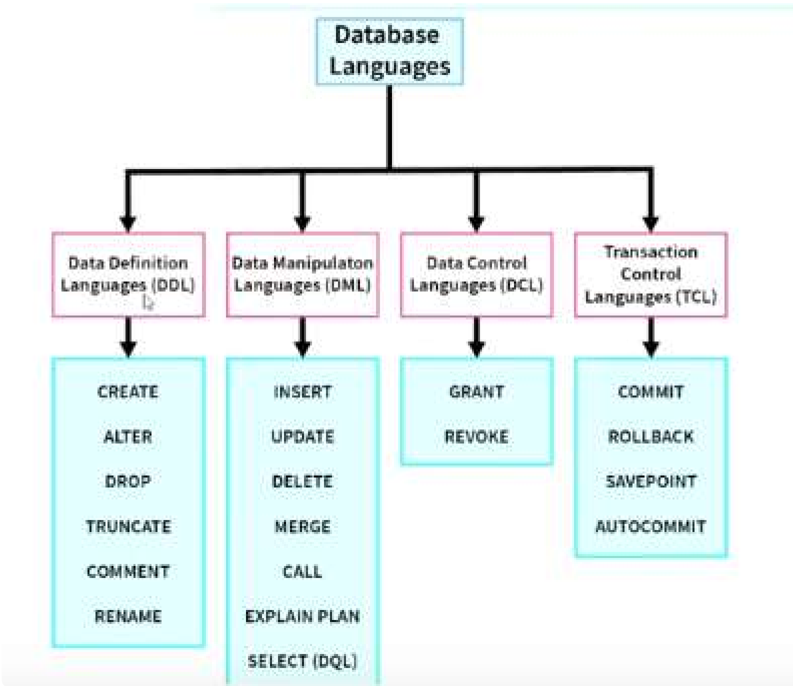
Here, the natural join of R and S will be :

a	b	c
1	3	2
1	3	4
1	3	6
1	5	2
1	5	4
1	5	6

So, it contains total of 6 tuples.

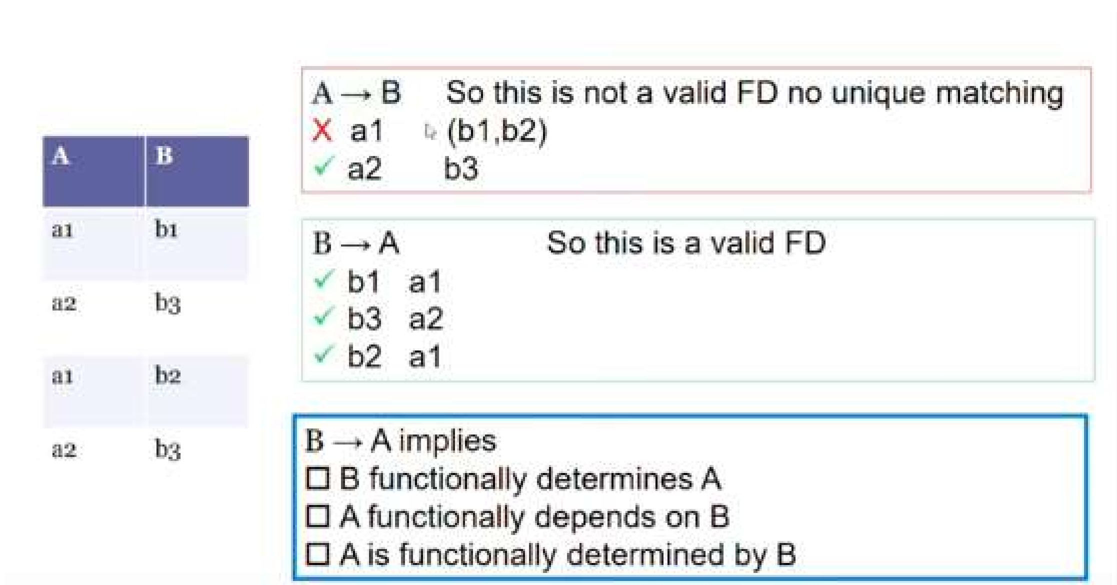
Maximum size of join = mn

4. Types of Database languages



5. Functional dependency

- A -> B
  - a1 has b1
  - But a1 also has b2
  - This means a1 does not have a unique matching
  - This means its not a functional dependency
- B -> A
  - Every match is unique
  - So its an FD



