expression which is saved with the name X.

- **Example-1:** Query to rename the relation Student as Male Student and the attributes of Student – RollNo, SName as (Sno, Name).

Sno	Name
2600	Ronny
2655	Raja

ρ `MaleStudent(Sno, Name)` $\pi_{\texttt{RollNo, SName}}(\sigma_{\texttt{Condition}}(\texttt{Student}))$

- **Example-2:** Query to rename the attributes Name, Age of table Department to A,B.

ρ `(A, B) (Department)`

- **Example-3:** Query to rename the name Project to Pro and its attrib to P, Q, R.

- **Example-2:** Query to rename the attributes Name, Age of table Department to A,B.

```
ρ (A, B) (Department)
```

- **Example-3:** Query to rename the table name Project to Pro and its attributes to P, Q, R.

```
ρ Pro(P, Q, R) (Project)
```

- **Example-4:** Query to rename the first attribute of the table Student with attributes A, B, C to P.

```
ρ (P, B, C) (Student)
```

use of positional notation to write the expression we saw earlier, which computes salaries that are not the largest:

$$\Pi_{\$4}\ (\sigma_{\$4\ <\ \$8}\ (instructor \times instructor))$$

Note that the Cartesian product concatenates the attributes of the two relations. Thus, for the result of the Cartesian product (*instructor* × *instructor*), \$4 refers to the *salary* attribute from the first occurrence of *instructor*, while \$8 refers to the *salary* attribute from the second occurrence of *instructor*. A positional notation can also be used to refer to relation names, if a binary operation needs to distinguish between its two operand relations. For example, $\$R1$ could refer to the first operand relation, and $\$R2$ could refer to the second operand relation of a Cartesian product. However, the positional notation is inconvenient for humans, since the position of the attribute is a number, rather than an easy-to-remember attribute name. Hence, we do not use the positional notation in this textbook.

6.1.2 Formal Definition of the Relational Algebra

The operations in Section 6.1.1 allow us to give a complete definition of an expression in the relational algebra. A basic expression in the relational algebra consists of either one of the following:

- A relation in the database
- A constant relation

A constant relation is written by listing its tuples within { }, for example { (22222, Einstein, Physics, 95000), (76543, Singh, Finance, 80000) }.

A general expression in the relational algebra is constructed out of smaller subexpressions. Let E_1 and E_2 be relational-algebra expressions. Then, the following are all relational-algebra expressions:

- $E_1 \cup E_2$
- $E_1 - E_2$
- $E_1 \times E_2$
- $\sigma_P(E_1)$, where P is a predicate on attributes in E_1
- $\Pi_S(E_1)$, where S is a list consisting of some of the attributes in E_1
- $\rho_x(E_1)$, where x is the new name for the result of E_1

6.1.3 Additional Relational-Algebra Operations

The fundamental operations of the relational algebra are sufficient to express any relational-algebra query. However, if we restrict ourselves to just the fundamental operations, certain common queries are lengthy to express. Therefore, we define additional operations that do not add any power to the algebra, but simplify

Note that there is often more than one way to write a query in relational algebra. Consider the following query:

$$\Pi_{name,\ course\_id}\ (\sigma_{instructor.ID = teaches.ID}((\sigma_{dept\_name = \text{"Physics"}}(instructor)) \times teaches))$$

Note the subtle difference between the two queries: in the query above, the selection that restricts *dept_name* to Physics is applied to *instructor*, and the Cartesian product is applied subsequently; in contrast, the Cartesian product was applied before the selection in the earlier query. However, the two queries are equivalent; that is, they give the same result on any database.

6.1.1.7 The Rename Operation

Unlike relations in the database, the results of relational-algebra expressions do not have a name that we can use to refer to them. It is useful to be able to give them names; the rename operator, denoted by the lowercase Greek letter rho (ρ), lets us do this. Given a relational-algebra expression E, the expression

$$\rho_x\,(E)$$

returns the result of expression E under the name x.

A relation r by itself is considered a (trivial) relational-algebra expression. Thus, we can also apply the rename operation to a relation r to get the same relation under a new name.

A second form of the rename operation is as follows: Assume that a relational-algebra expression E has arity n. Then, the expression

$$\rho_{x(A_1, A_2, \ldots, A_n)}\,(E)$$

returns the result of expression E under the name x, and with the attributes renamed to $A_1, A_2, \ldots, A_n$.

To illustrate renaming a relation, we consider the query "Find the highest salary in the university." Our strategy is to (1) compute first a temporary relation consisting of those salaries that are *not* the largest and (2) take the set difference between the relation $\Pi_{salary}\,(instructor)$ and the temporary relation just computed, to obtain the result.

1. Step 1: To compute the temporary relation, we need to compare the values of all salaries. We do this comparison by computing the Cartesian product *instructor* $\times$ *instructor* and forming a selection to compare the value of any two salaries appearing in one tuple. First, we need to devise a mechanism to distinguish between the two *salary* attributes. We shall use the rename operation to rename one reference to the instructor relation; thus we can reference the relation twice without ambiguity.

Conceptual

1.

salary
65000
90000
40000
60000
87000
75000
62000
72000
80000
92000

Figure 6.11 Result of the subexpression
$\Pi_{instructor.salary} (\sigma_{instructor.salary < d.salary} (instructor \times \rho_d (instructor)))$.

We can now write the temporary relation that consists of the salaries that are not the largest:

$$\Pi_{instructor.salary} (\sigma_{instructor.salary < d.salary} (instructor \times \rho_d (instructor)))$$

This expression gives those salaries in the *instructor* relation for which a larger salary appears somewhere in the *instructor* relation (renamed as *d*). The result contains all salaries *except* the largest one. Figure 6.11 shows this relation.

2. Step 2: The query to find the largest salary in the university can be written as:

$$\Pi_{salary} (instructor) - \\ \Pi_{instructor.salary} (\sigma_{instructor.salary < d.salary} (instructor \times \rho_d (instructor)))$$

Figure 6.12 shows the result of this query.

The rename operation is not strictly required, since it is possible to use a positional notation for attributes. We can name attributes of a relation implicitly by using a positional notation, where \$1, \$2, ... refer to the first attribute, the second attribute, and so on. The positional notation also applies to results of relational-algebra operations. The following relational-algebra expression illustrates the

salary
95000

Figure 6.12 Highest salary in the university.

Read-write conflict

- Read after write (W-R)
- Write after read (R-W)
- Write after write (W-W)

① W-R

T_1	T_2
R(A)	
W(A)	
	R(A)
	W(A)

→ W-R conflict

T_1 commit এর পূর্বে T_2 data read করছে

② R-W

T_1	T_2
R(A)	
	R(A)
	W(A)

→ R-W conflict

T_1 এর read করা Data T_2 write করছে,

③ W-W

T_1	T_2
W(A)	
	W(A)

W-W Conflict
T_1 এর write করা Data T_2 write করছে, T_1 এর commit এর পূর্বে

#

T_1	T_2
R(A)	
W(A)	
	R(A)
	W(A)
R(B)	
W(B)	

Conflicts
→ R-W
→ W-R
→ W-W

0:37 ··· 88

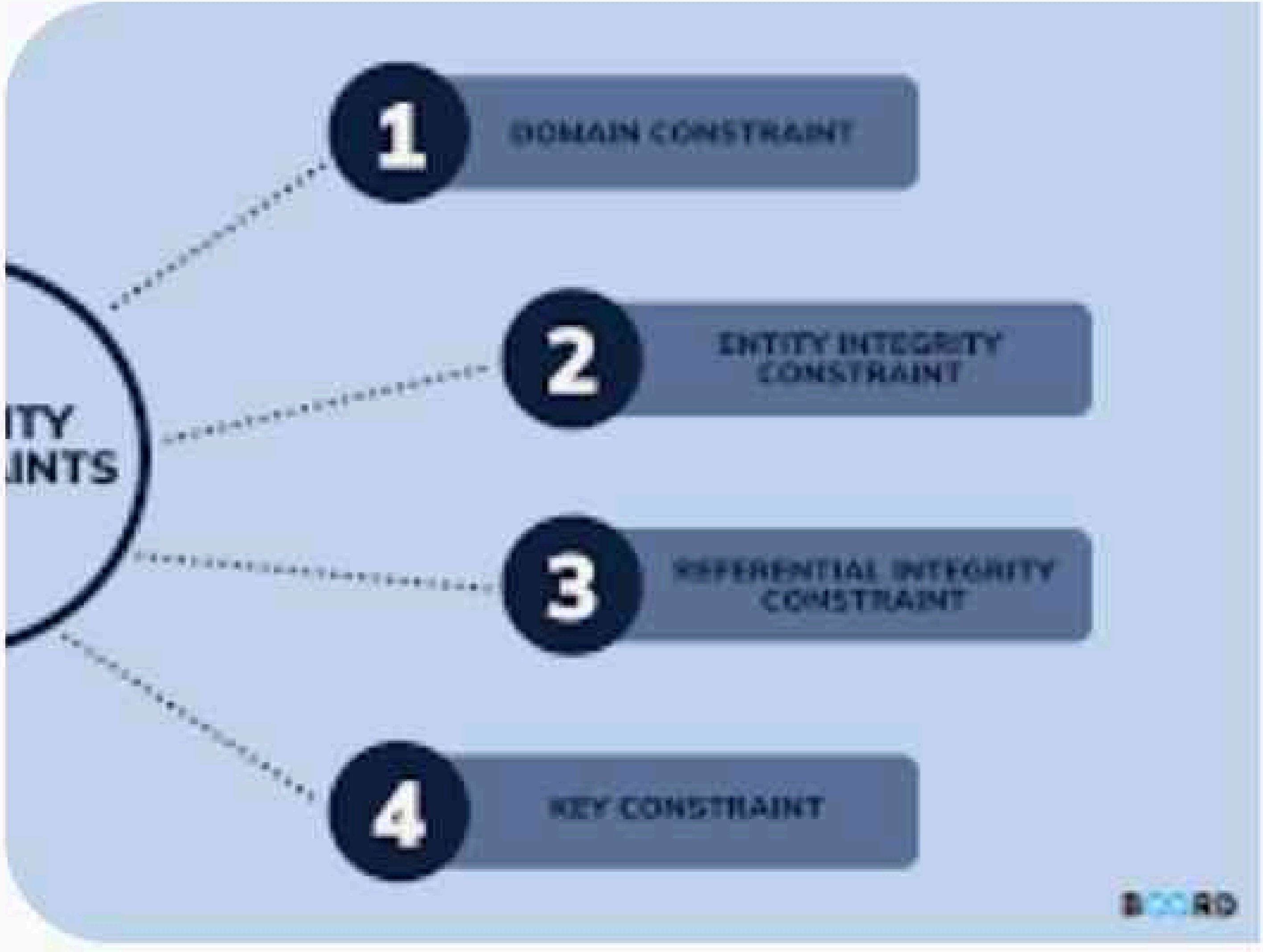
Google

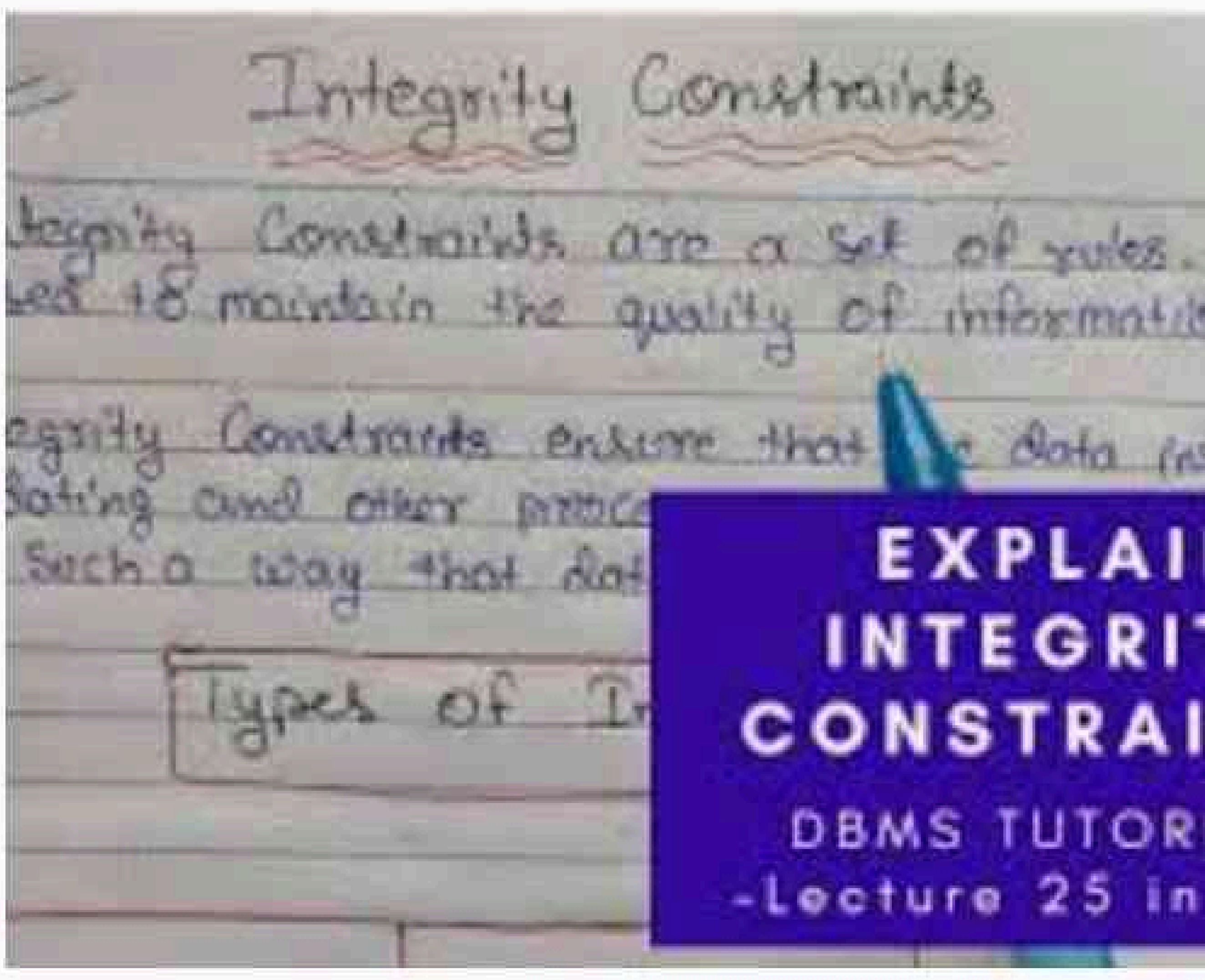
integrity constraints in db...

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন

ডিবিএমএসে অখণ্ডতার সীমাবদ্ধ





বাংলায় In English

Integrity constraints in DBMS are used to ensure that data is consistent and accurate. There are four main types of integrity constraints: domain, entity, referential, and key. Here, we'll take a closer look & explain the types of integrity constraints along with some examples. Sep 5, 2023

https://www.knowledgehut.com › in...

Discover Search Saved

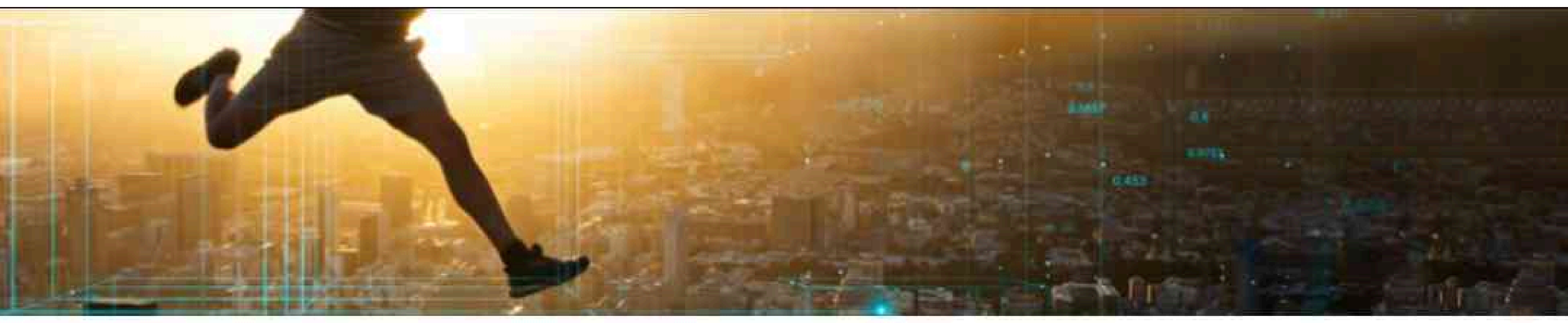
A DBMS uses a transaction log to keep track of all transactions that update the database. The information stored in this log is used by the DBMS for a recovery requirement triggered by a ROLLBACK statement, a program's abnormal termination, or a system failure such as a network discrepancy or a disk crash. Some RDBMSs use the transaction log to recover a database forward to a currently consistent state. After a server failure, for example, Oracle automatically rolls back uncommitted transactions and rolls forward transactions that were committed but not yet written to the physical database.

While the DBMS executes transactions that modify the database, it also automatically updates the transaction log.

The transaction log stores:

5:07 98

Login | Skills for All with Cisco
auth.socialgoodplatform.com



Go back

Welcome!

Please login to your account.

Email

jemi39shifa@gmail.com

Password

••••••••••••

Remember me

Forgot Password?

Login

Or continue with

Google

cisco Networking Academy

5:09 ··· 99

Sign in - Google Accounts
accounts.google.com

Sign in with Google

Sign in

to continue to Skills For All

Email or phone

jemi39shifa@gmail.com

Forgot email?

To continue, Google will share your name, email address, language preference, and profile picture with Skills For All. Before using this app, you can review Skills For All's privacy policy and terms of service.

Create account

Next

English (United States)

Help Privacy Terms

5:09

Sign in - Google Accounts

accounts.google.com

Sign in with Google

Welcome

jemi39shifa@gmail.com

Enter your password

Mashifa860%

Show password

To continue, Google will share your name, email address, language preference, and profile picture with Skills For All. Before using this app, you can review Skills For All's **privacy policy** and **terms of service**.

~ ` | • √ π ÷ × § Δ

€ ¥ $ ¢ ^ ° = { } \

?123 % © ® ™ ✓ [ ]

ABC < 12 34 English >

5:15 100

Open Weaver

Basics of JavaScript for beginners | Learn by Doing

Start your Journey in JavaScript with a focus on:

1. Basics of JavaScript focusing on data types, variable declaration, built in functions and HTML elements.

2. Hands-on session to "Create a User Login Page".

3. Learn how you can use kandi (pronounced like candy) to tap into millions of open-source assets to build your JavaScript application.

Sign up by providing your info below.

Email

jemi39shifa@gmail.com

First name

Habiba

Last name

Akter

Country

Select

By clicking Submit, you agree to send your info to Open Weaver who agrees to use it according to their privacy policy. Facebook will also use it subject to our Data Policy, including to auto-fill forms for ads. View Meta Privacy Policy. OW Privacy Policy

Submit

* Projection π → Distinct (Col filter)
Selection σ → condition (row filter)
Join → (Table filter)

$\sigma_{D\text{-}C} \cap C = 3$

ins ⋈ stu ⋈ Adv

~~σ ID~~ $\pi_{St.ID}(\sigma_{In.Name = E})$

* Aggregate → $G_{max(T_1)}$ → 6.1.4.2 Book

$G_{min(T_2)}$

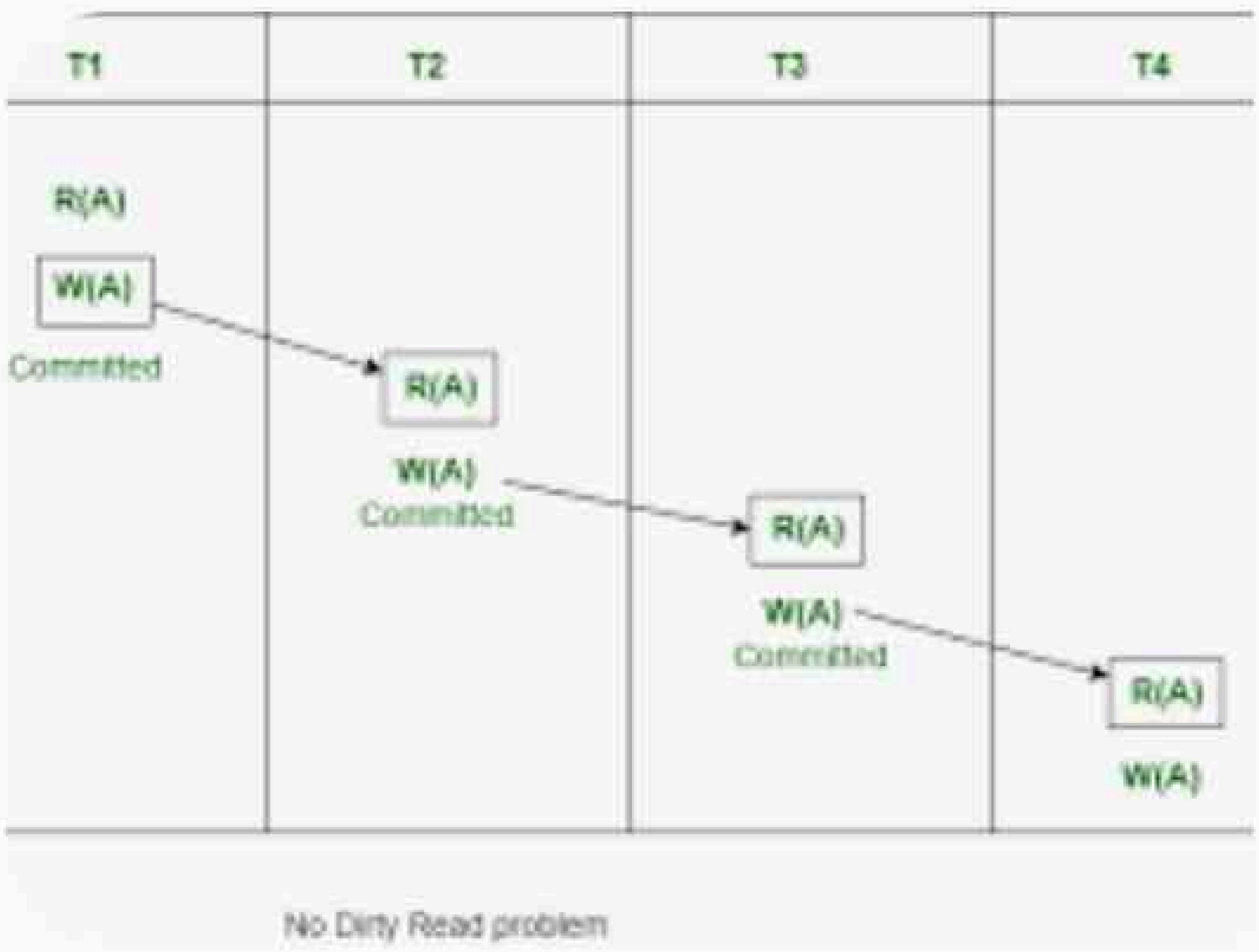
~~$\pi(G_{max})\ G_{min(T_1)}$~~

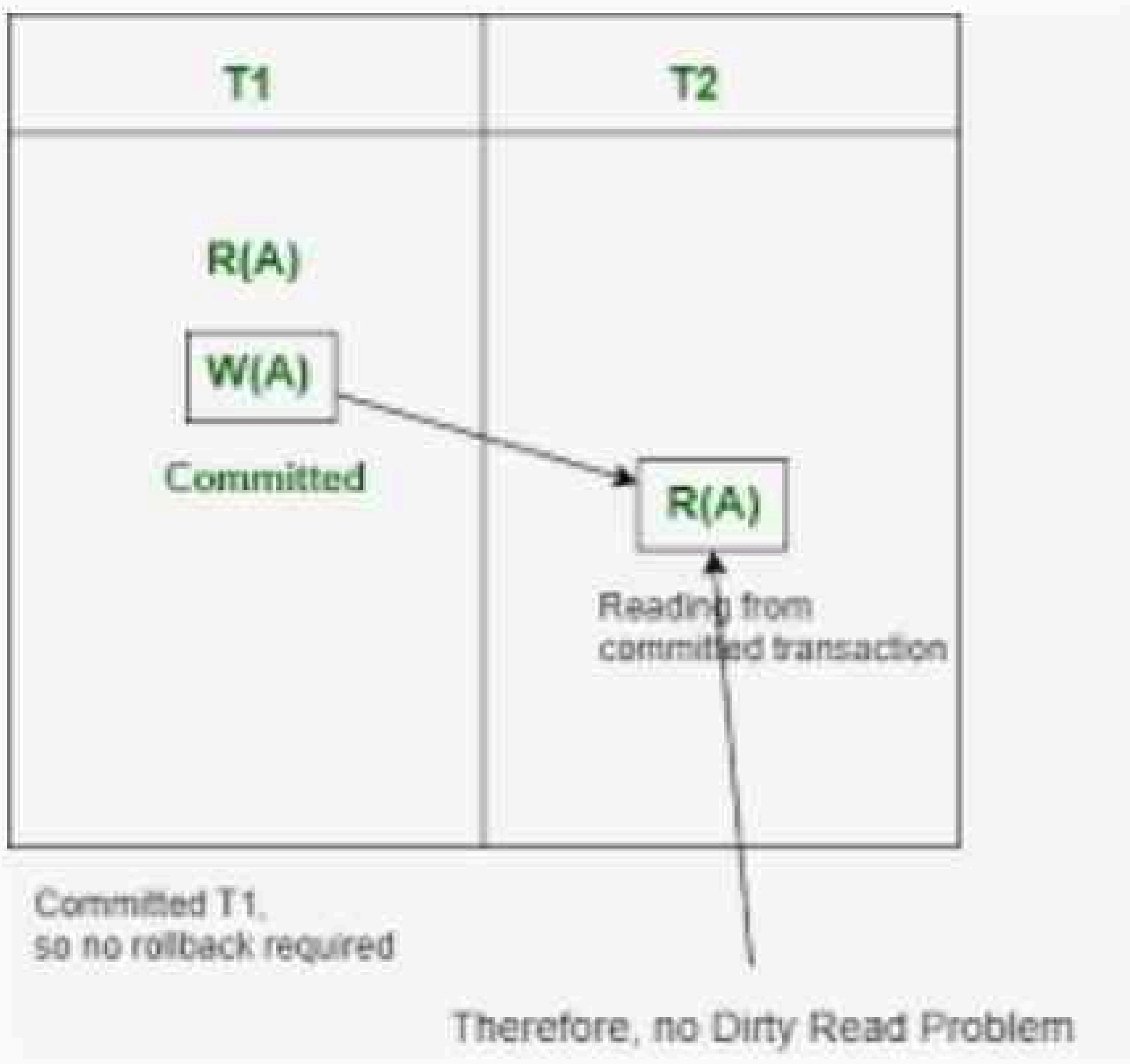
Showing results for *cascadeless* schedule in dbms