6. Closure set of attribute

- R(A,B,C,D)with FD={A→B, B→C,C→D,D→A}
- Closure of A, A^+=attribute which can be determined from A
- A^+=ABCD (ie using closure if we can cover all attribute then it is called Candidate Key CK)
- B^+=BCDA ✔CK
- C^+=CDAB ✔ CK
- D^+=DABC ✔ CK
- Candidate Keys of R are A,B,C,D

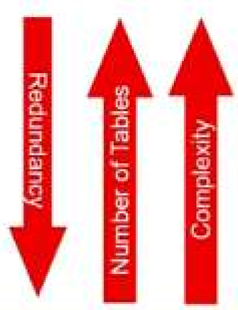
7. Prime attribute

Prime and Non Prime Attributes

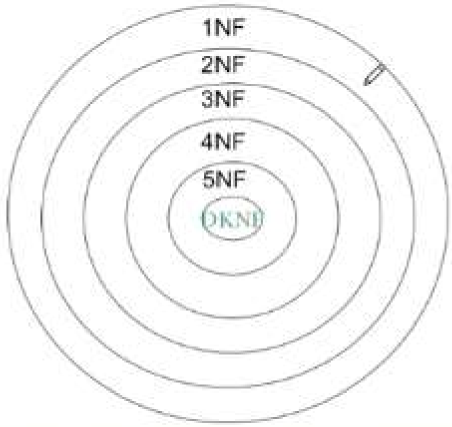
- An attribute of relation schema R is called a prime attribute of R if it is a member of some candidate key of R.
- An attribute is called nonprime if it is not a prime attribute—that is, if it is not a member of any candidate key.

8. Normalization

- Levels of normalization based on the amount of redundancy in the database.
- Various levels of normalization are:
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - Third Normal Form (3NF)
  - Boyce-Codd Normal Form (BCNF)



Most databases should be 3NF or BCNF in order to avoid the database anomalies.

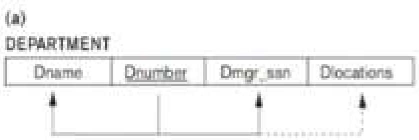


Each higher level is a subset of the lower level

First Normal Form

Problem

this is not in 1NF because Dlocations is not an atomic attribute



(b)
DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

- Here DLocations have multiple values, which is not atomic

Solution

- There are three main techniques to achieve first normal form for such a relation:
- 1. Remove the attribute Dlocations that violates 1NF and place it in a separate relation DEPT_LOCATIONS along with the primary key Dnumber of DEPARTMENT. The primary key of this relation is the combination {Dnumber, Dlocation},

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Second Normal Form

Second Normal Form

- Second normal form (2NF) is based on the concept of full functional dependency.
- A functional dependency $X \rightarrow Y$ is a **full functional dependency** **if removal of any attribute A from X means that the dependency does not hold any more**;
- that is, for any attribute $A \varepsilon X$, $(X - \{A\})$ does not functionally determine Y.

Qn.Consider Relation Std_Faculty(Sid,Cid, Mark, Faculty).With FD={Sid,Cid → Marks, Cid → Faculty}. Is this in 2NF?

Ans. Find CK,

(Sid,Cid)+=Sid, Cid,Marks, Faculty

So (Sid,Cid) is a candidate key here

Here Prime attribute=Sid,Cid

Non Prime attribute=Marks, Faculty

Now check for Parial functional dependency

Sid,Cid → Marks ✔Full functional dependent

Cid → Faculty ☐ Partial dependency

So not in 2NF

Third Normal form

- Definition. According to Codd's original definition, a relation schema R is in 3NF if it satisfies 2NF and **no** nonprime attribute of R is transitively dependent on the primary key.

Consider Relation Employee_Dept(Eid,Ename,Dno, Dmgrid).With FD={Eid → Ename, Eid → Dno, Dno → Dmgrid}. Is this in 3NF?