Search instead for casecadeless schedule in dbms

বাংলায় অনুসন্ধান করুন

ডিবিএমএসে ক্যাসকেডলেস সম.





বাংলায়

In English

Cascadeless Schedule avoids cascading aborts/rollbacks (ACA). **Schedules in which transactions read values only after all transactions whose changes they are going to read commit** are called cascadeless schedules. Avoids that a single transaction abort leads to a series of transaction rollbacks. Aug 12, 2019

https://www.geeksforgeeks.org › ca...

Cascadeless in DBMS - GeeksforGeeks

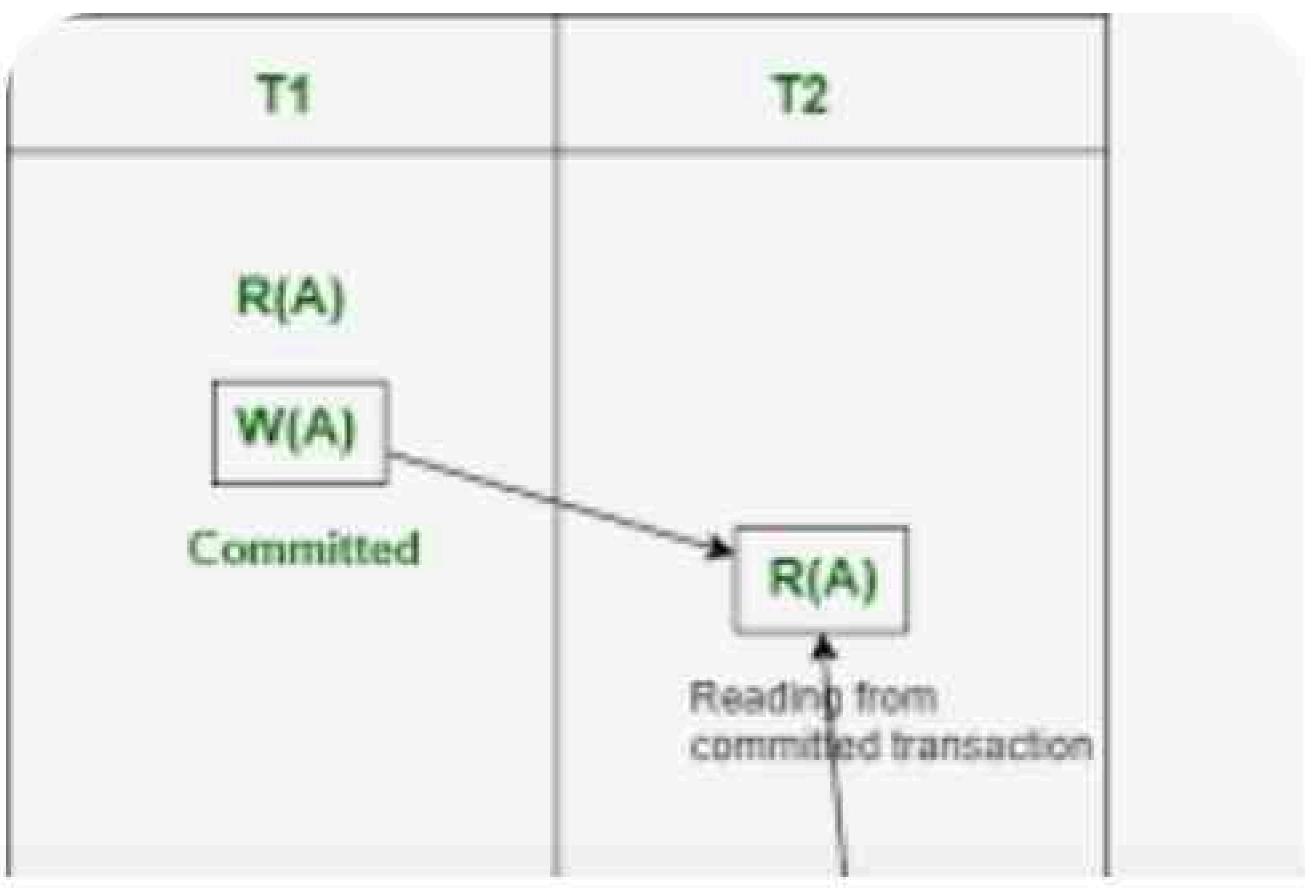
T1	T2	T3
R (A)		
W (A)		
Commit		
	R (A)	
	W (A)	
	Commit	
		R (A)
		W (A)
		Commit

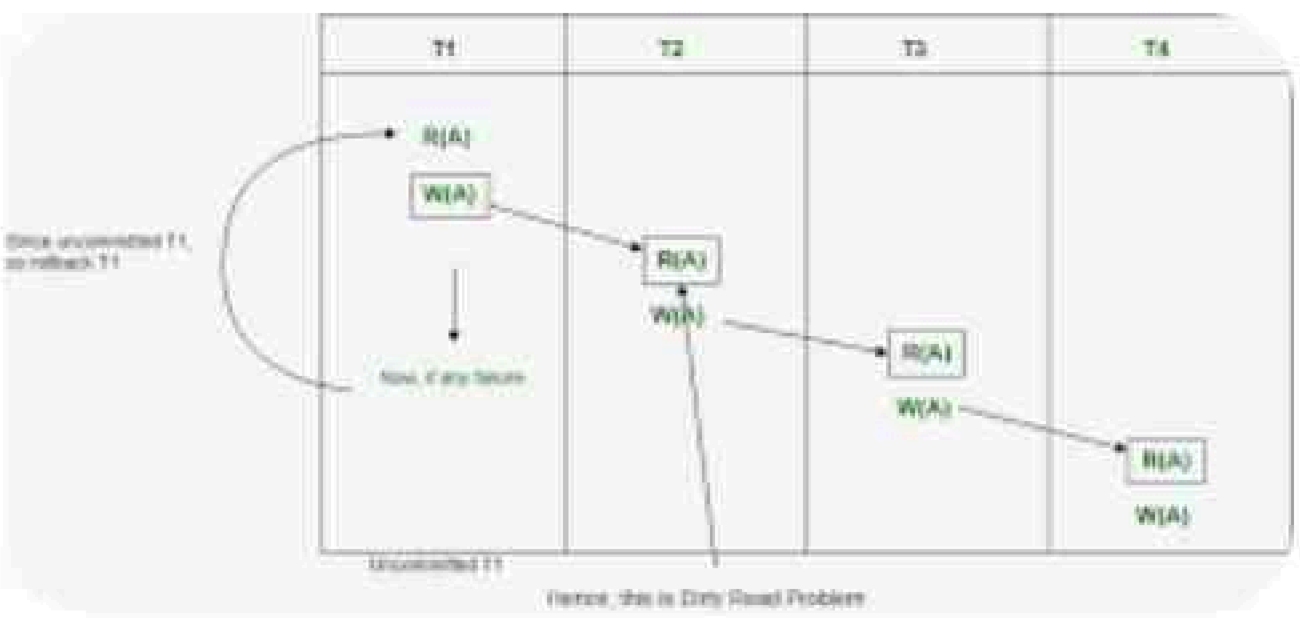
Cascadeless Schedule

Cascading Rollback | Cascadeless Schedule | Gate...

Visit

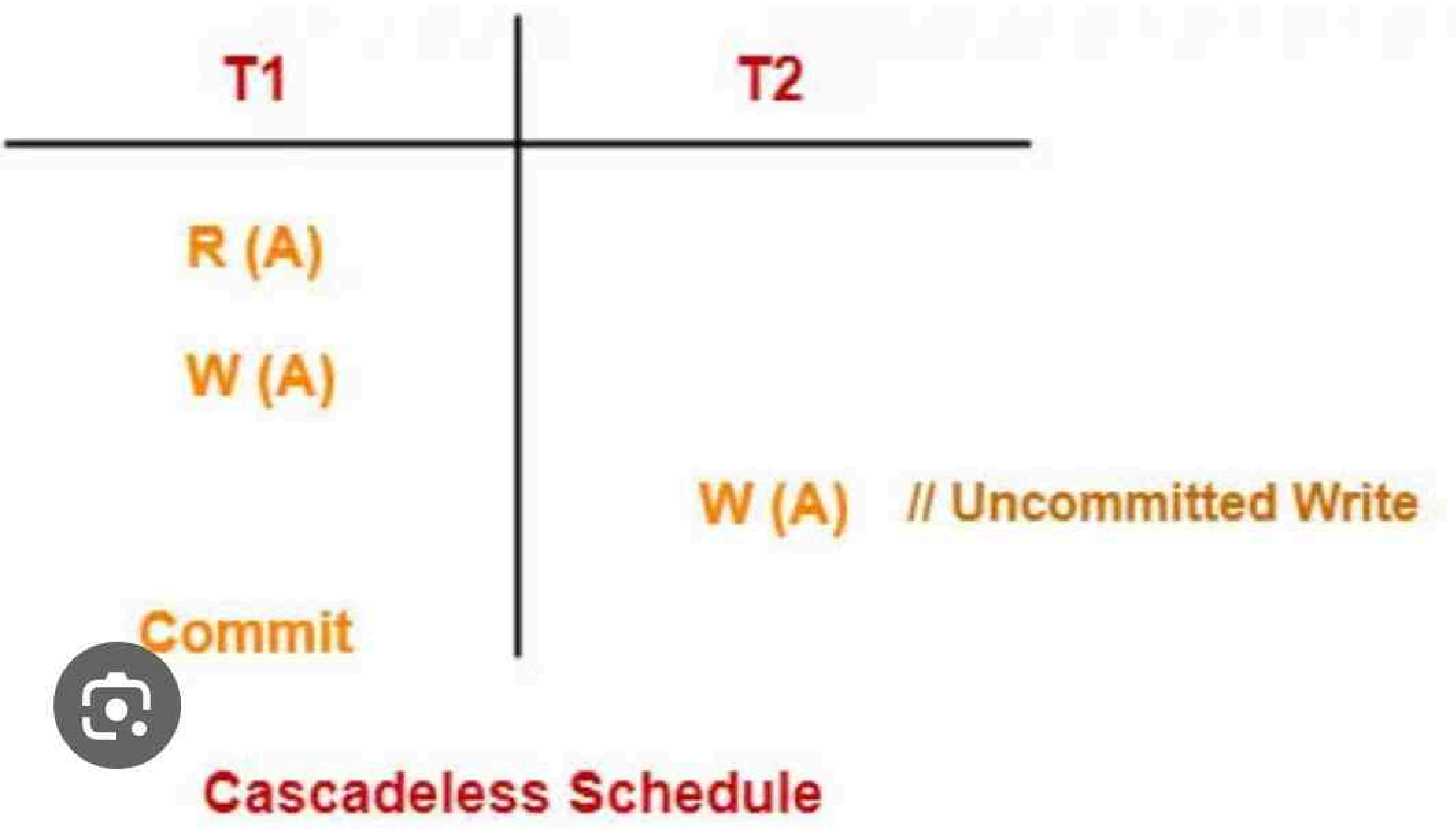
Share Save





GeeksforGeeks
Cascadeless in DBMS - G

Gate Vidyalay

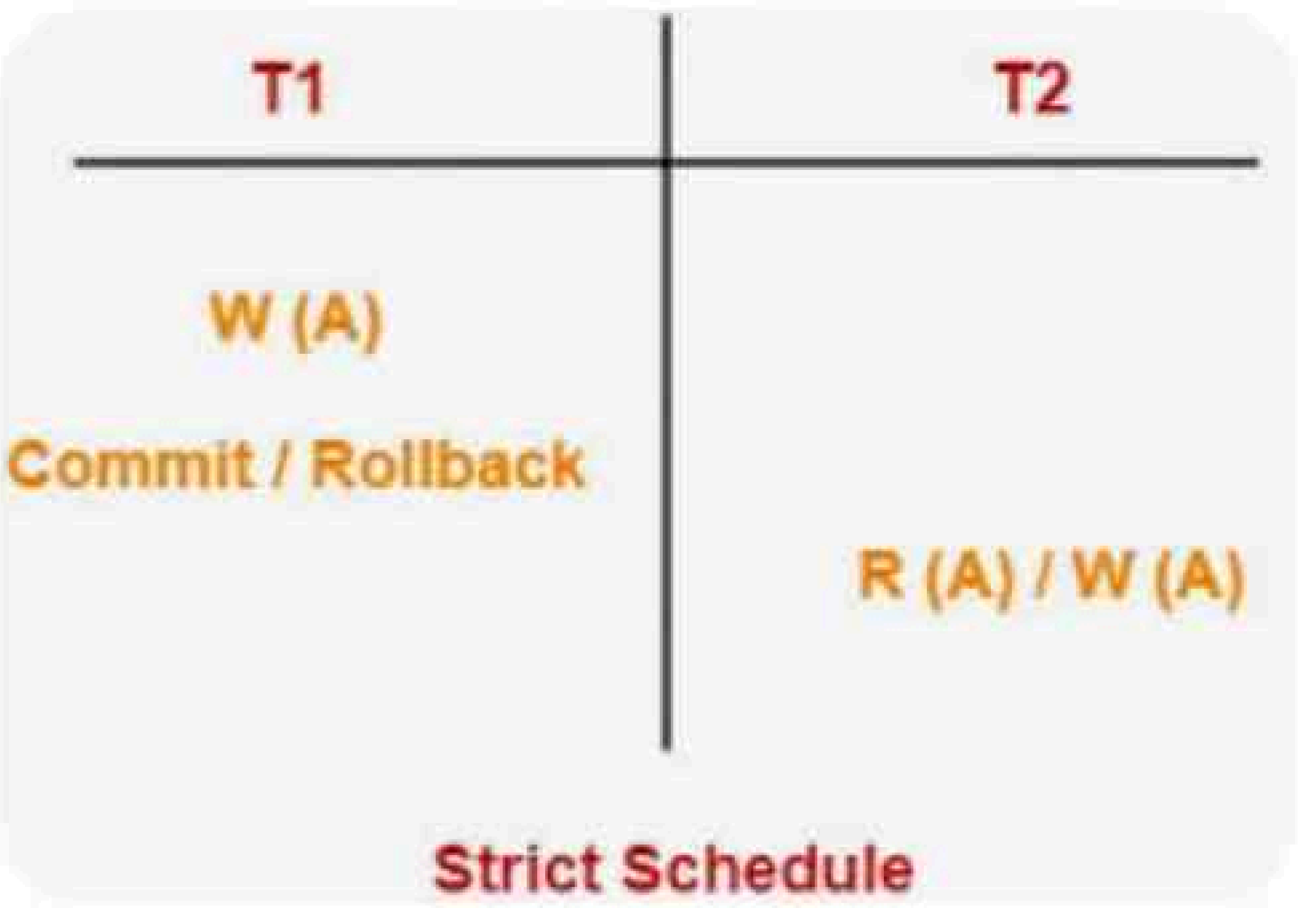


Cascading Rollback | Cascadeless Schedule | Gate...

Visit >

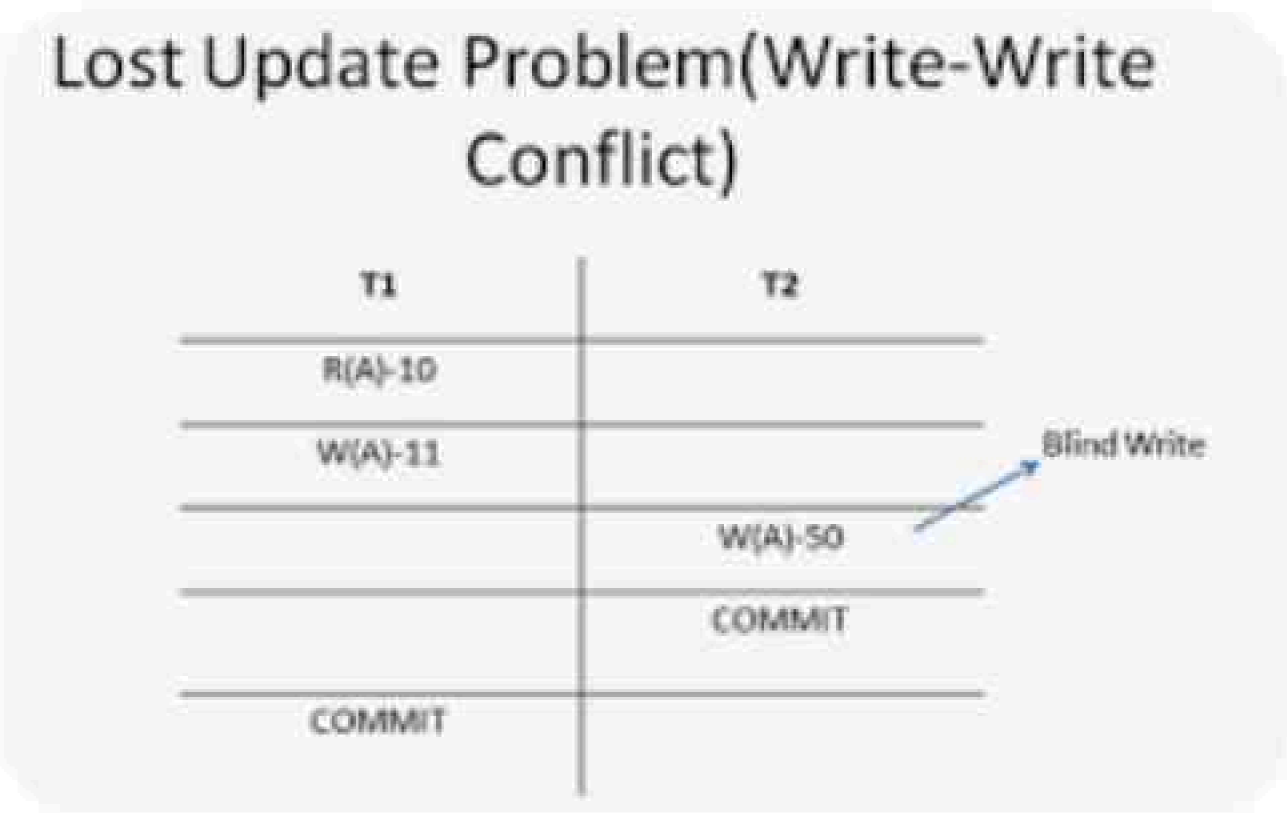
Share

Save



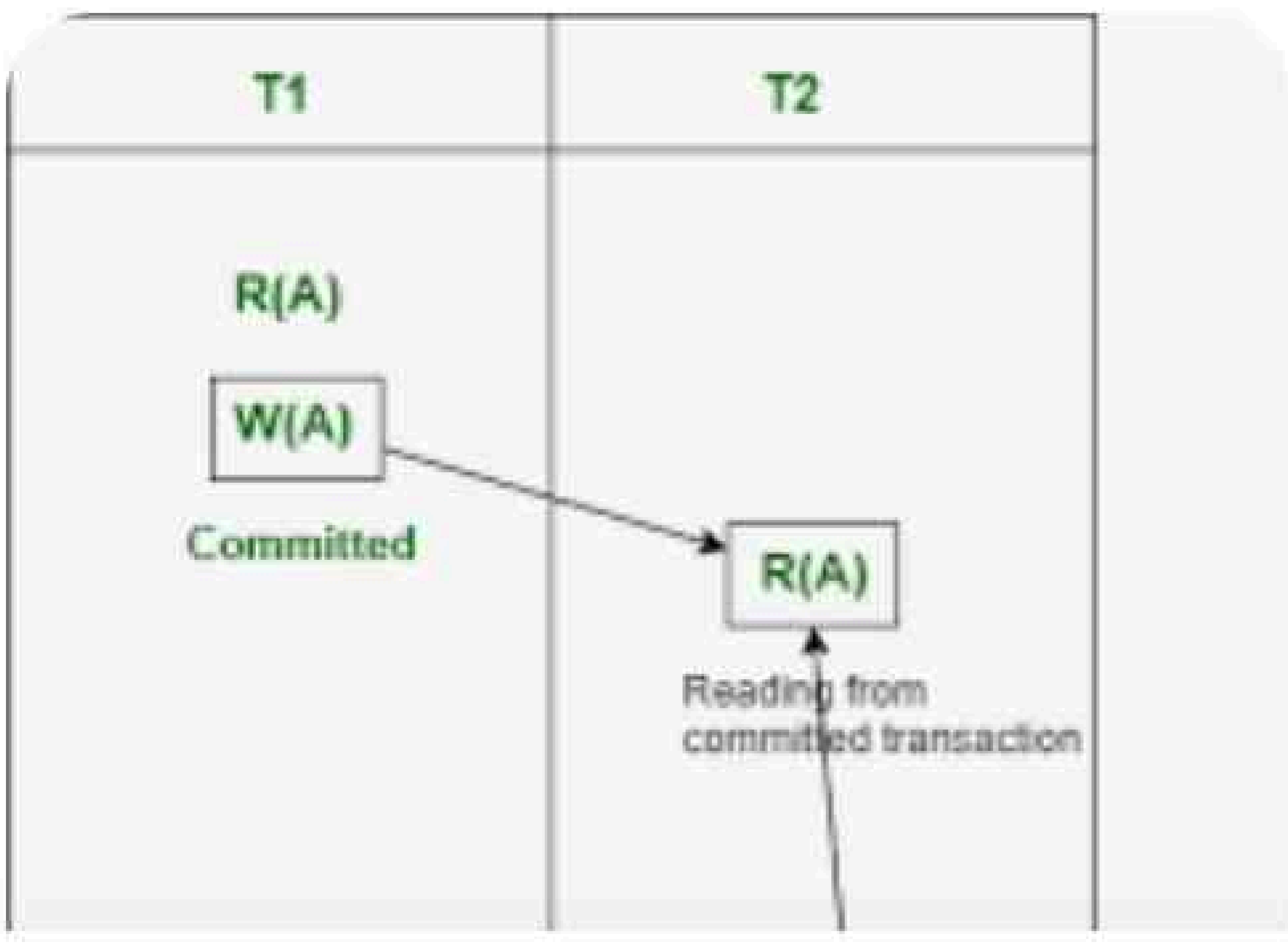
Gate Vidyalay

Cascading Rollback | Cas...



Nandita Hans - Medium

Concurrency in Transacti...



22:13 84

Google

casecading schedule in db...

All Images Videos News Shopping Books

Showing results for ***cascading*** schedule in dbms
Search instead for casecading schedule in dbms

বাঙলায় অনুসন্ধান করুন

dbms মধ্যে ক্যাসকেডিং সময়সূচি

বাংলায় In English

A cascading schedule is classified as a recoverable schedule. A recoverable schedule is basically a schedule in which the commit operation of a particular transaction that performs read operation is delayed until the uncommitted transaction either commits or roll backs. Dec 22, 2022

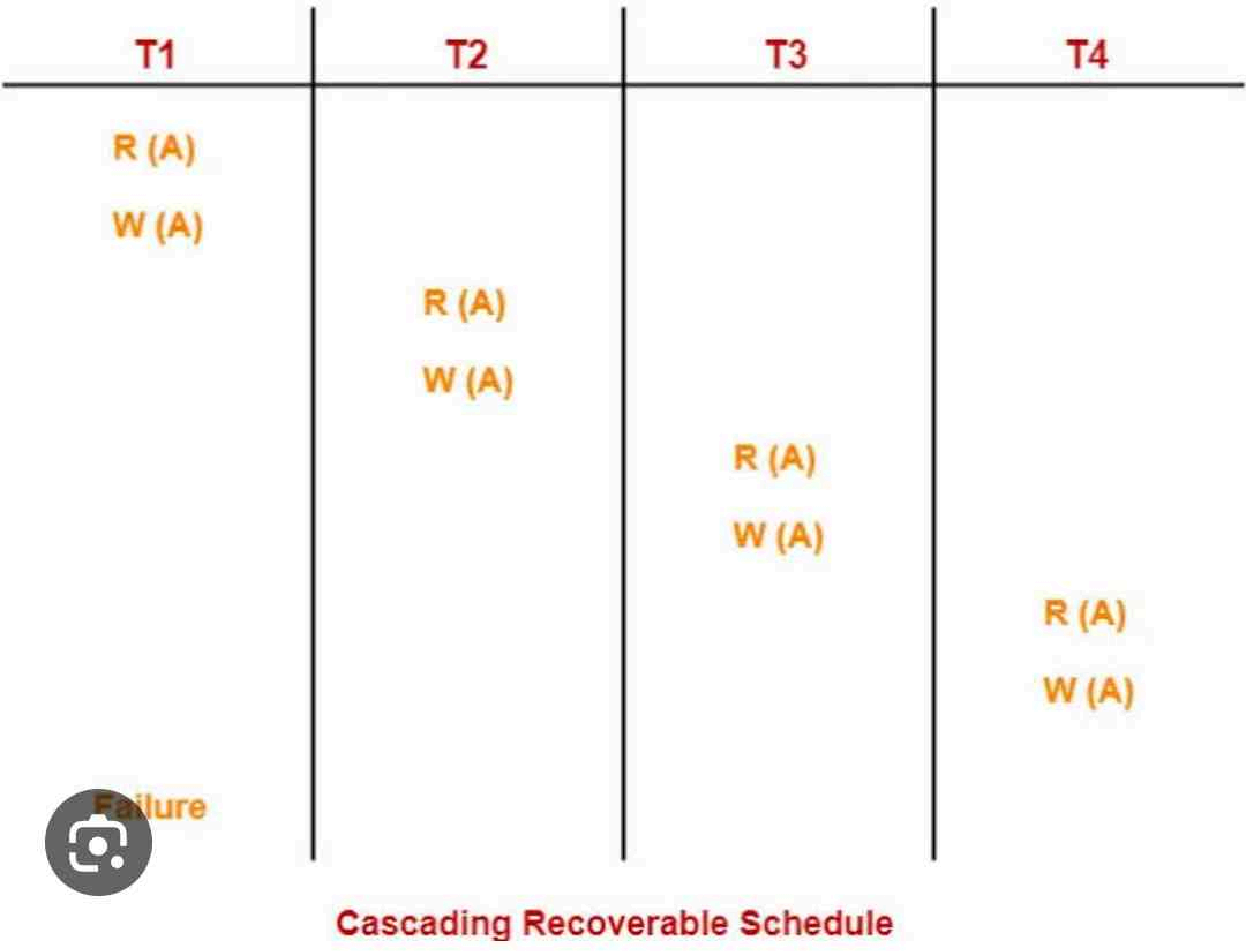
https://www.tutorialspoint.com › wh...

What are different types of recoverability of schedules(DBMS)? - Tutorialspoint

Discover Search Saved

22:13 ··· 84

Gate Vidyalay

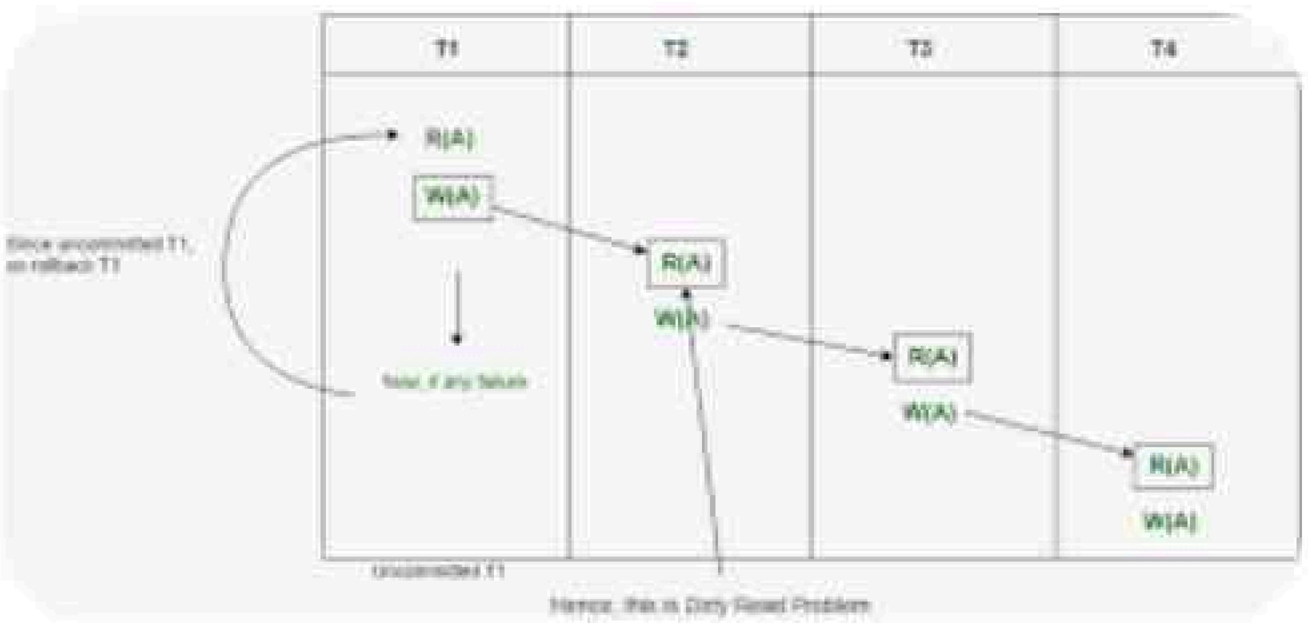


Cascading Rollback | Cascadeless Schedule | Gate...

Visit >

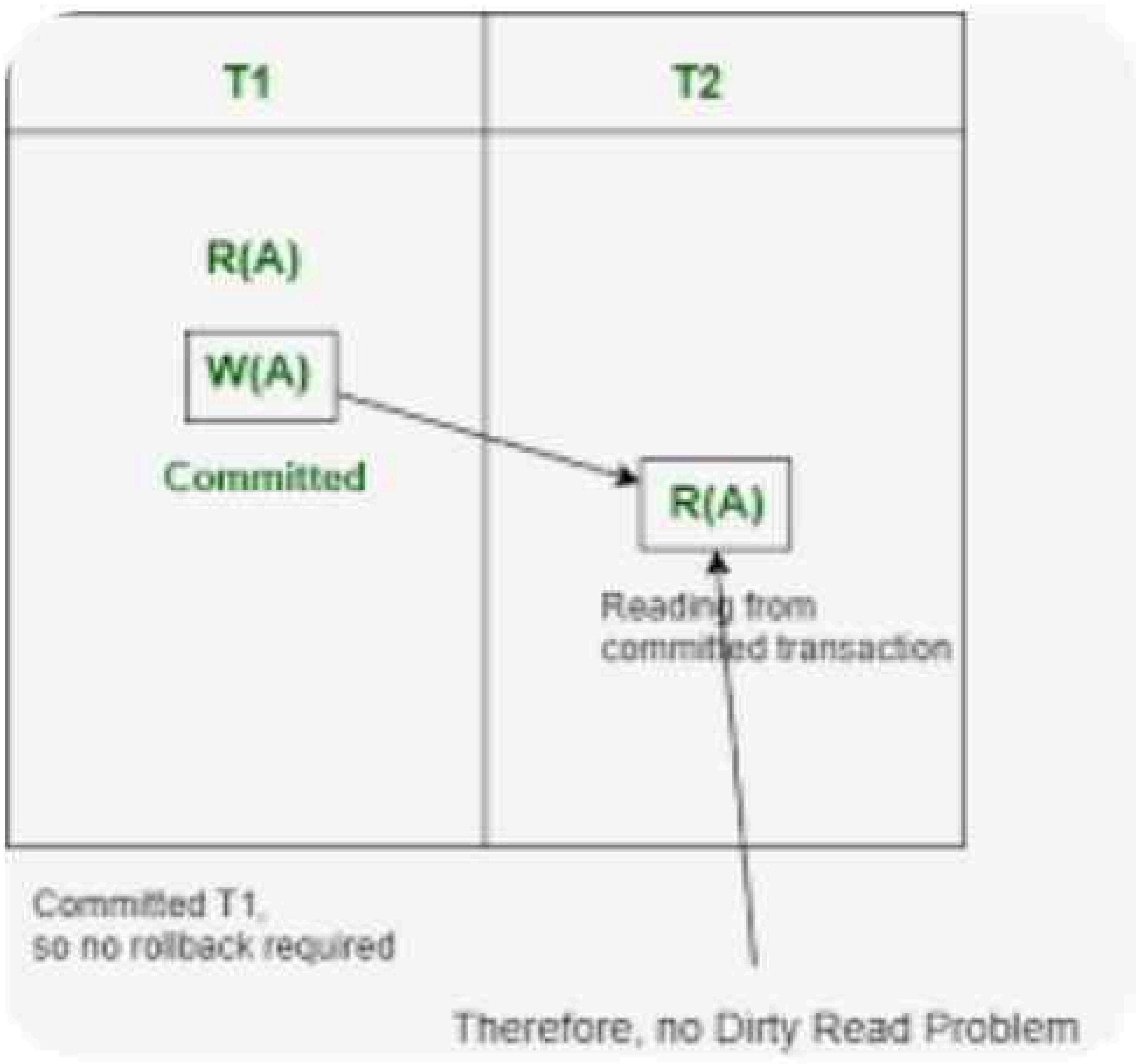
Share

Save



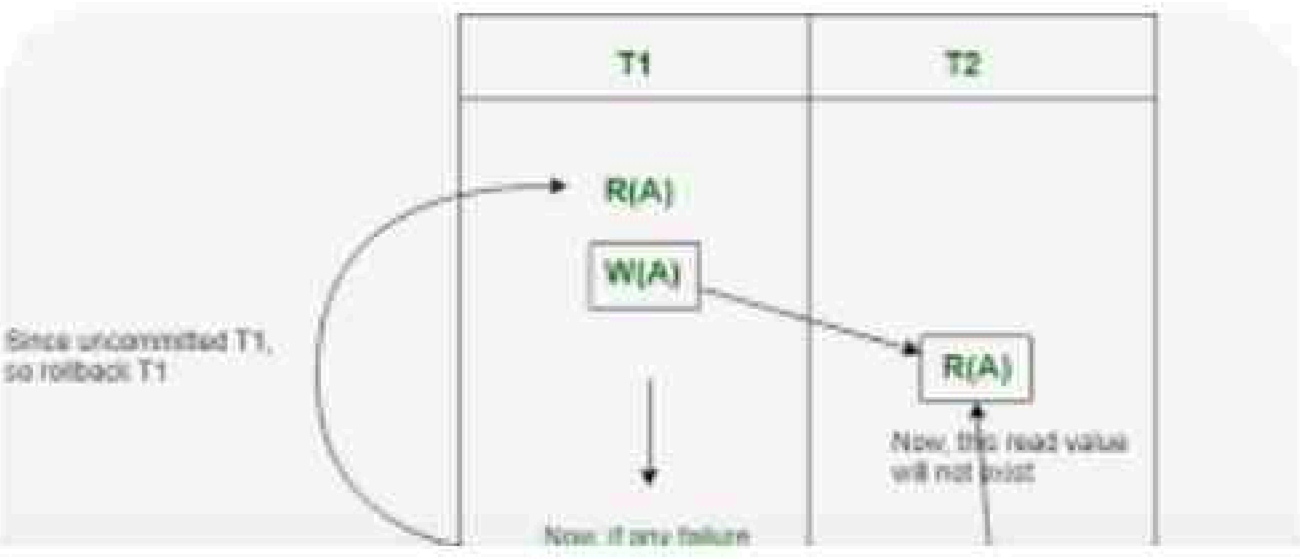
GeeksforGeeks

Cascadeless in DBMS - G...



GeeksforGeeks

Cascadeless in DBMS - G...



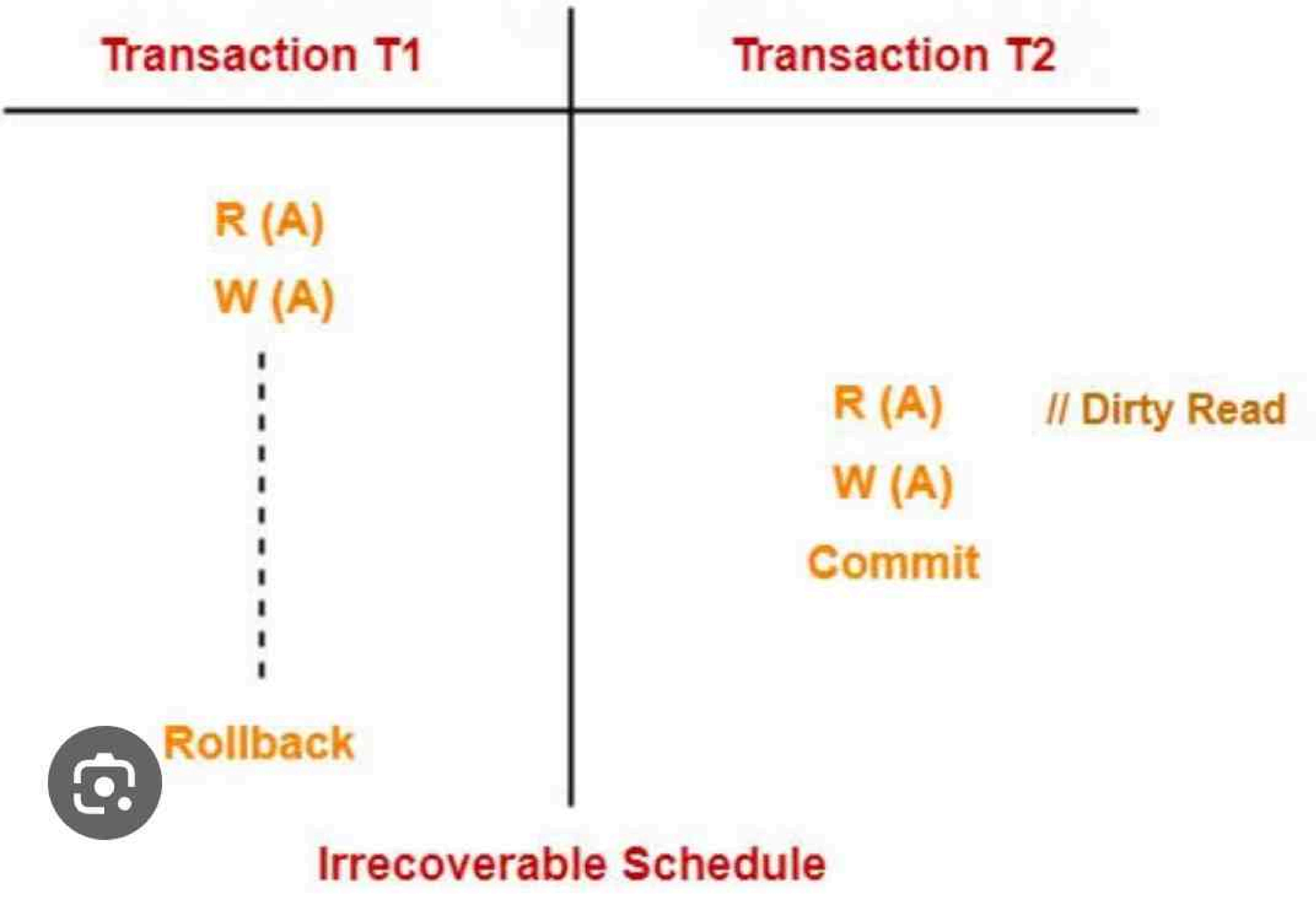
Discover

Search

Saved

22:13

Gate Vidyalay

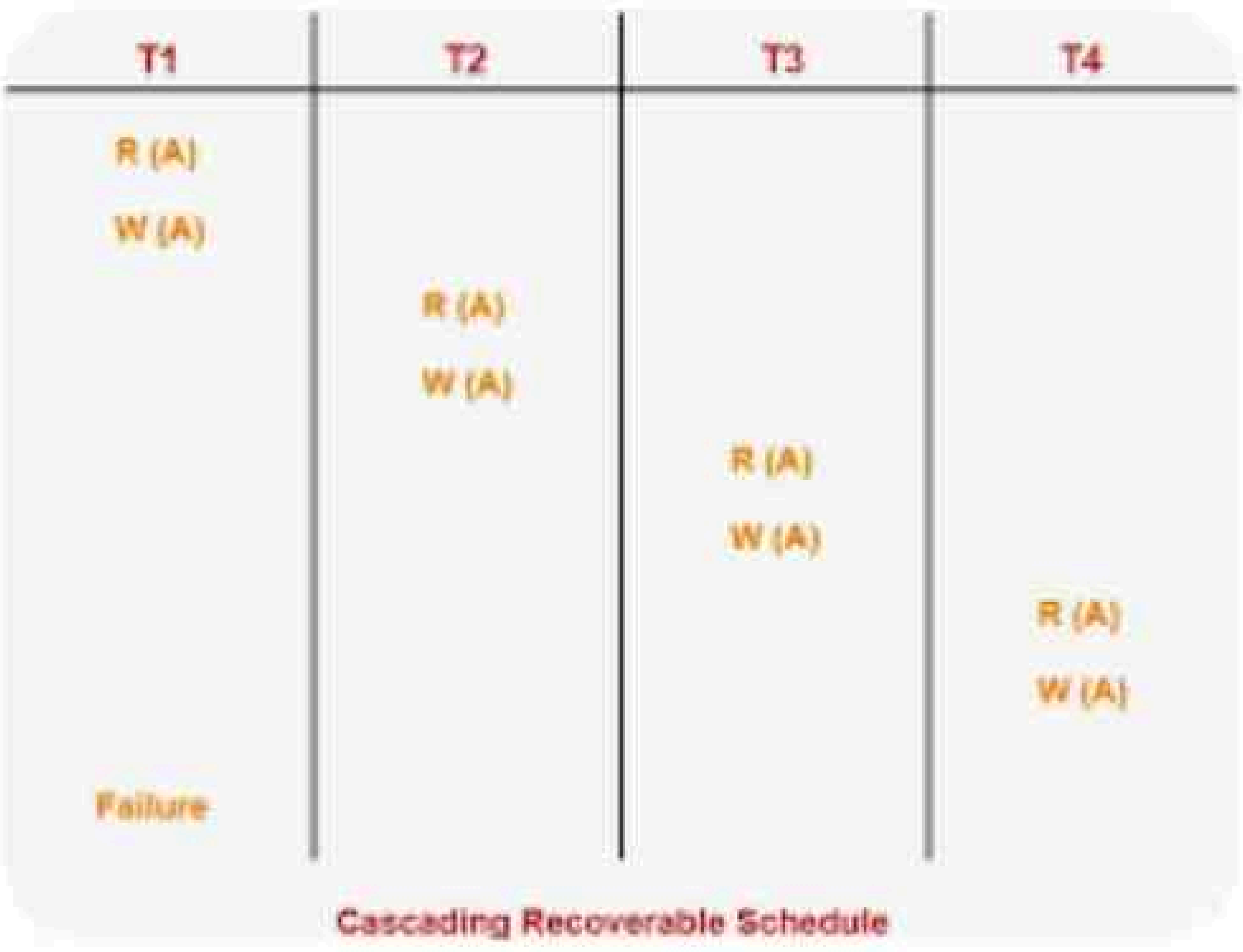


Schedules and Recoverability in DBMS | Gate Vidyalay

Visit >

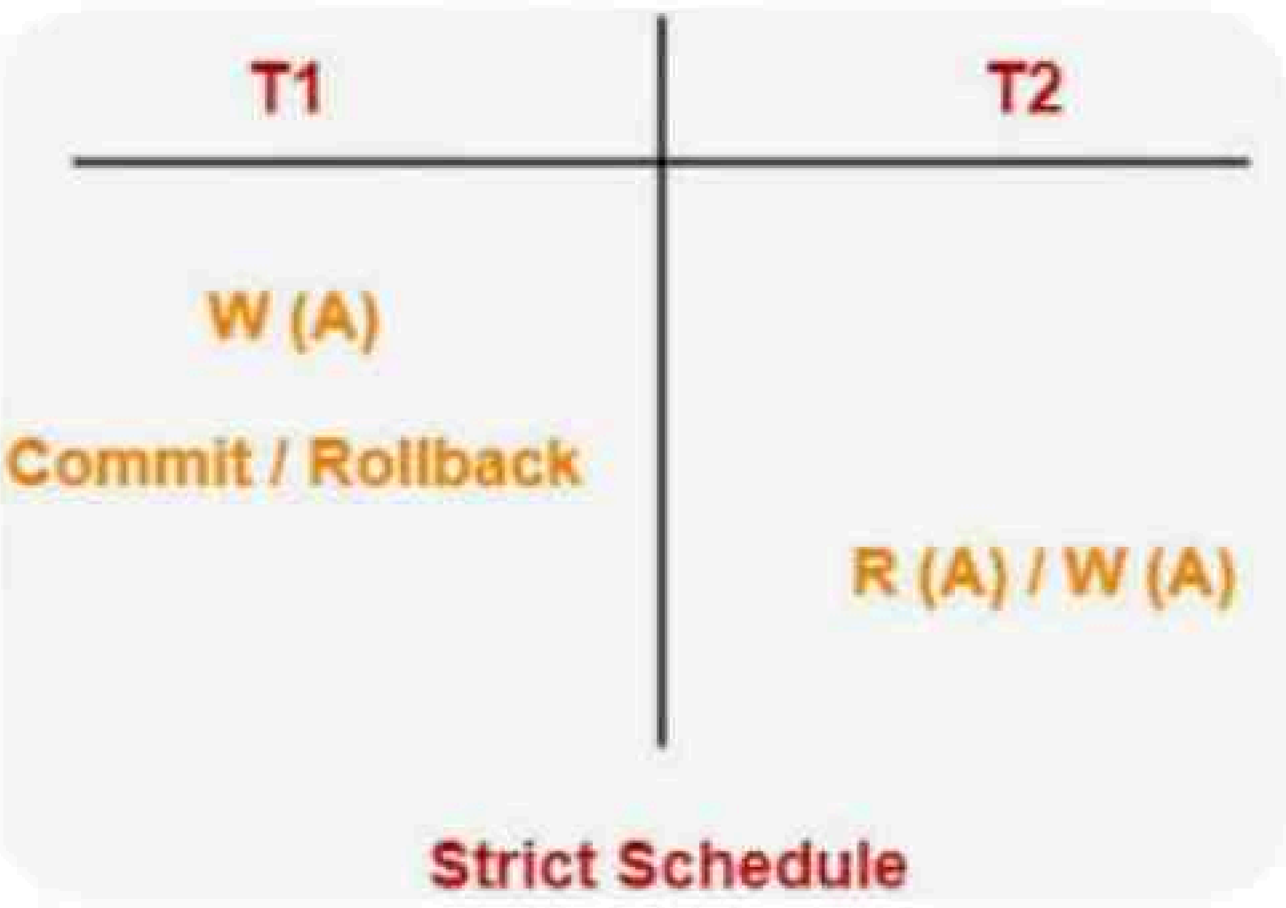
Share

Save



Gate Vidyalay

Schedules and Recovera...