Ans. First check for 2NF

No partial dependency so is in 2NF

Find CK, Eid+=Eid,Ename,Dno,Dmgrid ✔CK

Prime Attribute= {Eid}

Non Prime Attributes={Ename,Dno, Dmgrid}

Now check for transitive dependency,

Eid → Dno,

Dno → Dmgrid

- This is transitive dependency. So not in 3NF

BCNF (Boyce Codd Normal Form)

- It is an advance version of 3NF that's why it is also referred as 3.5NF.
- BCNF is stricter than 3NF.
- A table complies with BCNF if it is in 3NF and for every functional dependency X->Y, **X should be the super key** of the table.

EMP_ID	EMP_COUNTRY	EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
264	India	Designing	D394	283
264	India	Testing	D394	300
364	UK	Stores	D283	232
364	UK	Developing	D283	549

In the above table Functional dependencies are as follows:

EMP_ID → EMP_COUNTRY

EMP_DEPT → {DEPT_TYPE, EMP_DEPT_NO}

The table is not in BCNF because neither EMP_DEPT nor EMP_ID alone are keys.

To convert the given table into BCNF, we decompose it into three tables:

EMP_COUNTRY table:

EMP_ID	EMP_COUNTRY
264	India
264	India

EMP_DEPT table:

EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
Designing	D394	283
Testing	D394	300
Stores	D283	232
Developing	D283	549

Summary

Summary

- 1NF: Ensure Atomicity
- 2NF: Must be in 1NF + Ensure no partial dependency

Proper subset of any key of R → Non Prime attributes
or //BOTH NOT ALLOWED IN 2NF
Prime attribute → Non Prime attributes

- 3NF: Must be in 2NF & No transitive dependency

A relation R is in 3NF if for every FD X → Y
Either X is a SK or
Y is a prime attribute of R

- BCNF: X → Y Where X is a Super Key

9. Lossless and Lossy Decomposition

Lossless	Lossy
The decompositions R1, R2, R2...Rn for a relation schema R are said to be Lossless if there natural join results the original relation R.	The decompositions R1, R2, R2...Rn for a relation schema R are said to be Lossy if there natural join results into addition of extraneous tuples with the original relation R.

10. ACID Properties

Desirable Properties of a Transaction ACID properties:

1. Atomicity
2. Consistency
3. Isolation
4. Durability

The Lost Update Problem

- This occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect.

Time

T_1	T_2
read_item(X);	
$X := X - N$;	
	read_item(X);
	$X := X + M$;
write_item(X);	
read_item(Y);	
	write_item(X);
$Y := Y + N$;	
write_item(Y);	

Item X has an incorrect value because its update by T_1 is *lost* (overwritten).

11. Schedules of transactions

Schedules(Histories) of Transactions

- A schedule S of n transactions T1, T2,.. Tn is an ordering of the operation of the transactions
- When transactions are executing concurrently in an interleaved fashion, the order of execution of operations from the various transactions forms what is known as a transaction schedule (or history).

1. Serial Schedule
2. Non Serial Schedule
3. Serializable Schedule

Serial Schedule

- Entire transactions are performed in serial order
- Only One transaction is active at a time
- Serial schedule : **T1T2**

T1	T2
R(A)	
A=A+50	
R(B)	
B=B-30	
W(B)	
	R(A)
	A=A+40
	W(A)
	R(B)
	B=B-60
	W(B)

Initially	Final Value
A=100	A=190
B=200	B=110

Non Serial Schedules

- Executing the transaction in an interleaved or instructions of transactions are interleaved. Schedule S1

T1	T2
R(A)	
A=A+50	
W(A)	
	R(A)
	A=A+40
	W(A)
R(B)	
B=B-30	
W(B)	
	R(B)
	B=B-60
	W(B)

Initially	Final Value
A=100	A=190
B=200	B=110

Schedule S1 gives the correct result same as serial schedule

Serializable Schedule

- A non serial schedule of n transactions is said to be serializable if it is equivalent to some serial schedule of same n transactions
- Every serializable schedules are considered as correct

12. Transaction Conflicts

- Write-Read (WR) conflict
- Read-Write (RW) conflict
- Write-Write (WW) conflict

Write-Read Conflict

What is Write-Read (WR) conflict?

This conflict occurs when a transaction read the data which is written by the other transaction before committing.