Gate Vidyalay

cascadeless schedule | G...

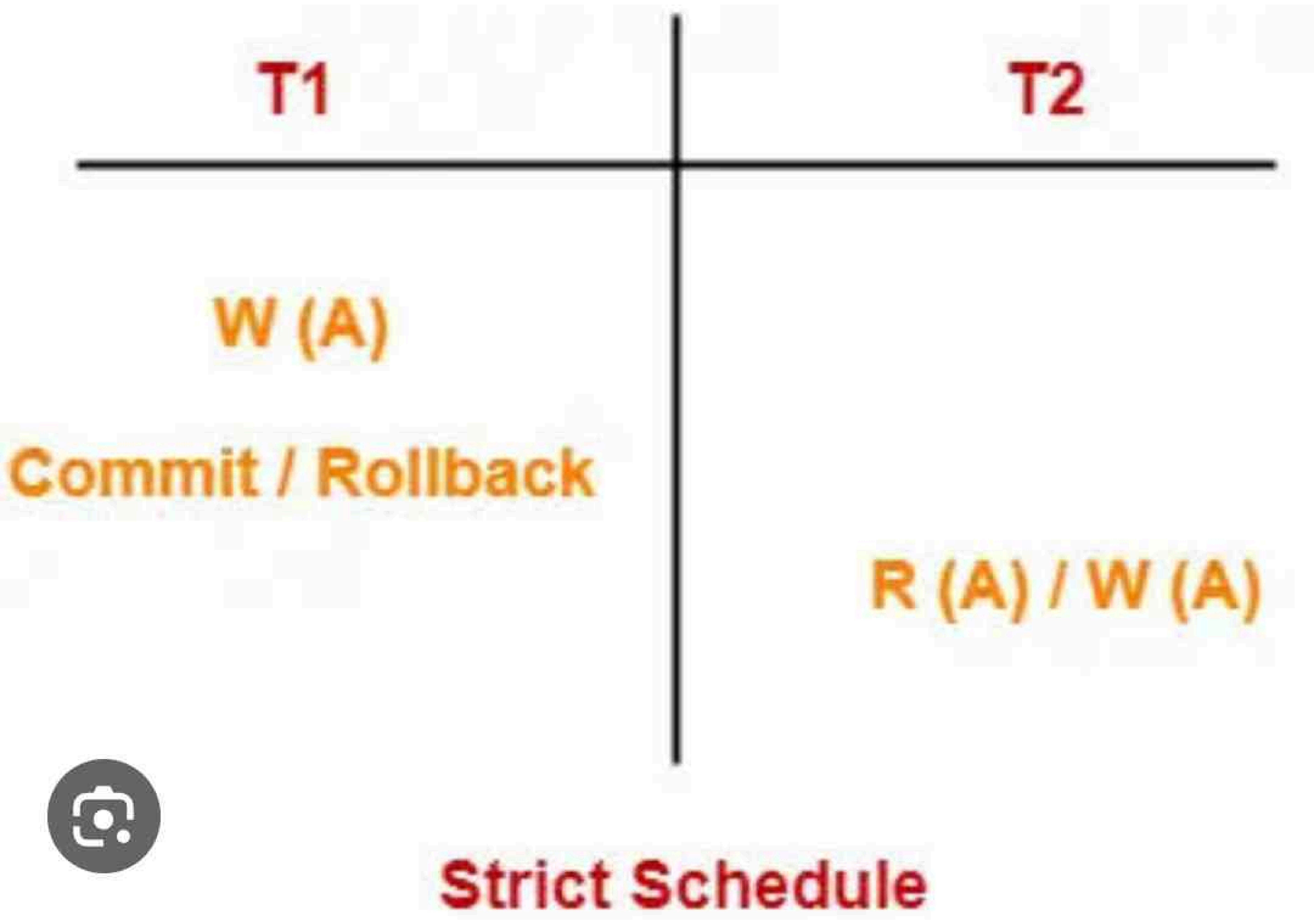
Discover

Search

Saved

22:13

Gate Vidyalay

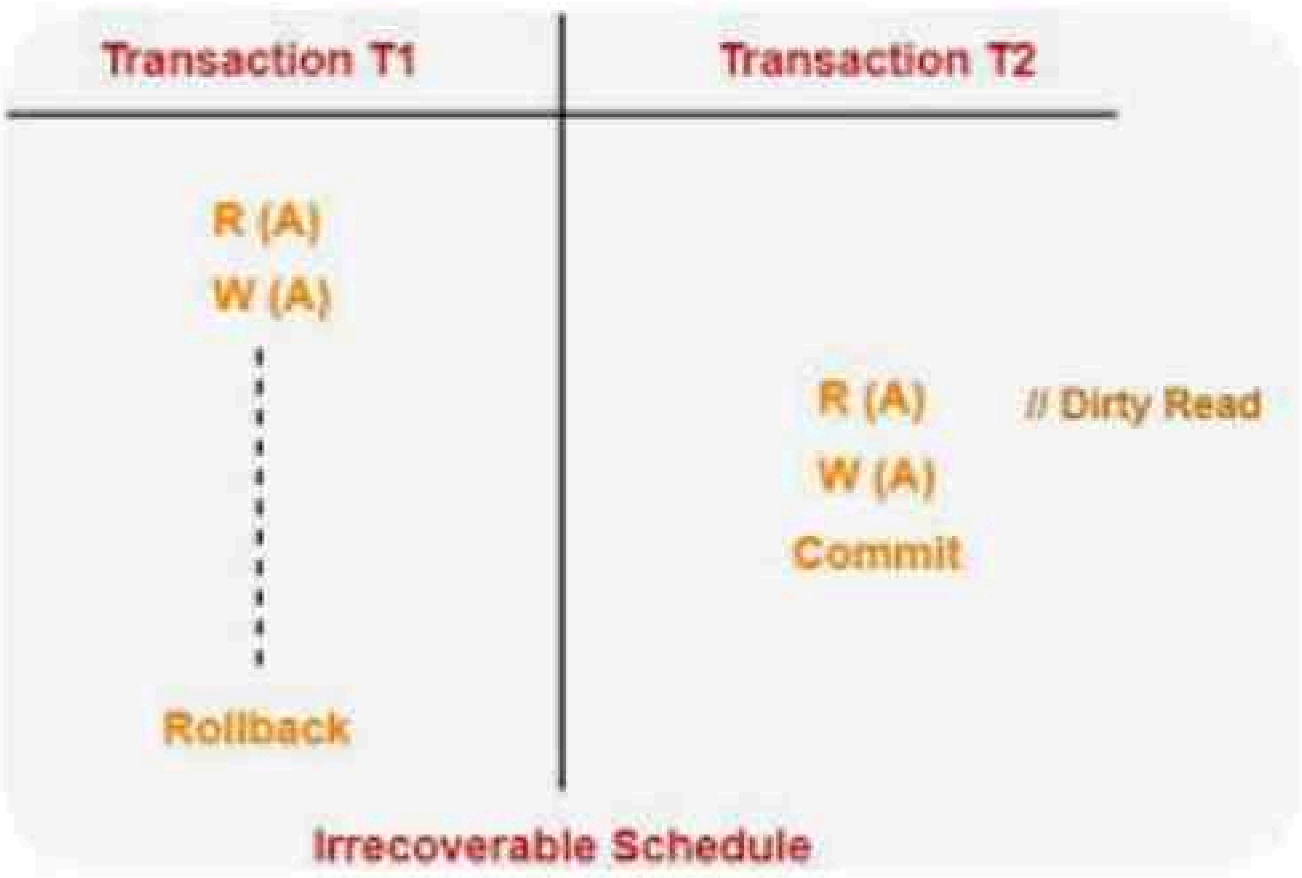


cascadeless schedule | Gate Vidyalay

Visit >

Share

Save

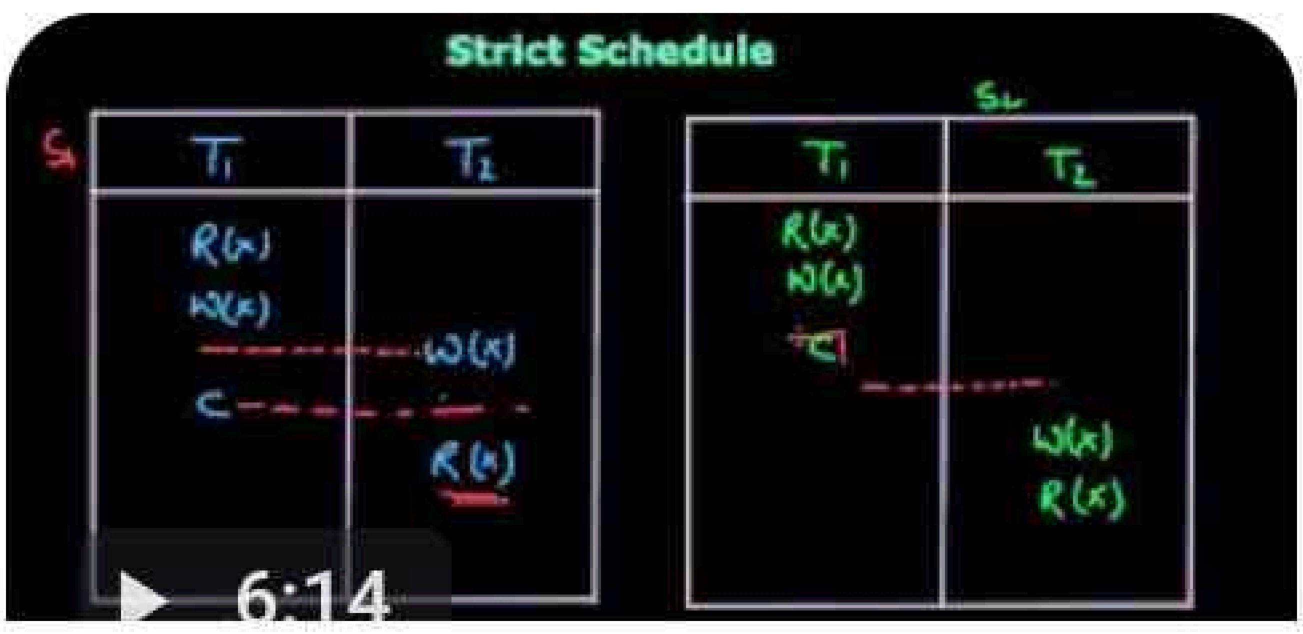


Gate Vidyalay

Schedules and Recovera...

Gate Vidyalay

cascadeless schedule | G...



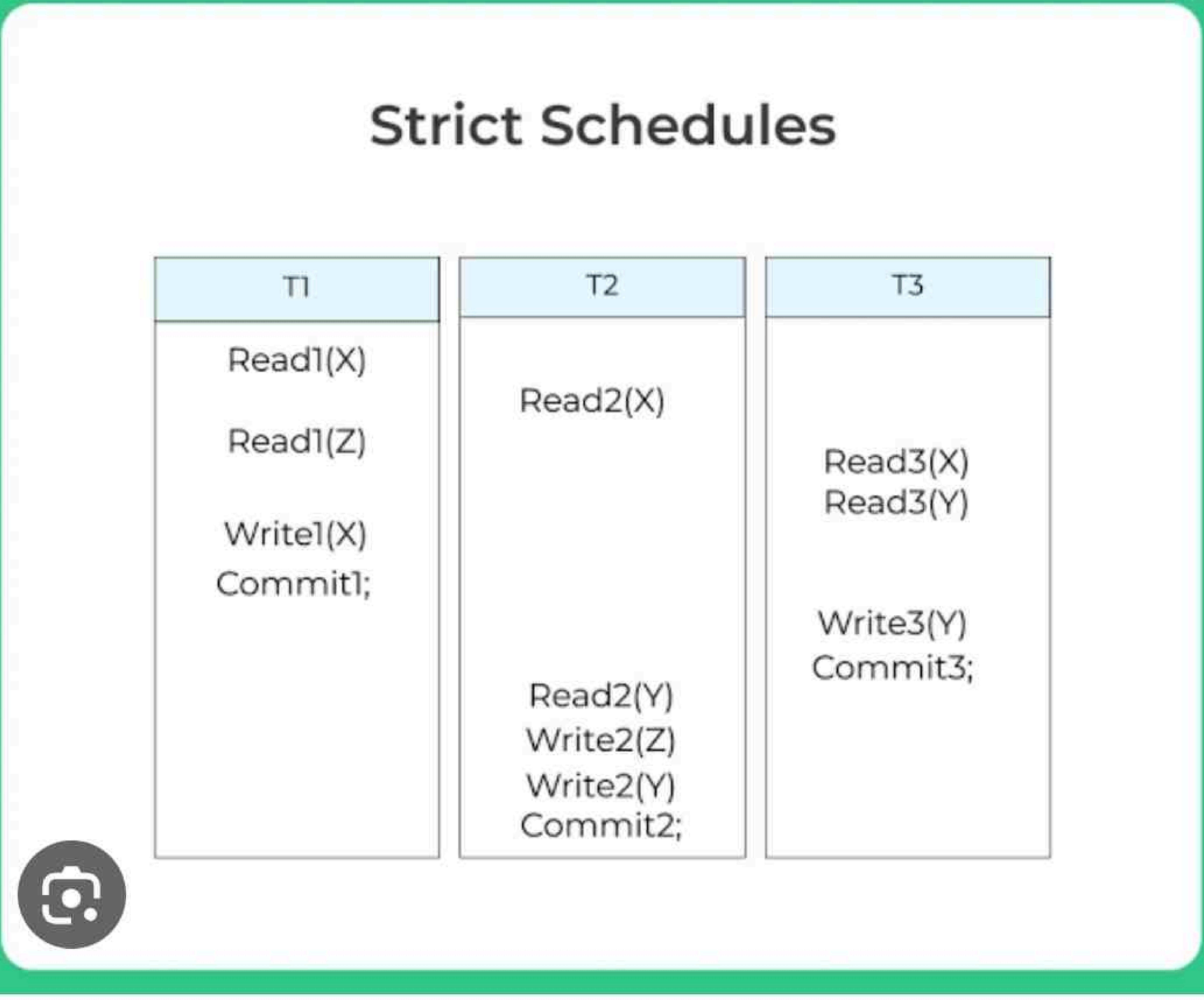
Discover

Search

Saved

22:13

PrepInsta



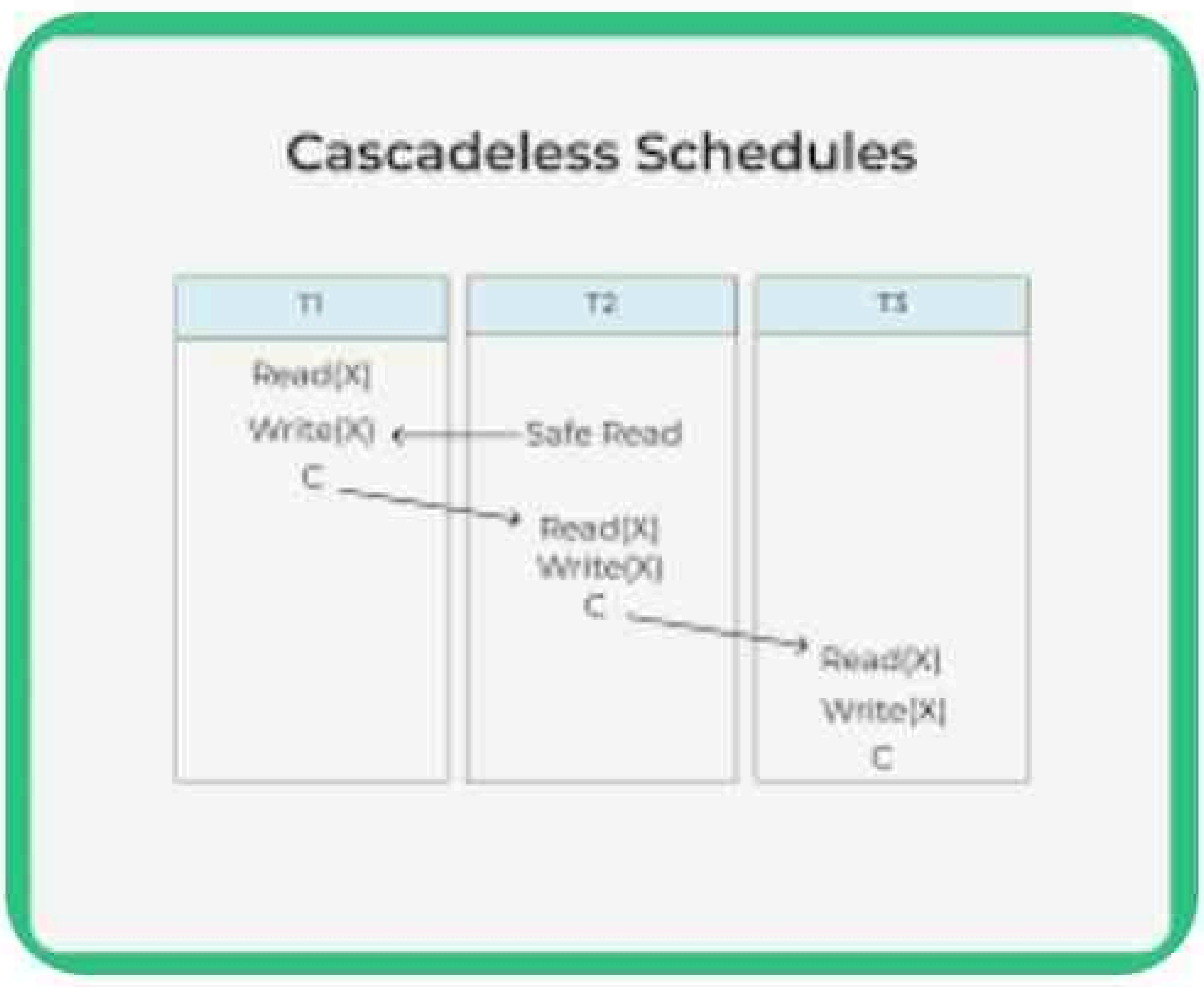
Recoverability of Schedules in DBMS | Database Managemen...

Visit

Images may be subject to copyright. **Learn more**

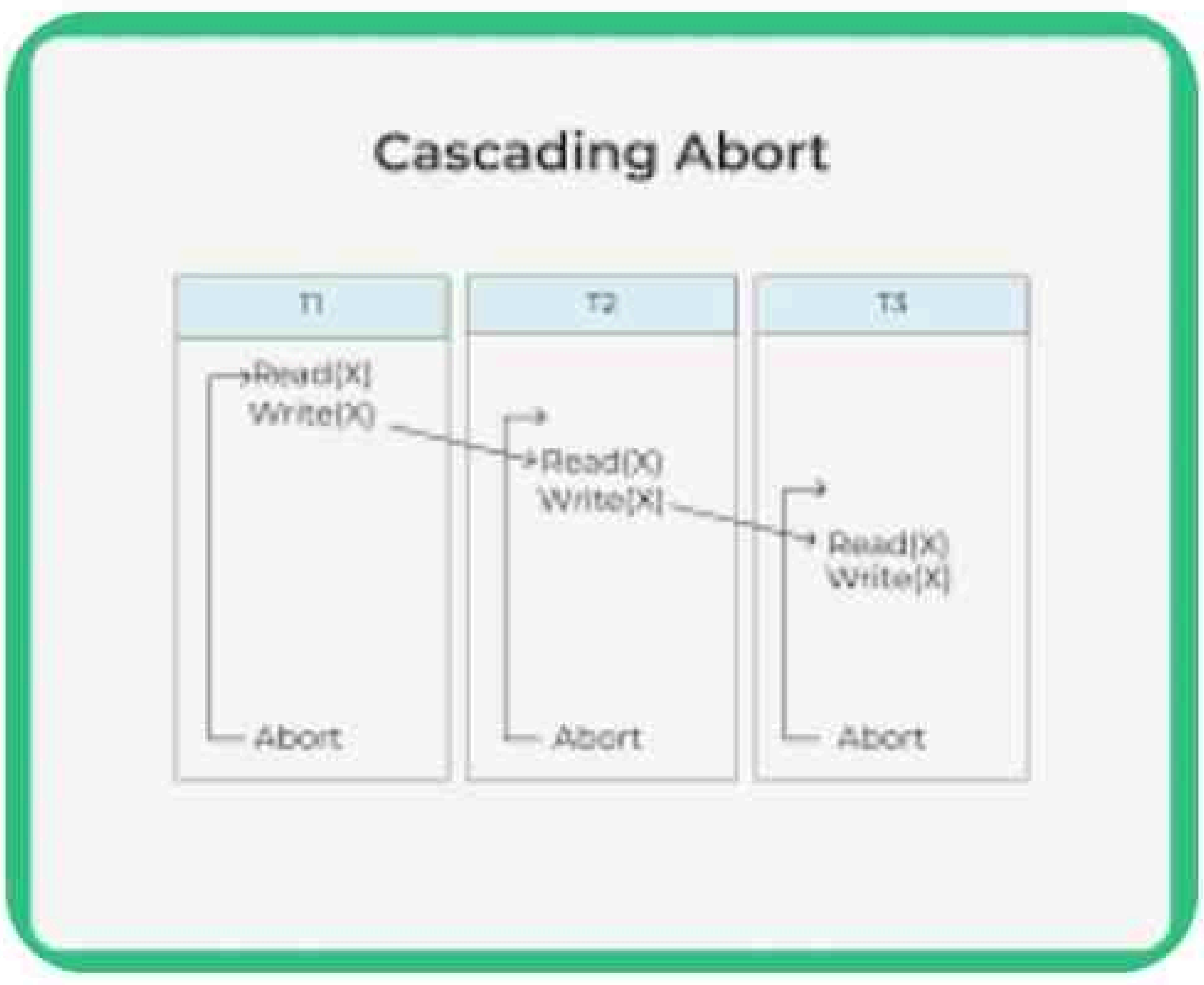
Share

Save



PrepInsta

Recoverability of Schedul...