In the following diagram, I am using the following notions

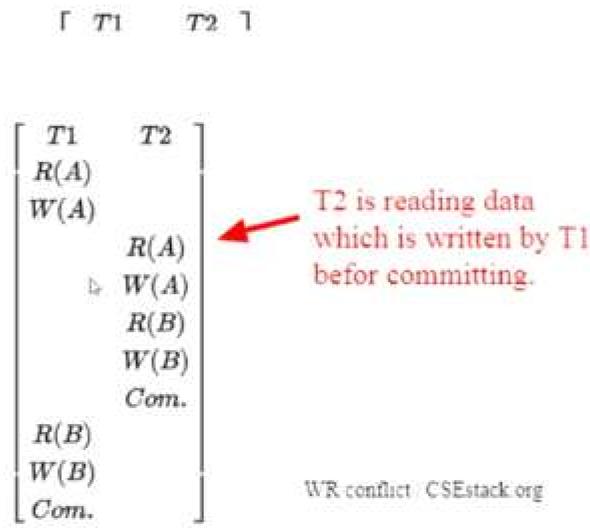
```
A, B – two different data objects from database

T1, T2 – two different transactions

R(A) – reading data A

W(A) – Writing data A

Com. – committing transaction
```



Here, the transaction T2 is reading the data which is written by the T1 before T2 commits. It is also called as **Dirty Read**.

It violates the ACID property of data consistency rule.

Read-Write Conflict

What is Read-Write (RW) conflict?

$$\begin{bmatrix} T1 & T2 \\ R(A) & \\ & R(A) \\ & W(A) \\ & Com. \\ R(A) & \\ W(A) & \\ Com. & \end{bmatrix}$$

T2 is writing data which is read by T1.

RW Comflict : CSEstack.org

Transaction T2 is Writing data which is previously read by transaction T1.

Here if you look at the diagram above, data read by transaction T1 before and after T2 commits is different.

What is Write-Write (WW) conflict?

$$\begin{bmatrix} T1 & T2 \\ W(A) & \\ & W(B) \\ W(B) & \\ Com. & \\ & W(A) \\ & Com. \end{bmatrix}$$

T2 is witting data which is written by T1 before T1 commits.

WW conflict : CSEstack.org

Here Transaction T2 is writing data which is already written by other transaction T1. T2 overwrites the data written by T1. It is also called as a **blind write operation**.

Data written by T1 has vanished. So it is **data update loss**.

13. Conflict serializable

To check a schedule is conflict serializable or not, we can use precedence graph

If the precedence graph is free from cycles(acyclic) then schedule is conflict serializable schedule

Rules

Rules for constructing Precedence Graph:

Rule 1

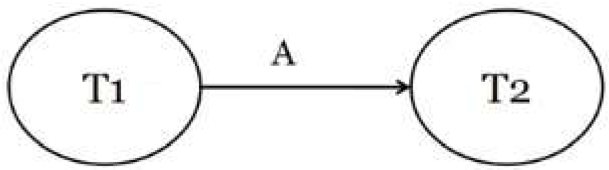
S1:

T1	**T2**
R(A)	
	W(A)

Rule 2

S2:

T1	**T2**
W(A)	
	W(A)



Rule 3

S3:

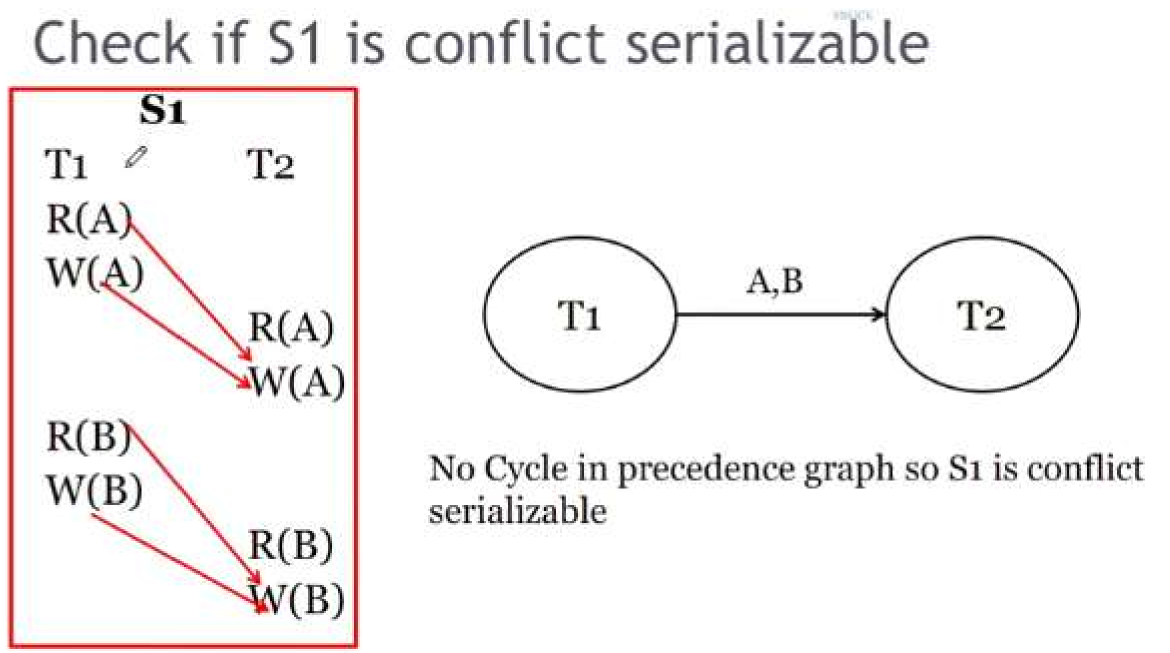
T1	**T2**
W(A)	
	R(A)

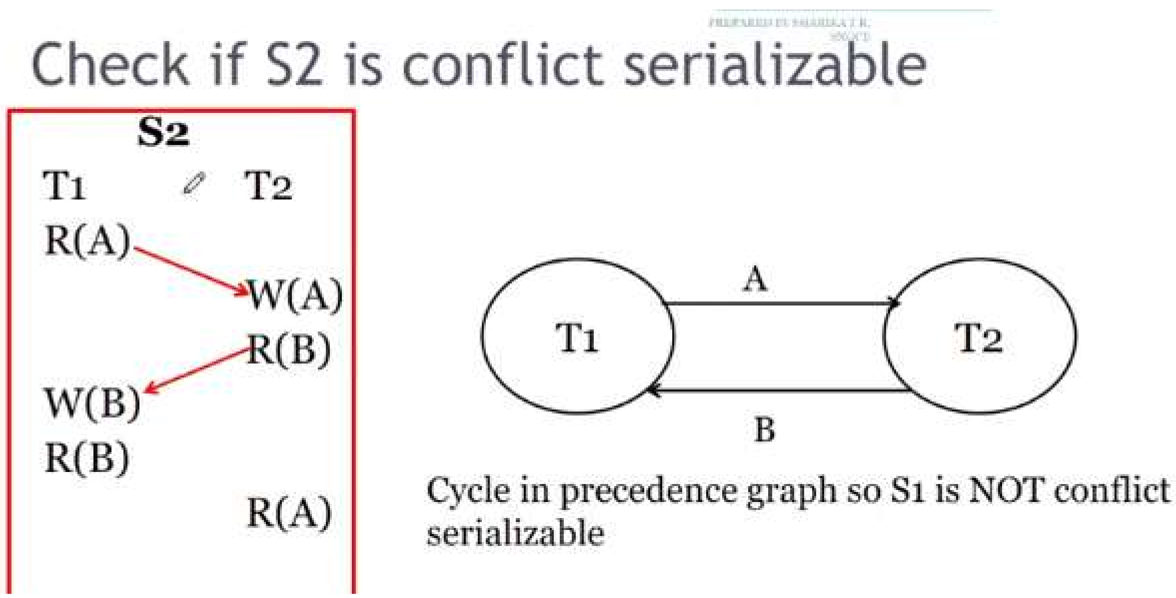
Rule 4

S4:

T1	**T2**
R(A)	
	R(A)

Example





14. NoSQL Database

Features of NoSQL

PREPARED BY SHARIKA T R, SNGCE

- Non-relational
  - NoSQL databases never follow the relational model
  - Never provide tables with flat fixed-column records
  - Work with self-contained aggregates or BLOBs
  - Doesn't require object-relational mapping and data normalization
  - No complex features like query languages, query planners,referential integrity joins,