PrepInsta

Recoverability of Schedul...

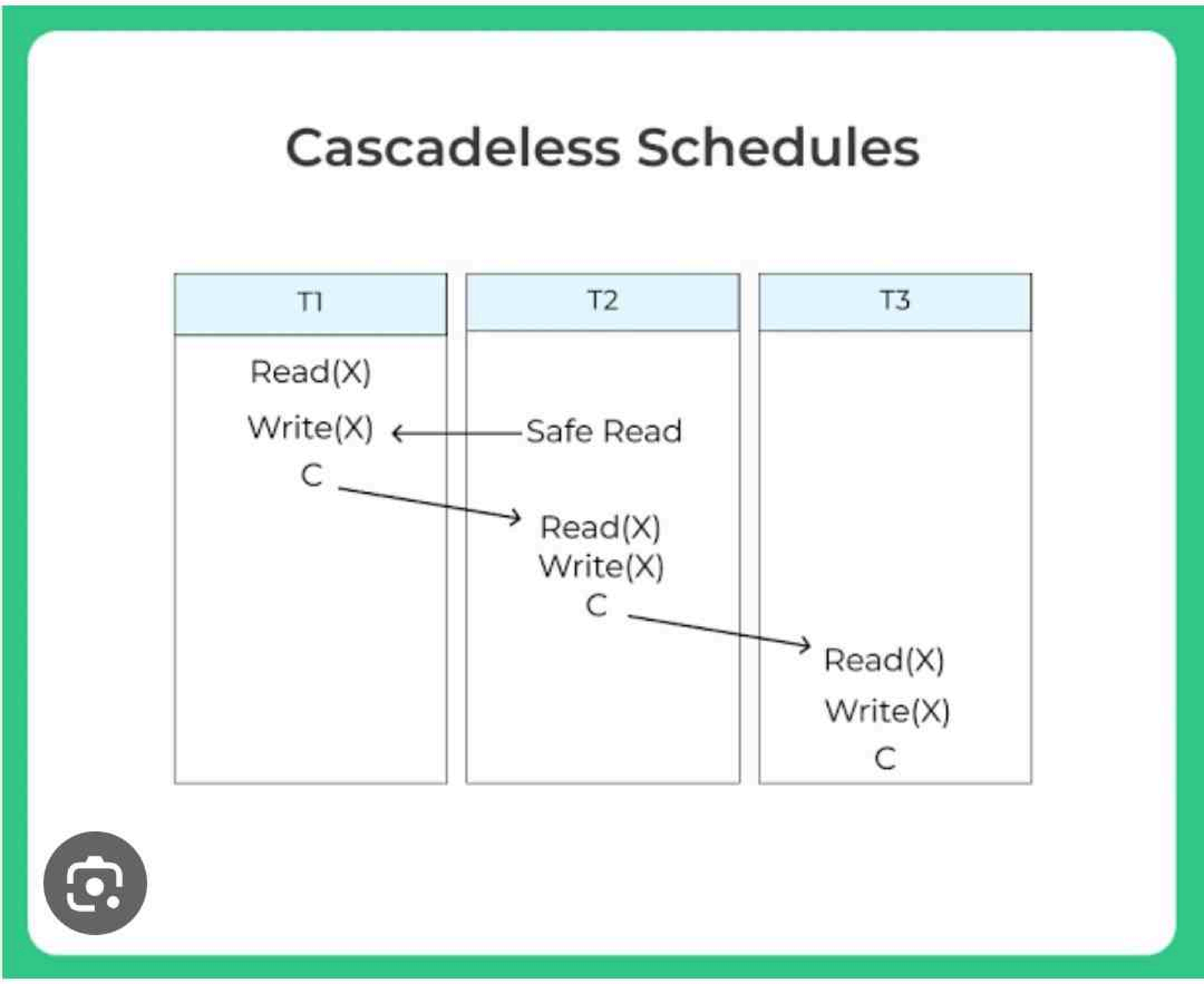
Discover

Search

Saved

22:14

PrepInsta

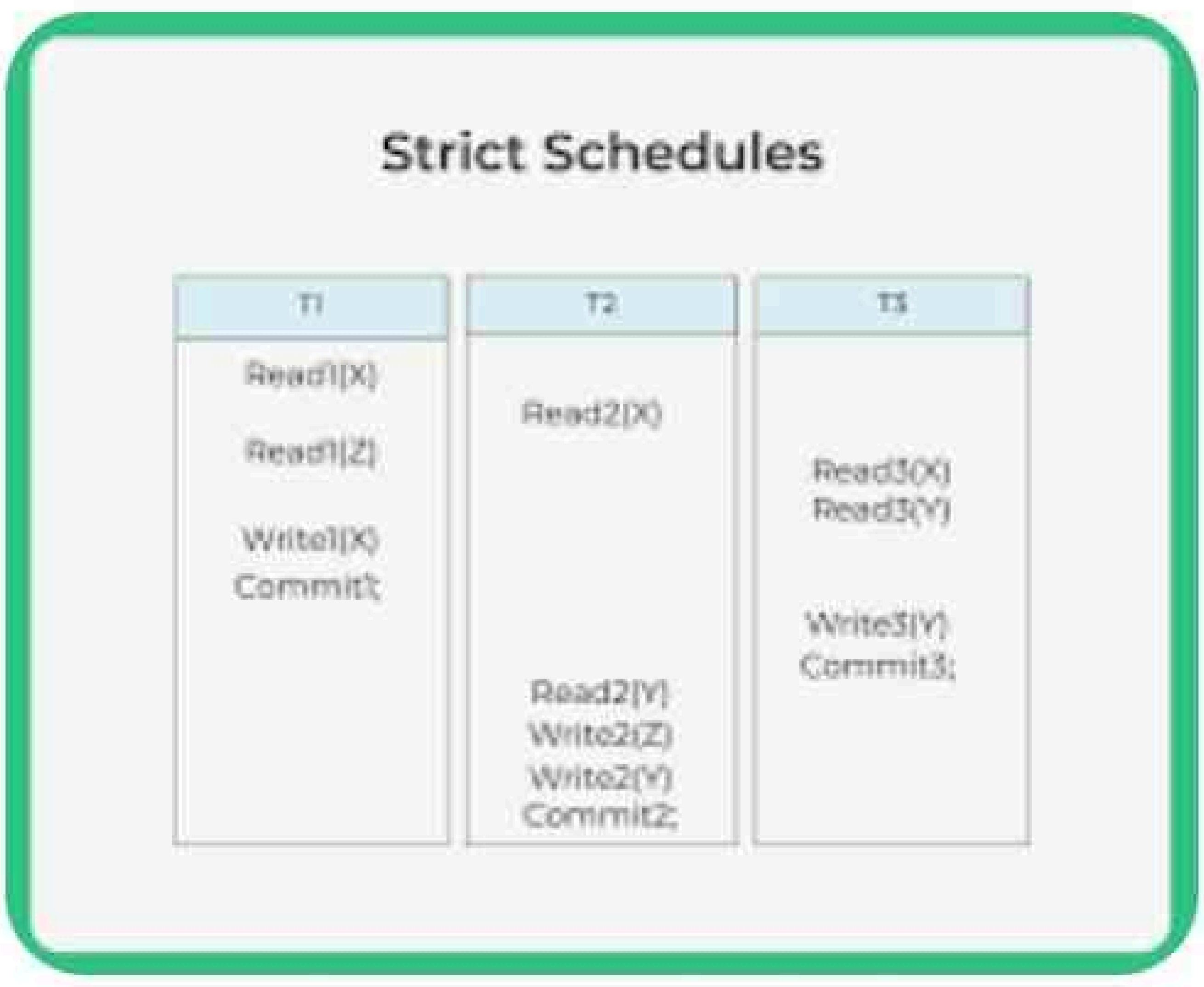


Recoverability of Schedules in DBMS | Database Managemen...

Visit >

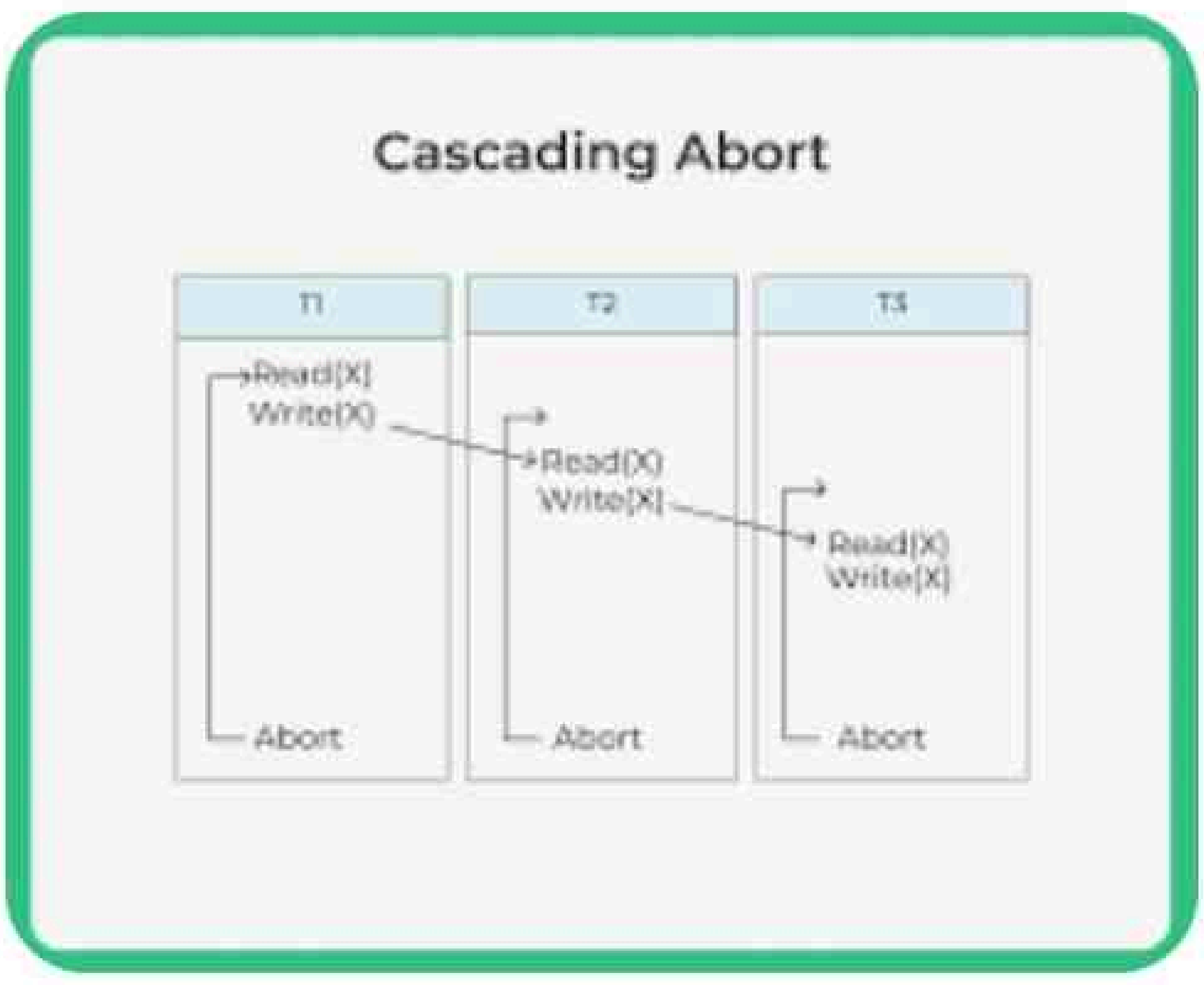
Share

Save



PrepInsta

Recoverability of Schedul...



PrepInsta

Recoverability of Schedul...

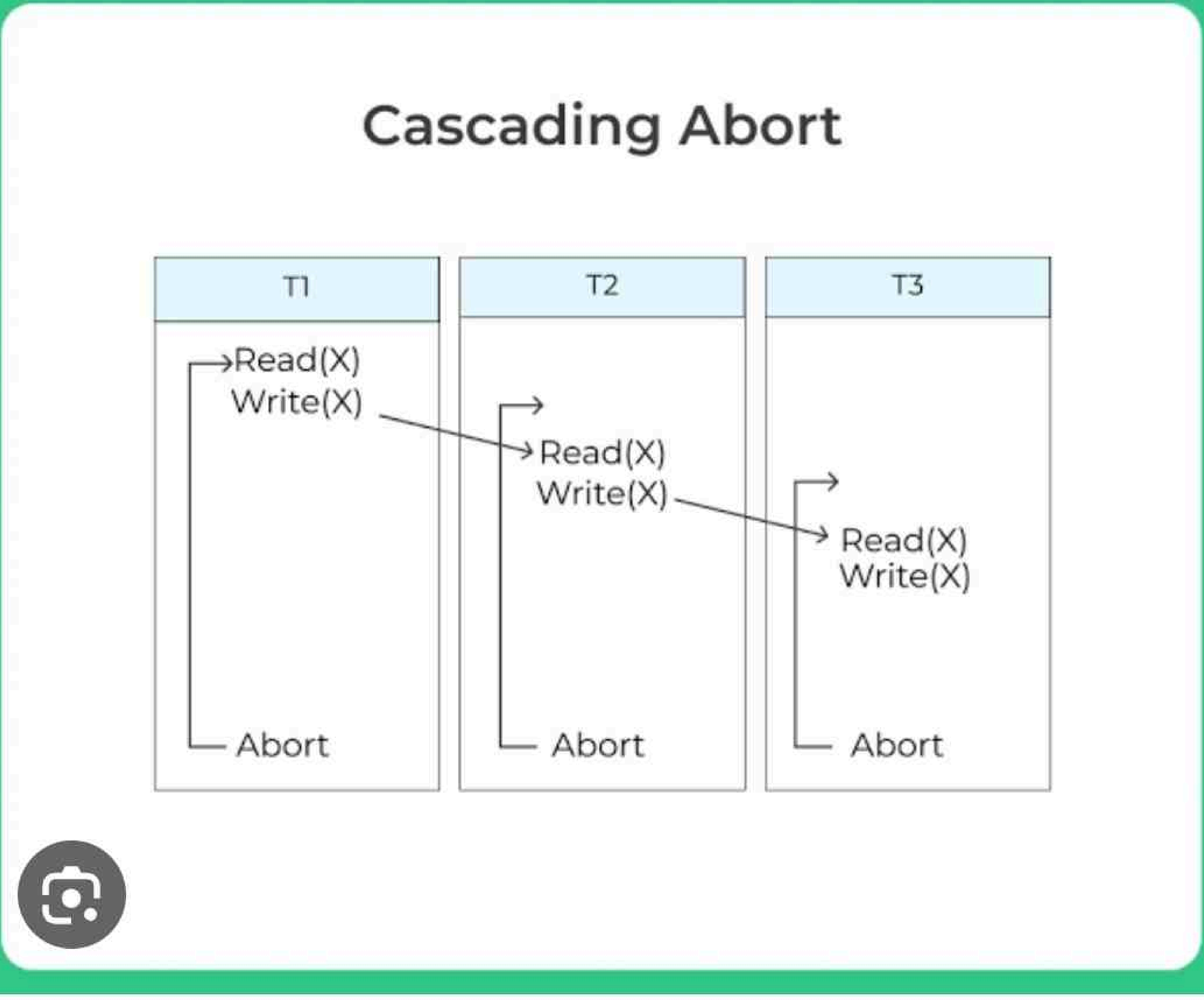
Discover

Search

Saved

22:14 84

PrepInsta

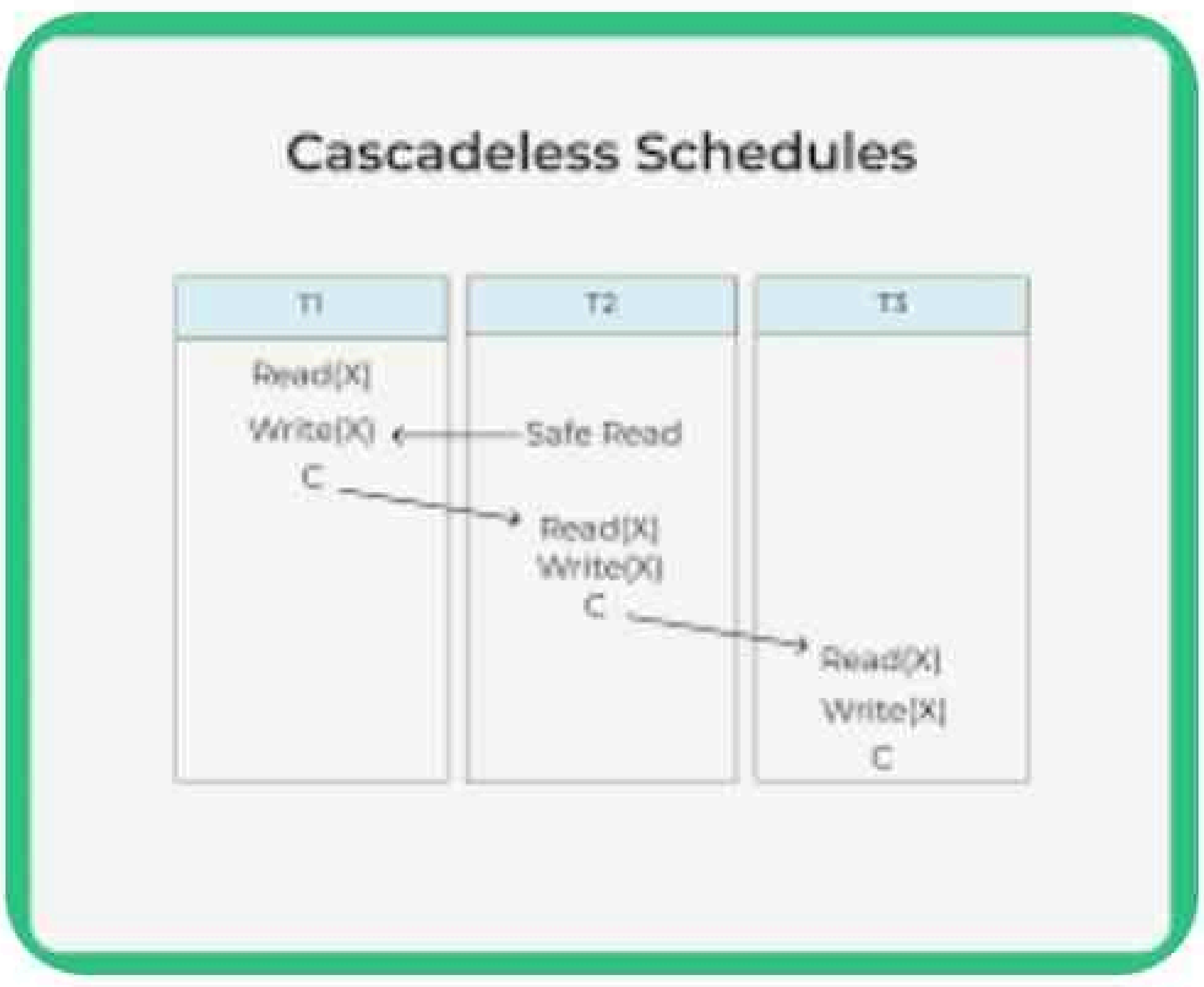


Recoverability of Schedules in DBMS | Database Managemen...

Visit >

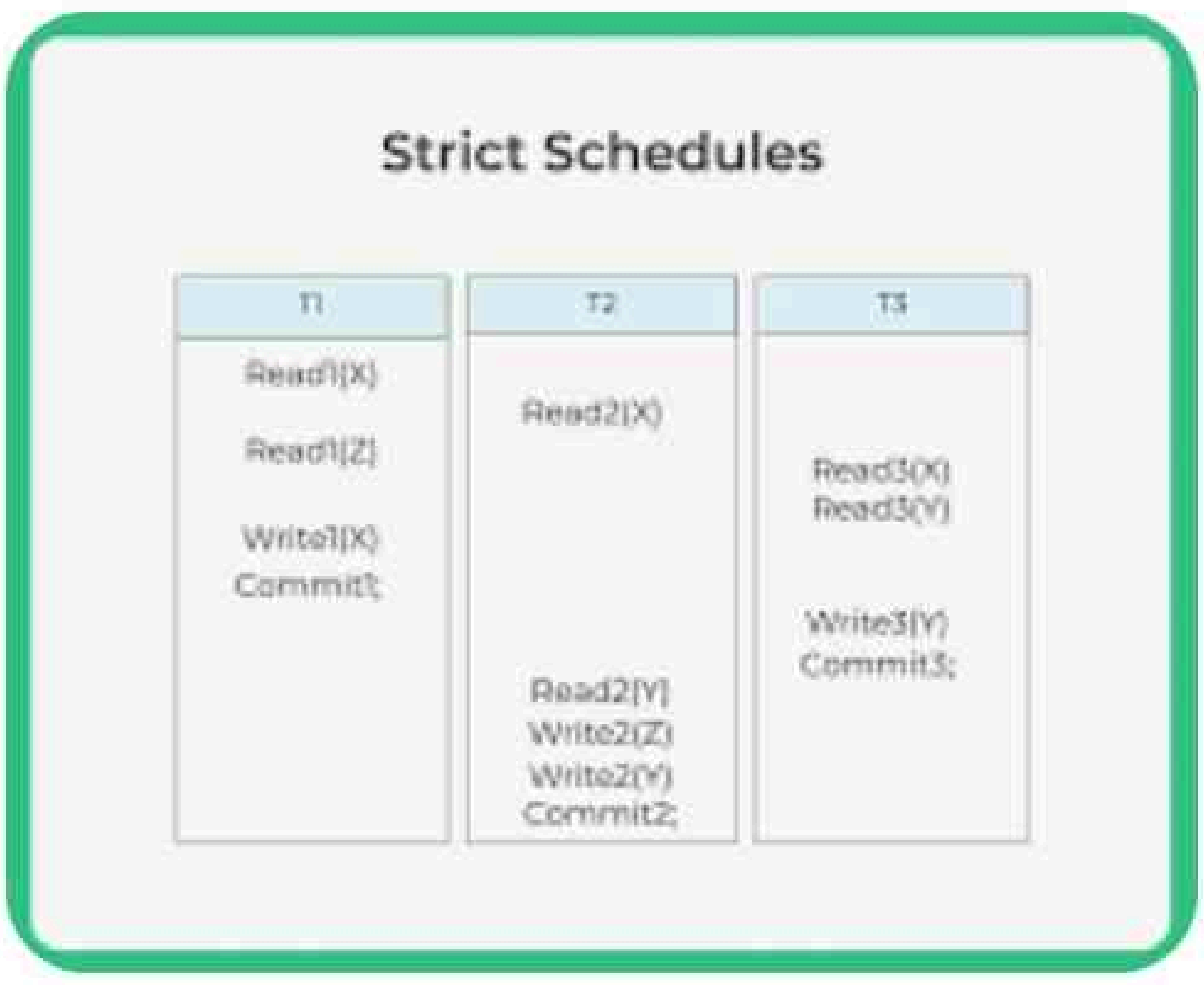
Share

Save



PrepInsta

Recoverability of Schedul...



PrepInsta

Recoverability of Schedul...

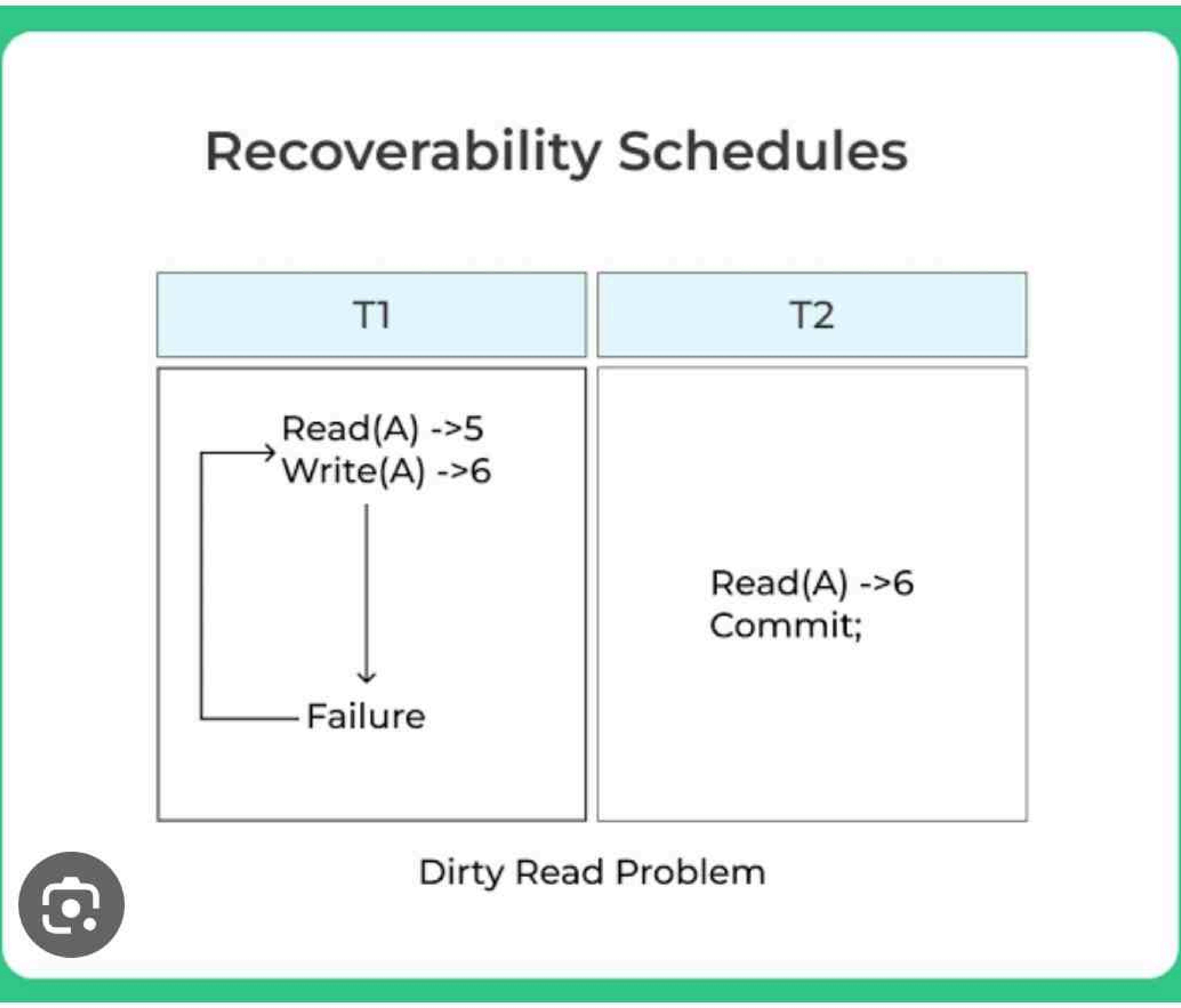
Discover

Search

Saved

22:14 84

PrepInsta

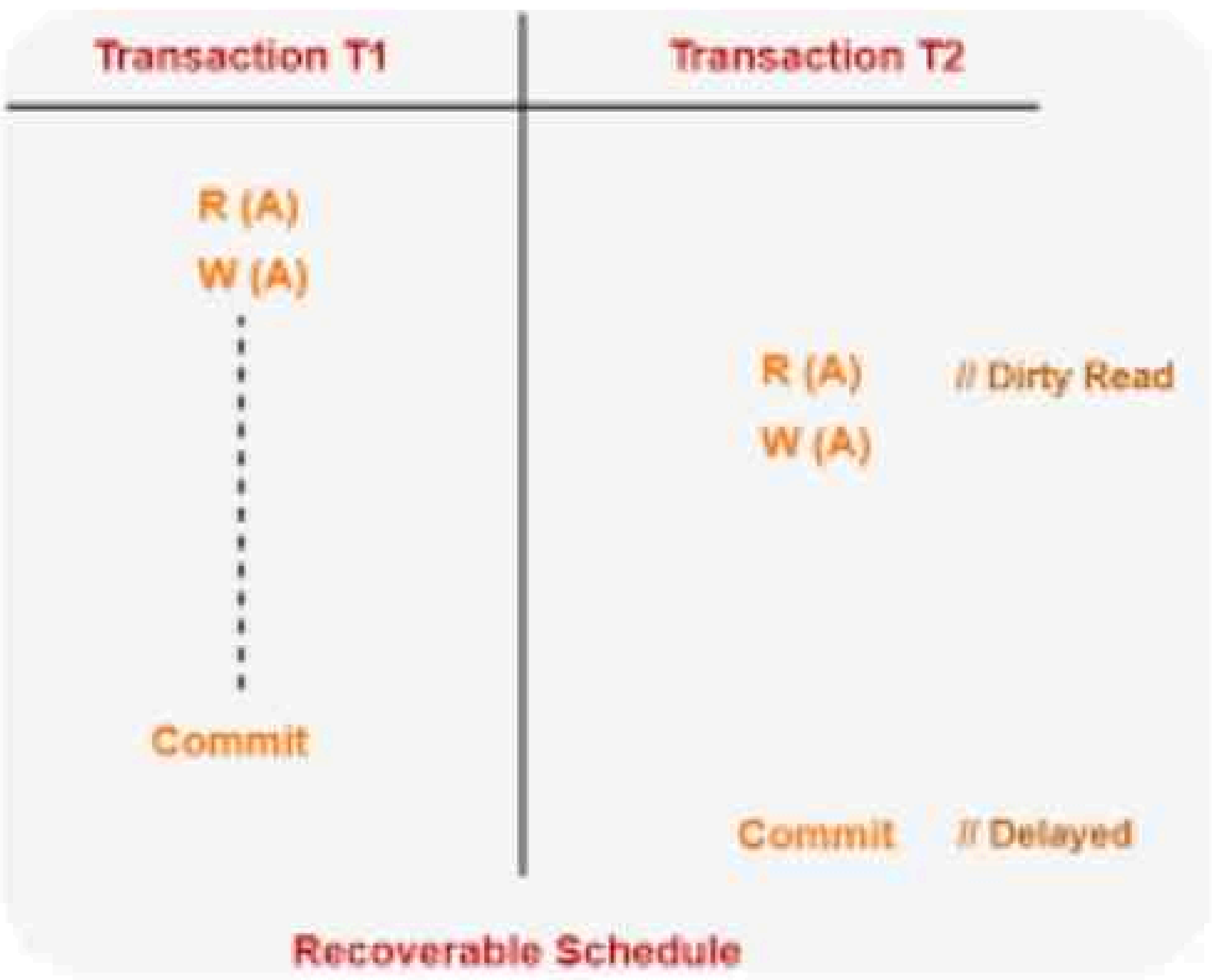


Recoverability of Schedules in DBMS | Database Managemen...

Visit >

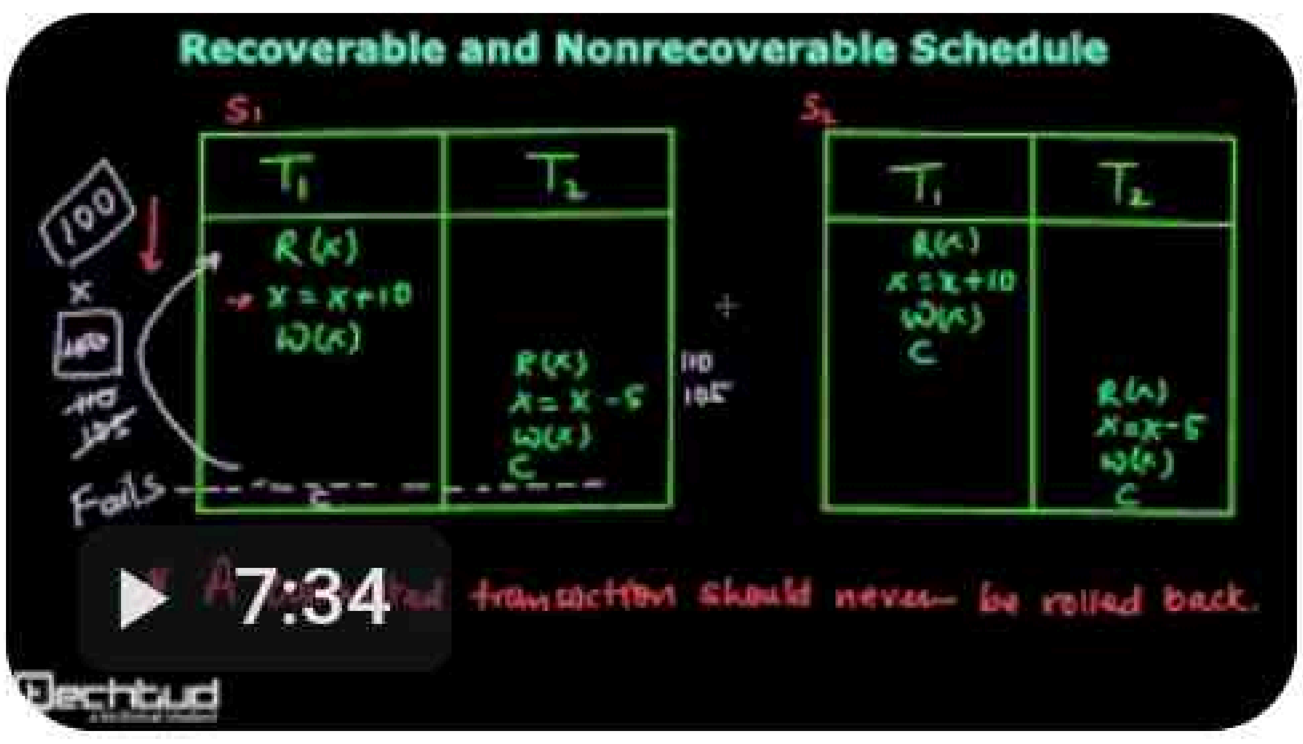
Share

Save



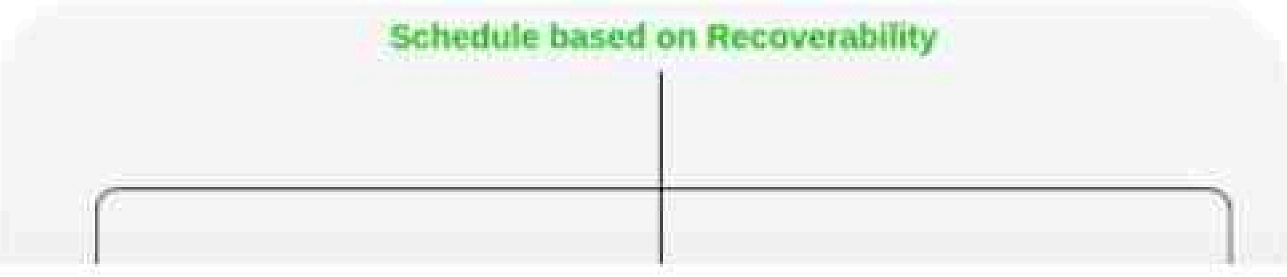
Gate Vidyalay

Schedules and Recovera...



YouTube

Recoverable and Nonrec...



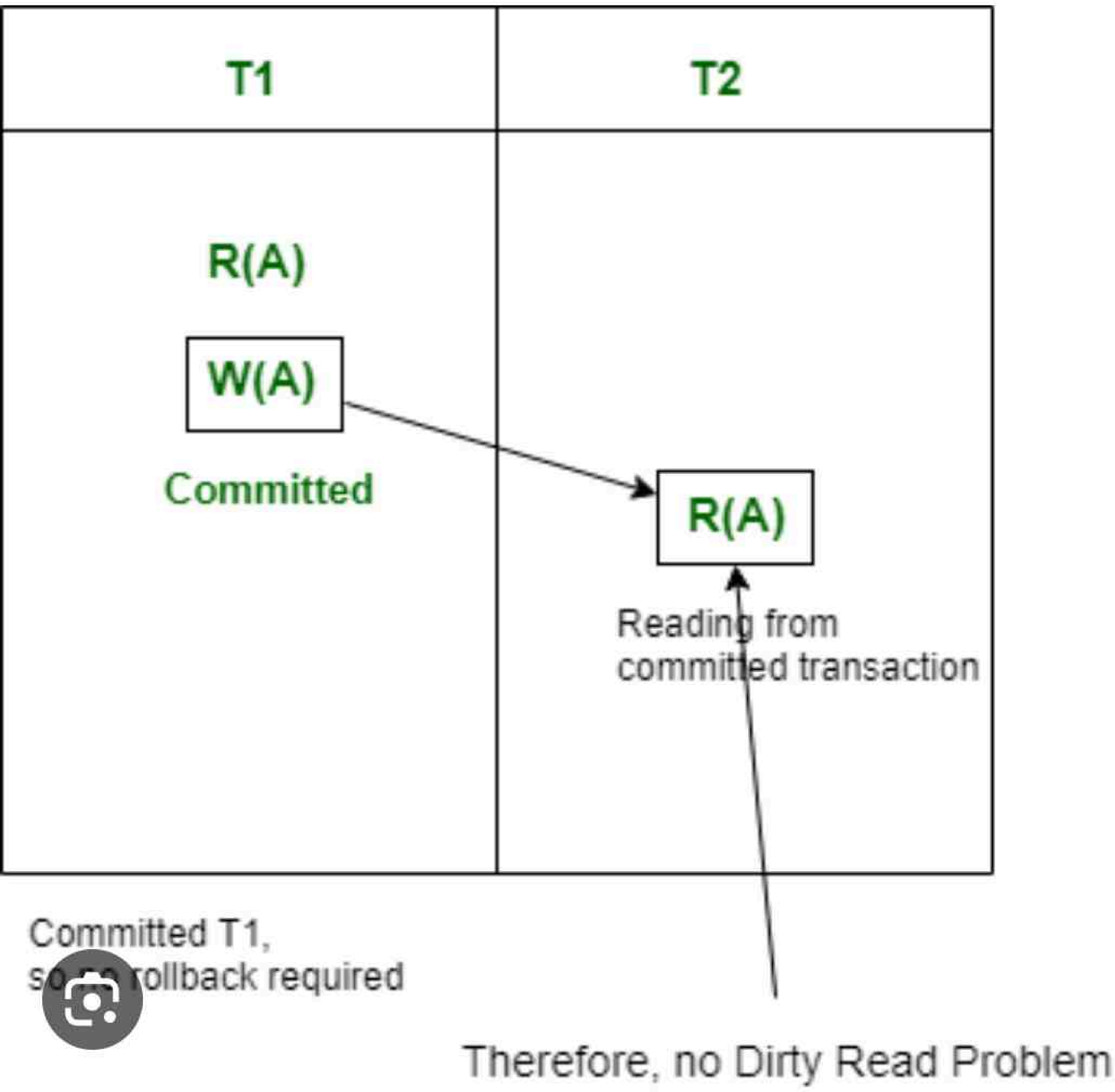
Discover

Search

Saved

22:14 ··· 85

GeeksforGeeks

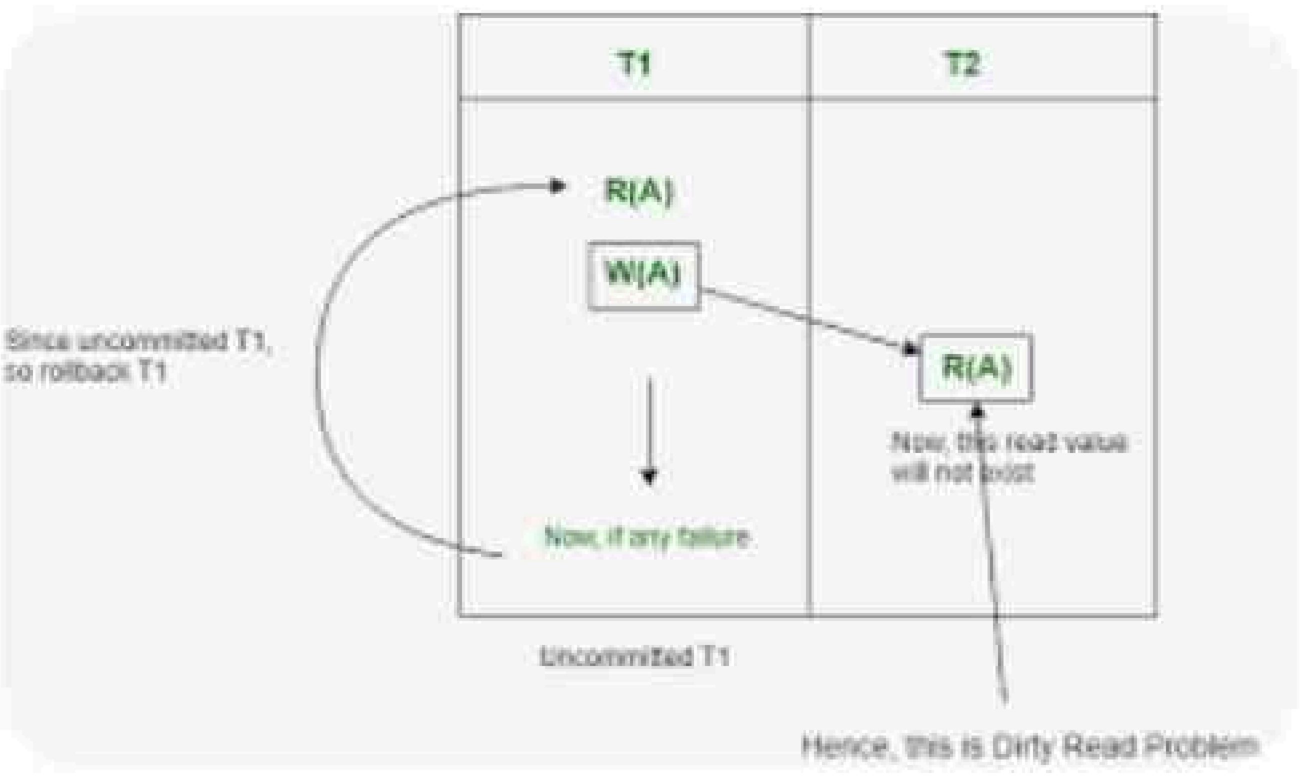


Cascadeless in DBMS - GeeksforGeeks

Visit >

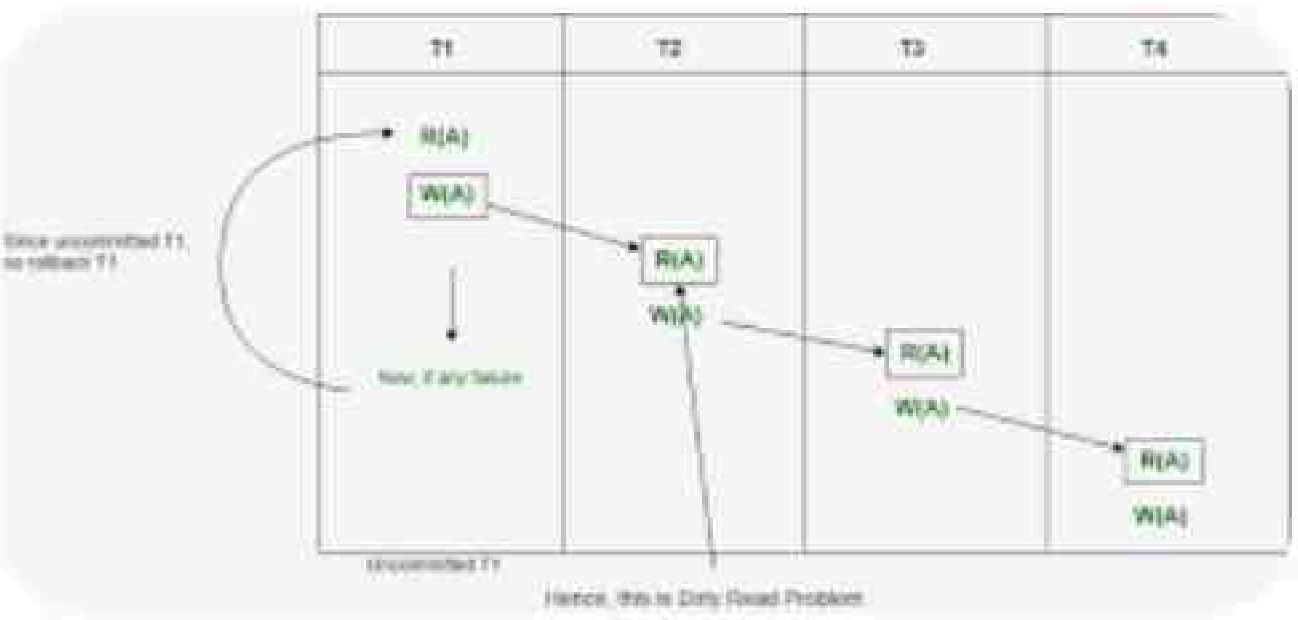
Share

Save



GeeksforGeeks

Cascadeless in DBMS - G...



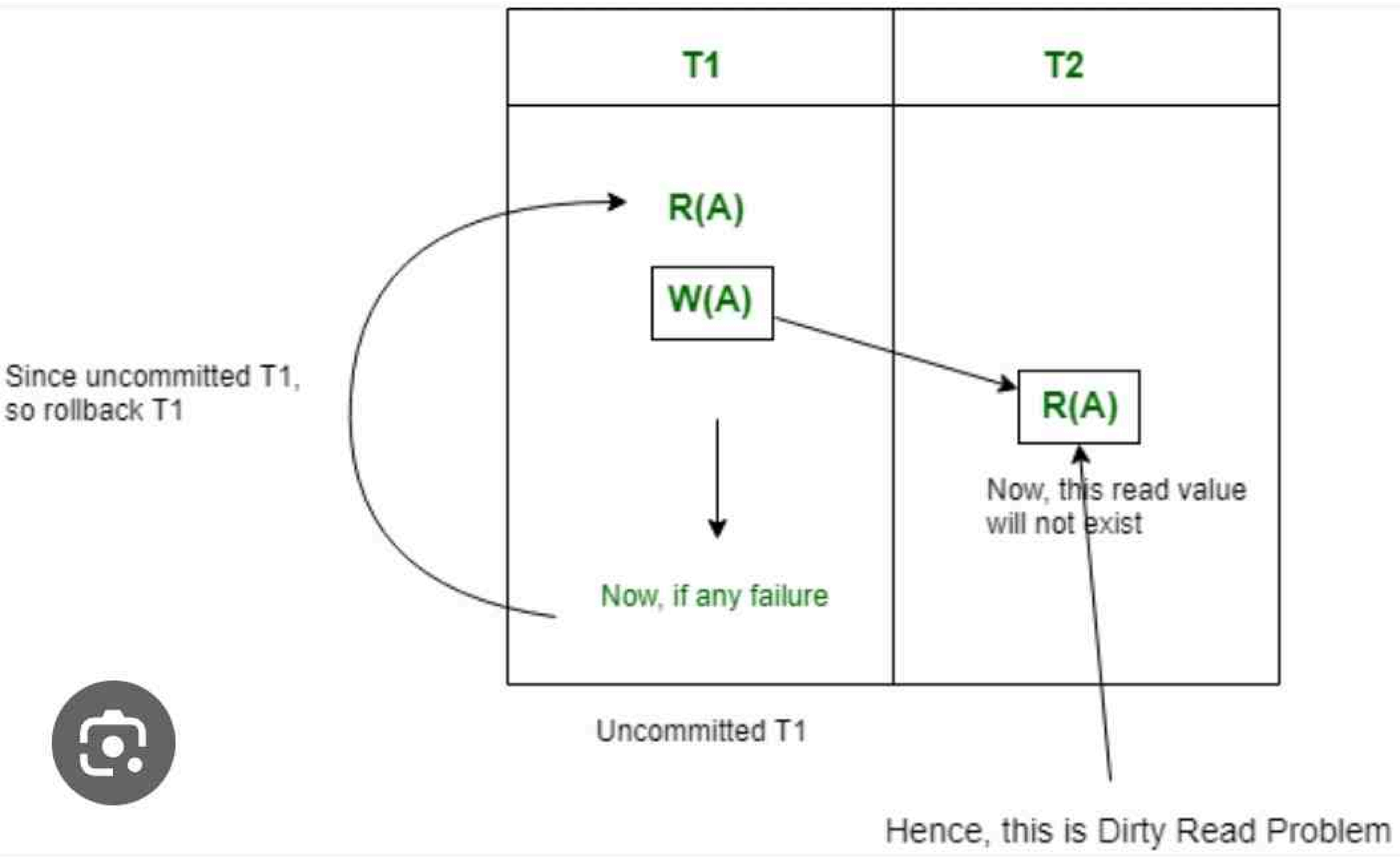
GeeksforGeeks

Cascadeless in DBMS - G...

Discover

Search

Saved

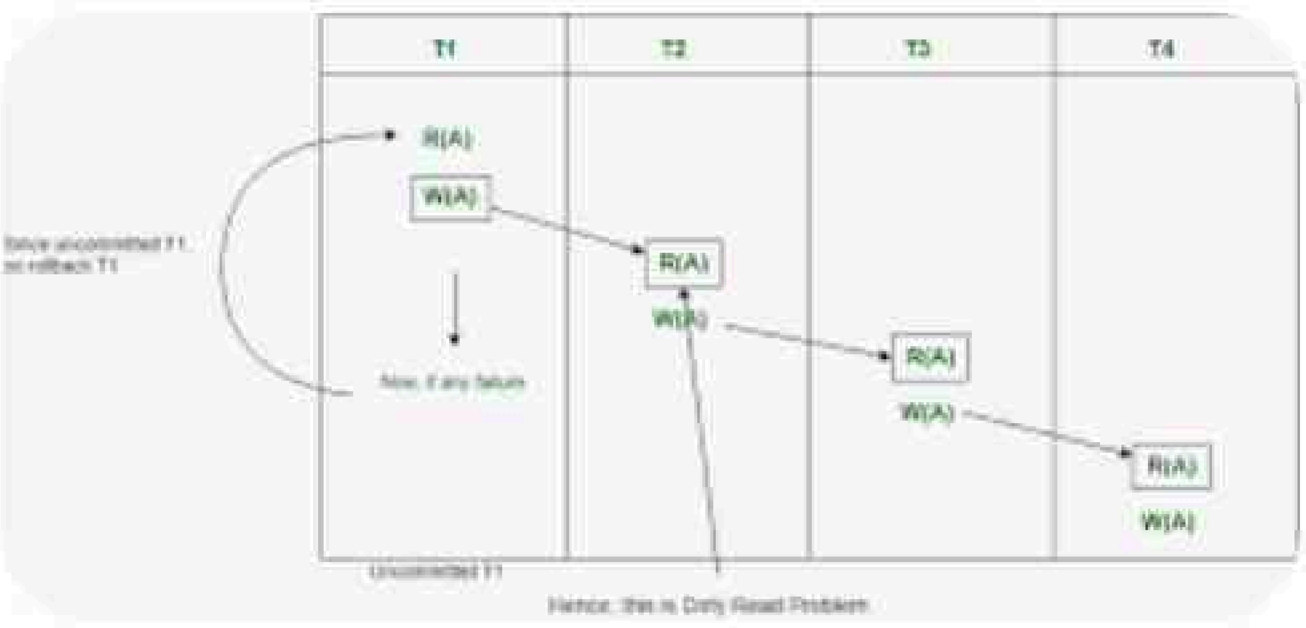


Cascadeless in DBMS - GeeksforGeeks

Visit >

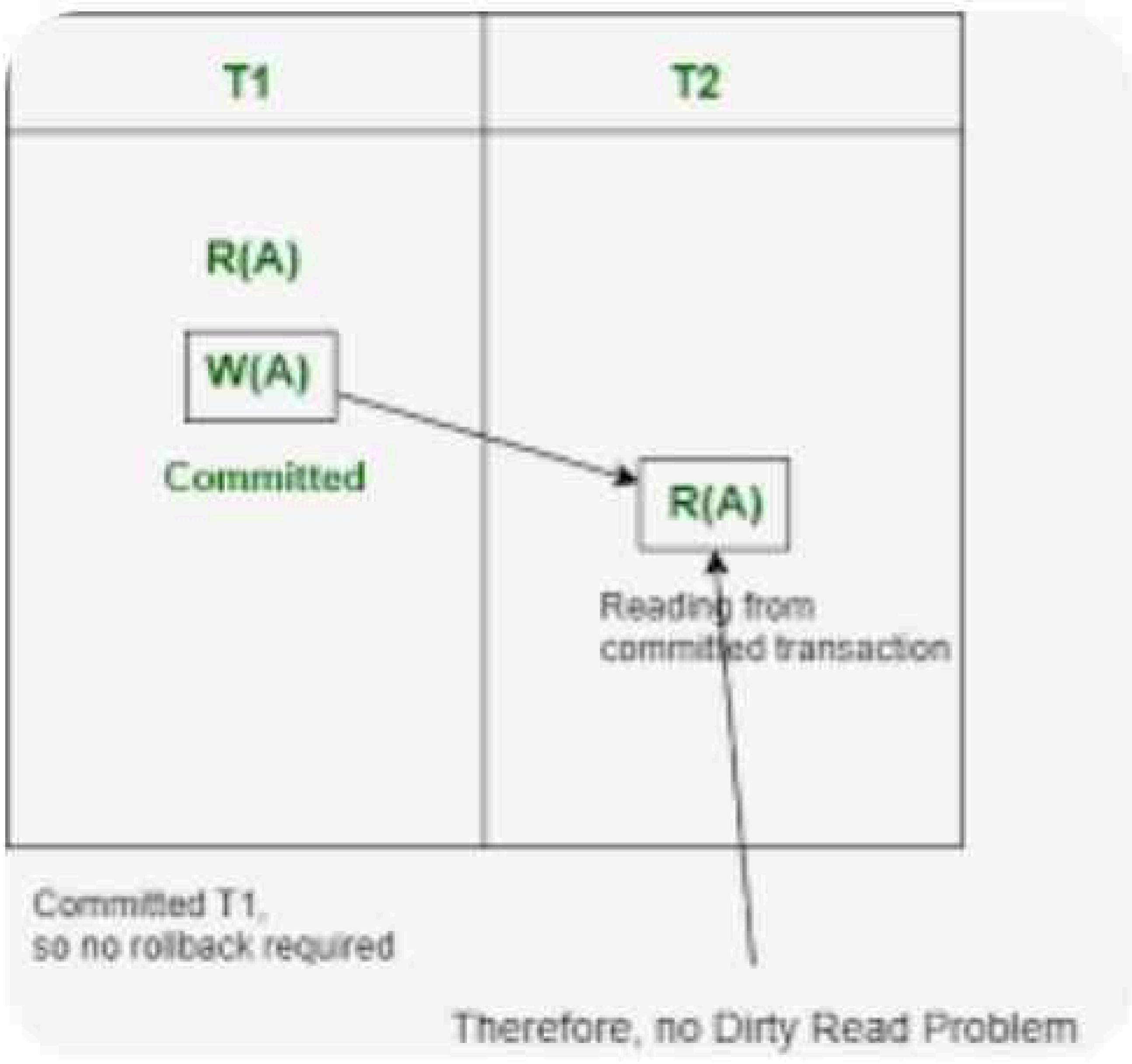
Share

Save



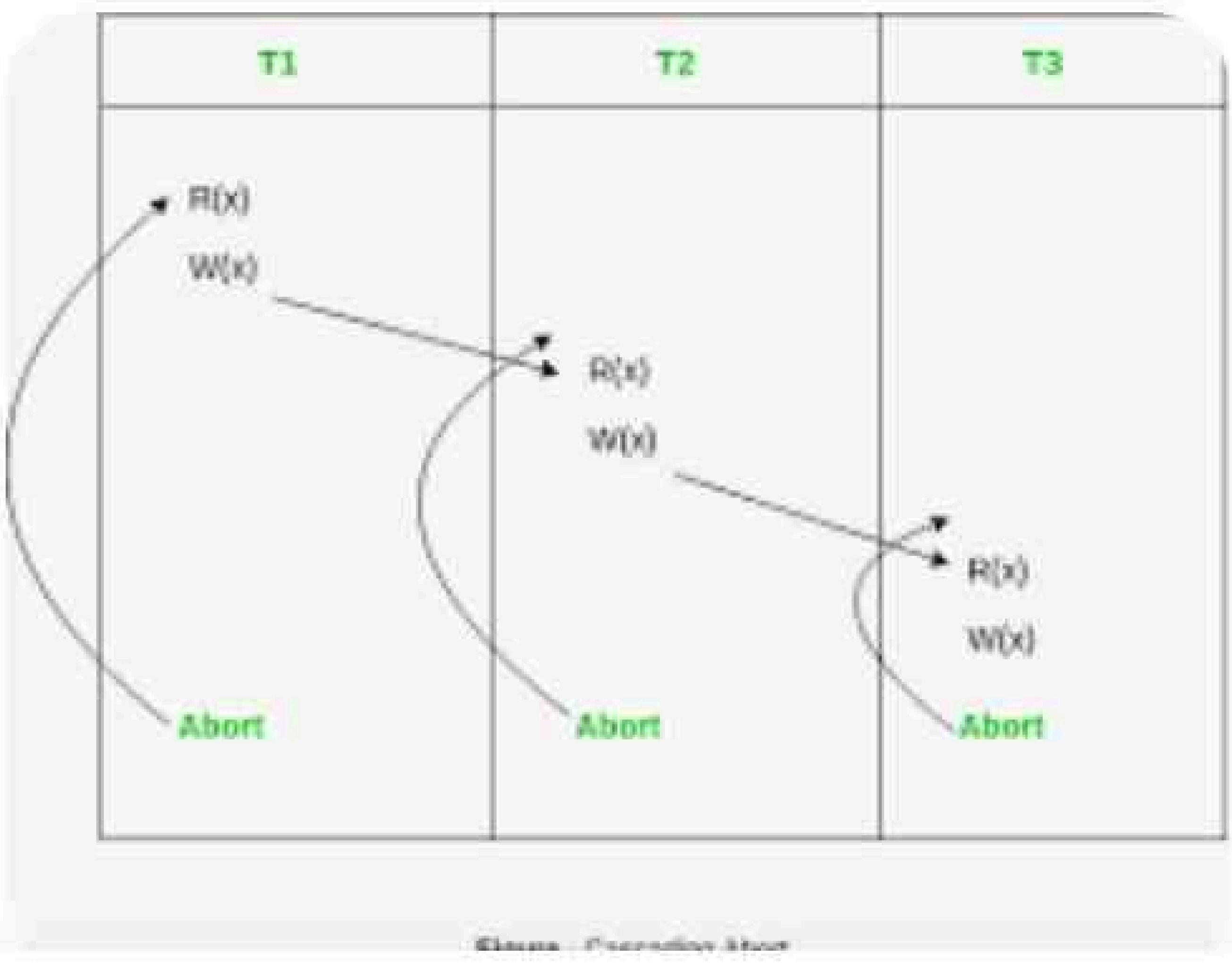
GeeksforGeeks

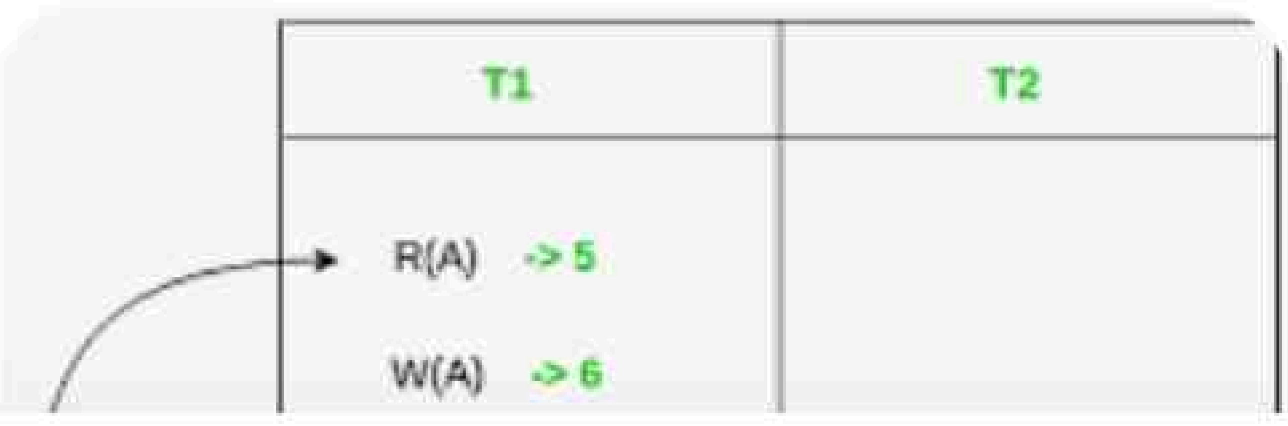
Cascadeless in DBMS - G...



GeeksforGeeks

Cascadeless in DBMS - G...





22:14

GeeksforGeeks

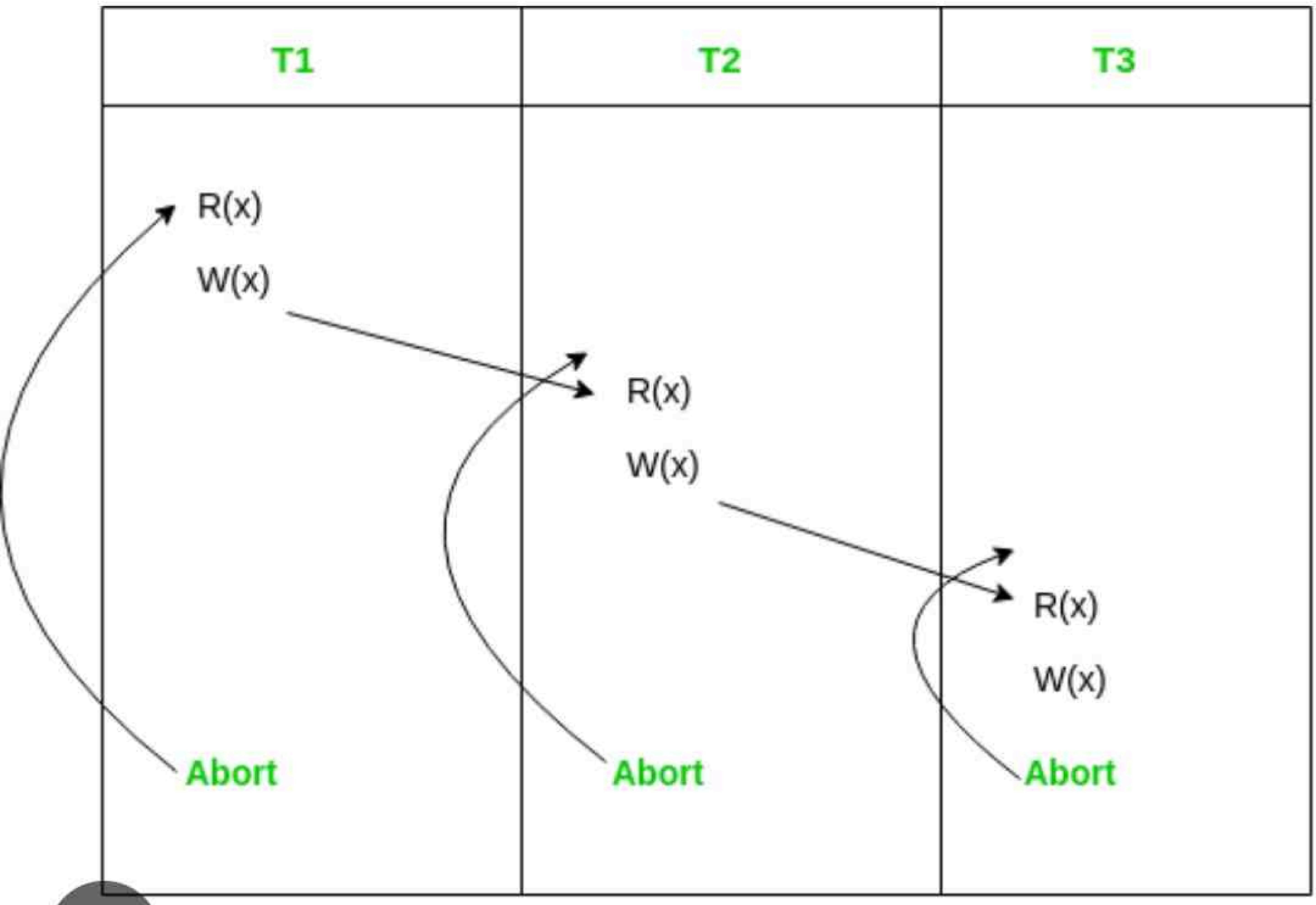


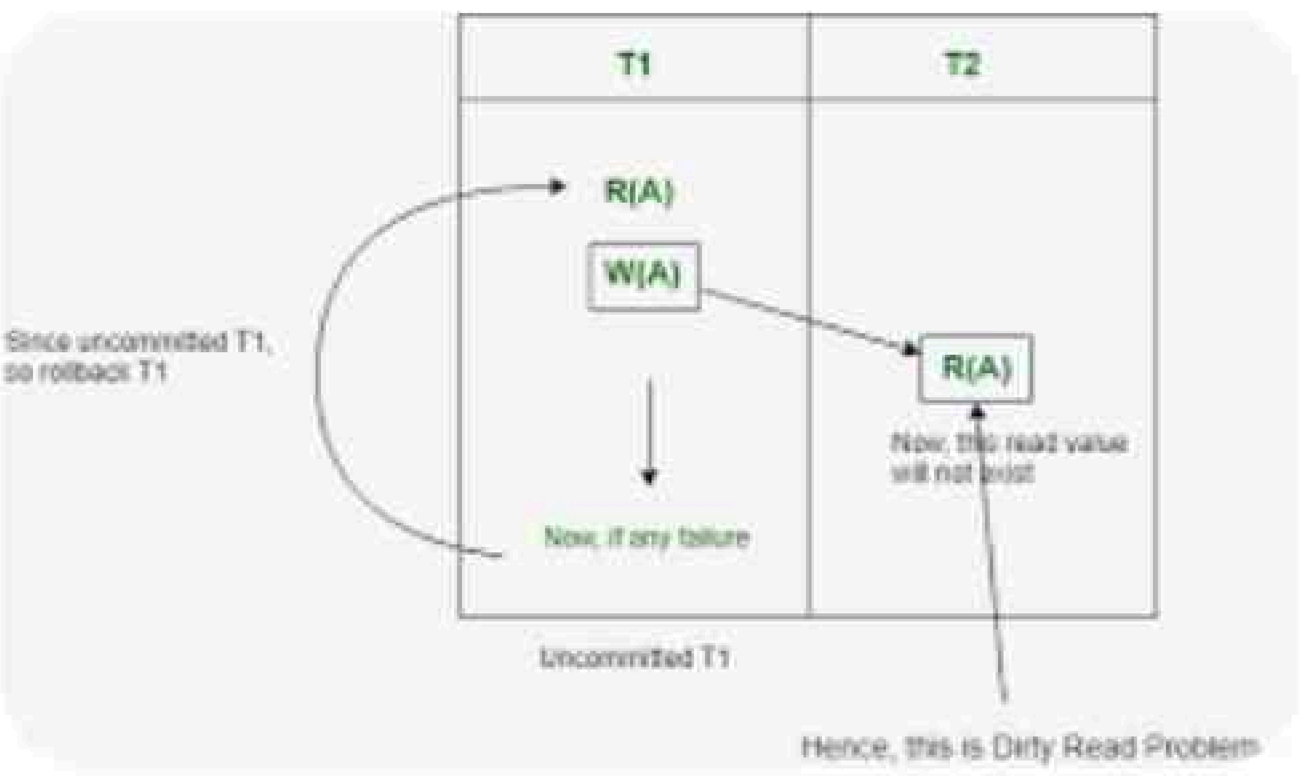
Figure - Cascading Abort

Types of Schedules in DBMS - GeeksforGeeks

Visit >

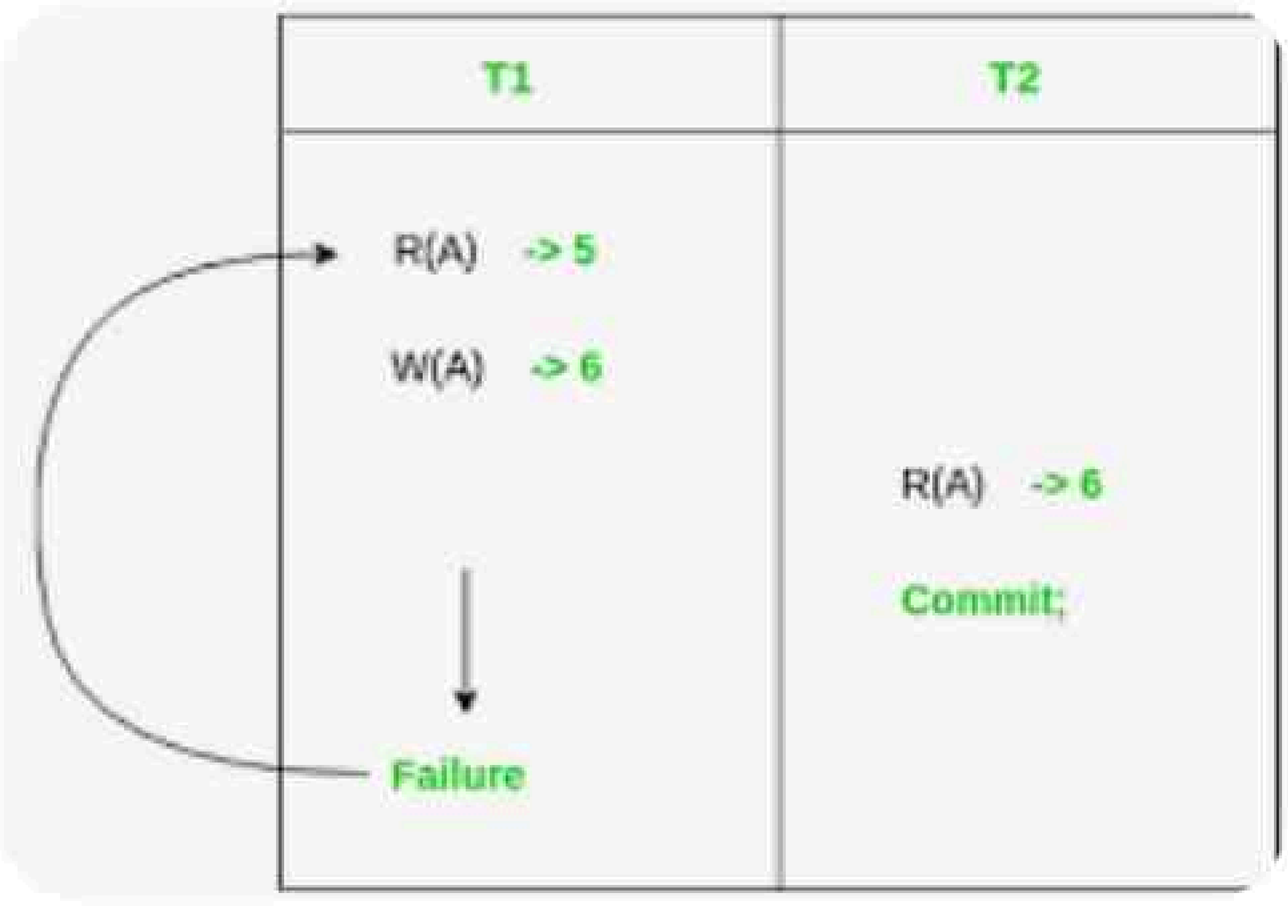
Share

Save



GeeksforGeeks

Cascadeless in DBMS - G...



EduRev

Types of Schedules base...

Discover

Search

Saved

22:15 85

GeeksforGeeks

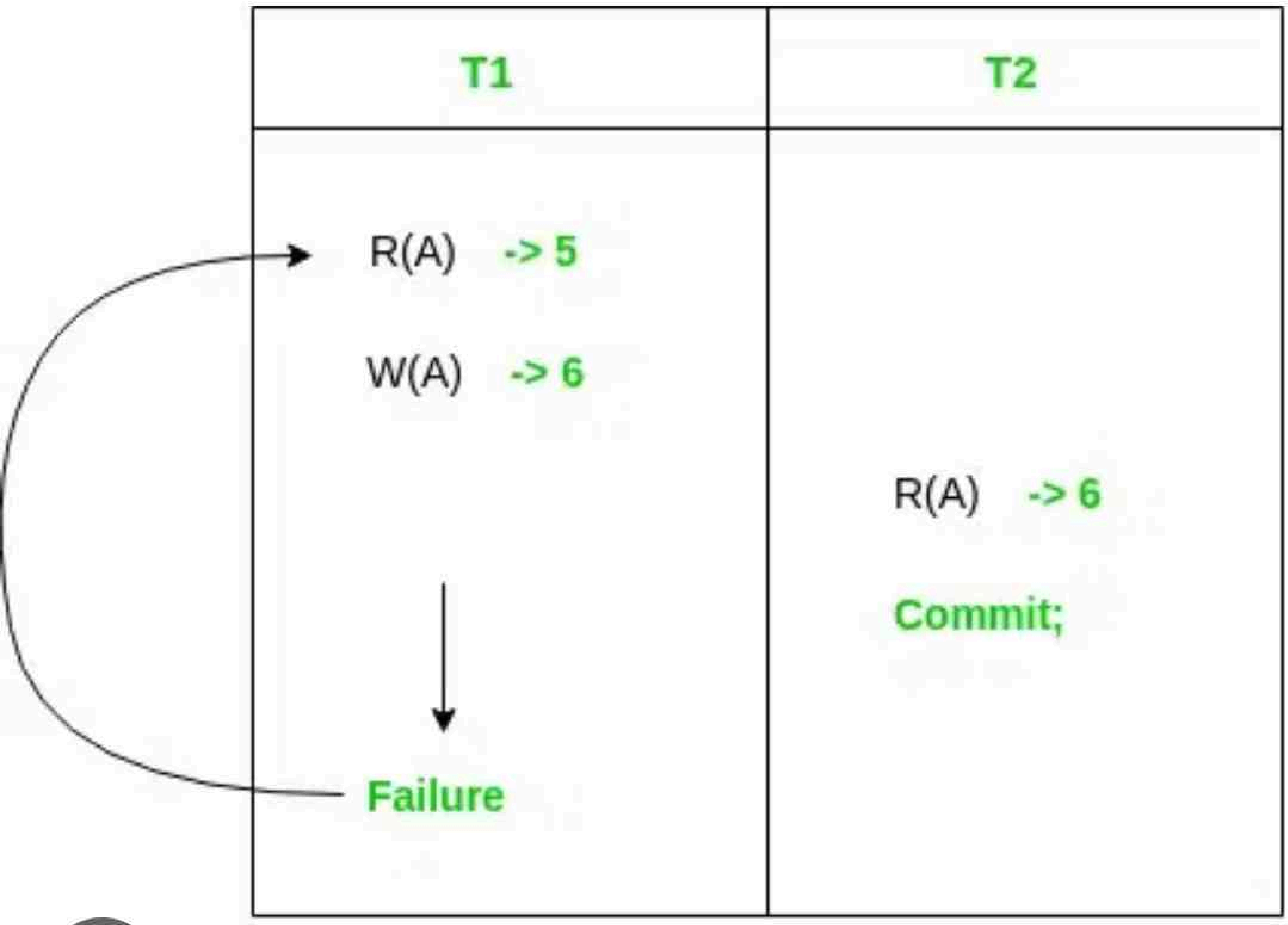


Figure - Dirty Read Problem

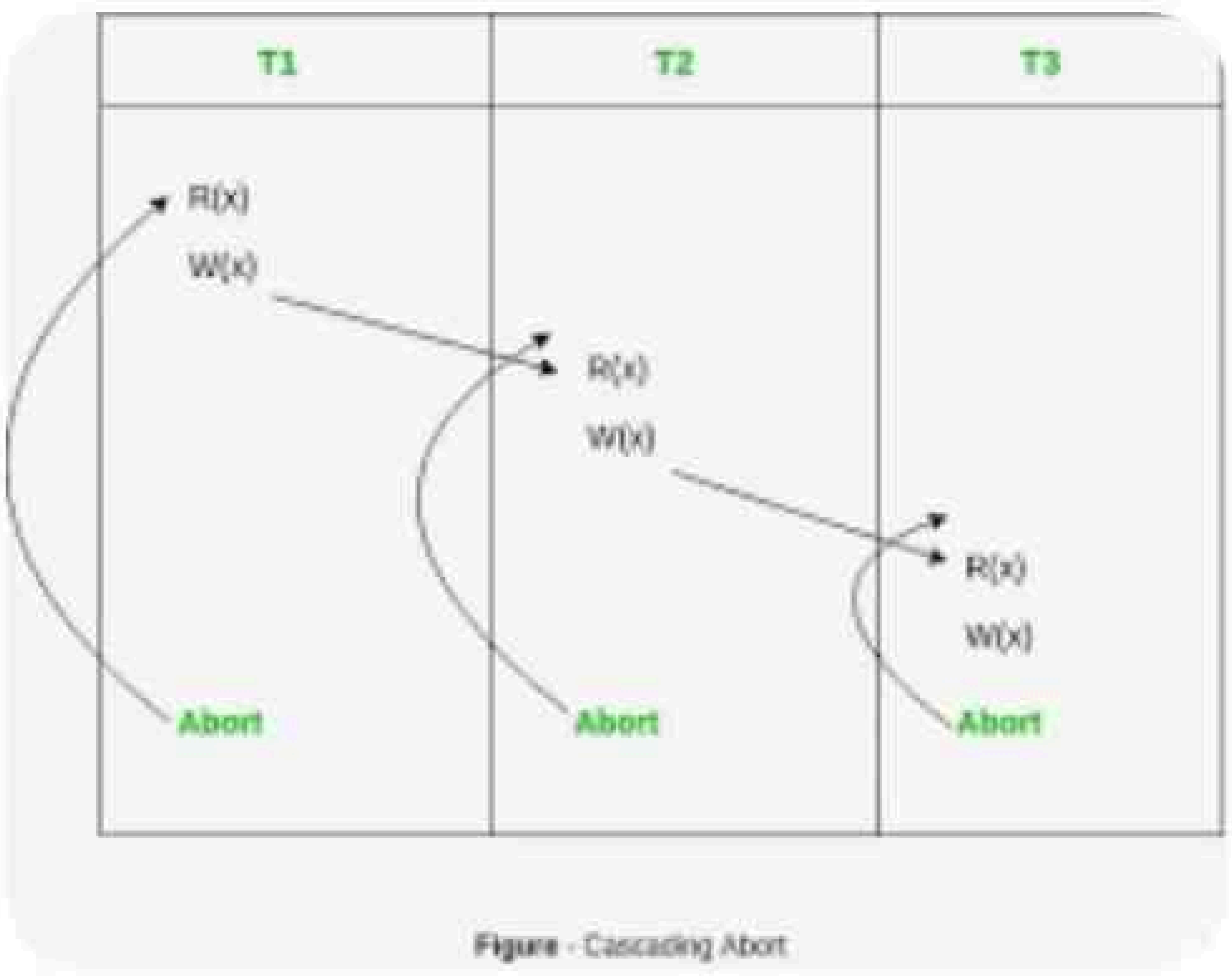
Types of Schedules based Recoverability in DBMS -...

Visit >

Images may be subject to copyright. **Learn more**

Share

Save



GeeksforGeeks

Types of Schedules base...

GeeksforGeeks

Types of Schedules base

Discover

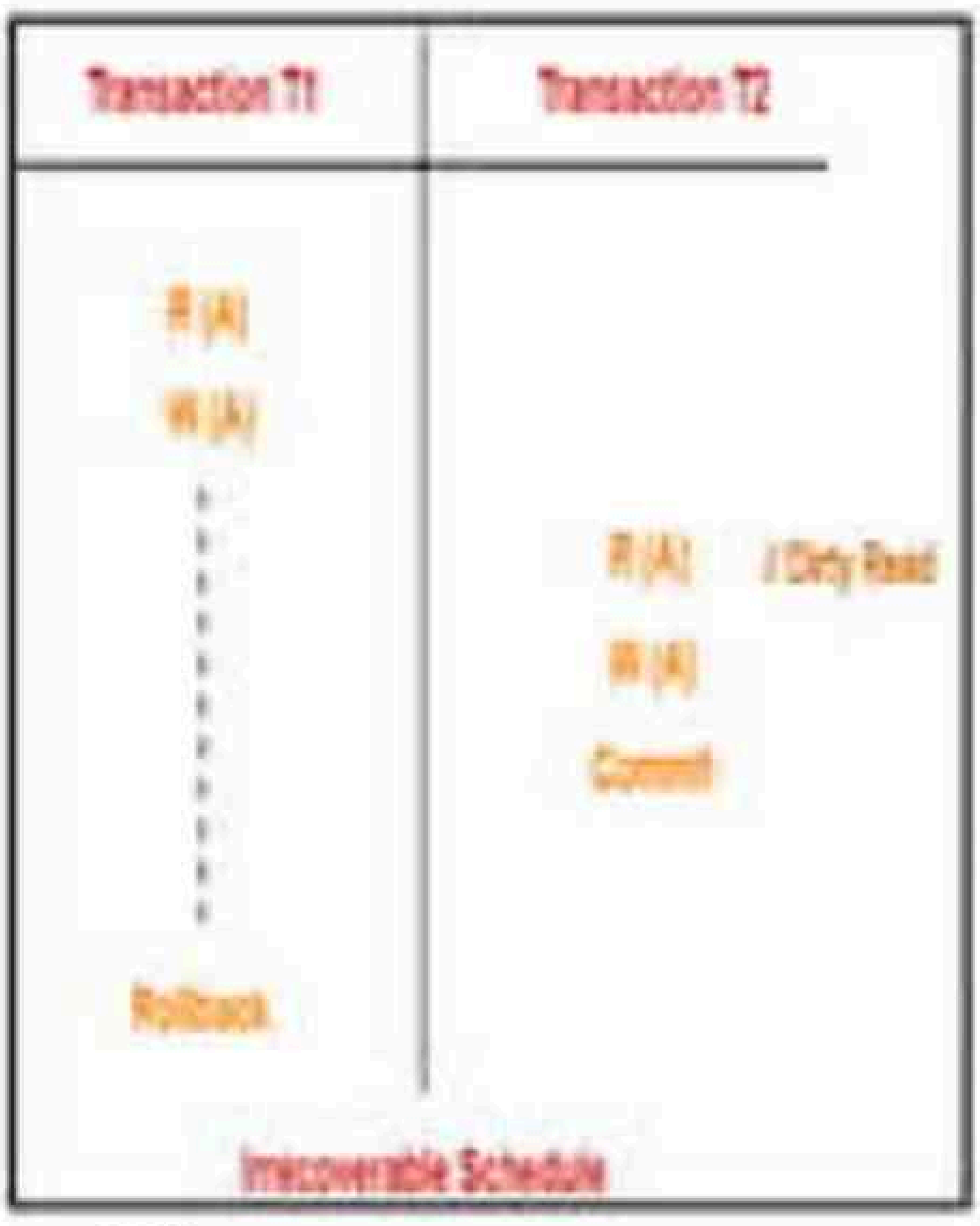
Search

Saved

22:16 85

SlideShare

Non -Recoverable Schedule



Here,

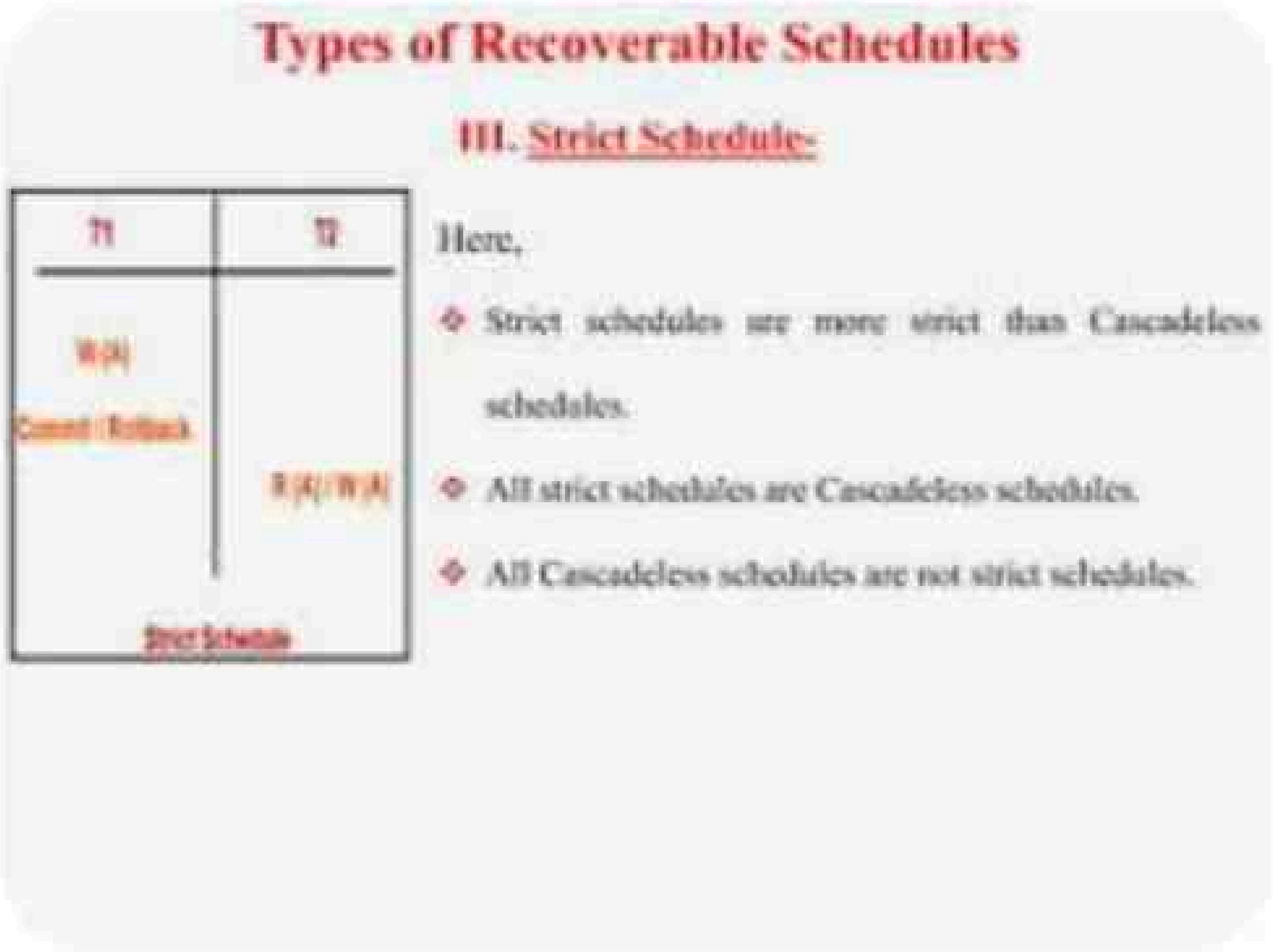
- T2 performs a dirty read operation.
- T2 commits before T1.
- T1 fails later and roll backs.
- The value that T2 read now stands to be incorrect.
- T2 can not recover since it has already committed.

DBMS UNIT IV.pptx

Visit >

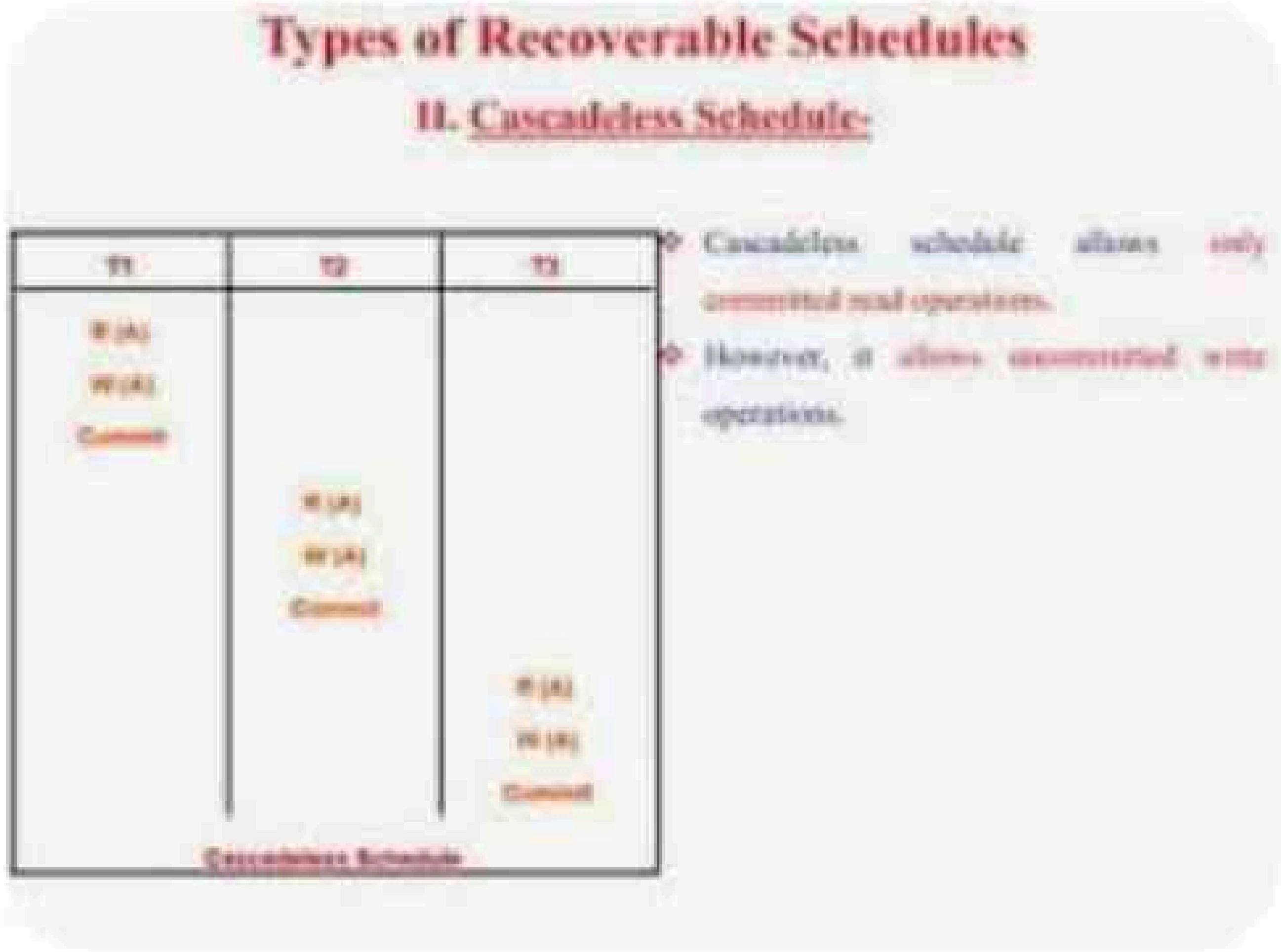
Share

Save



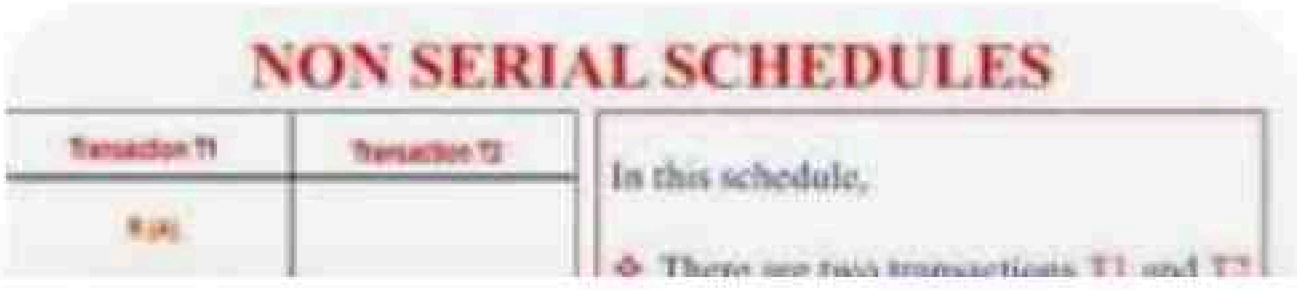
SlideShare

DBMS UNIT IV.pptx



SlideShare

DBMS UNIT IV.pptx



Discover

Search

Saved

22:16

SlideShare

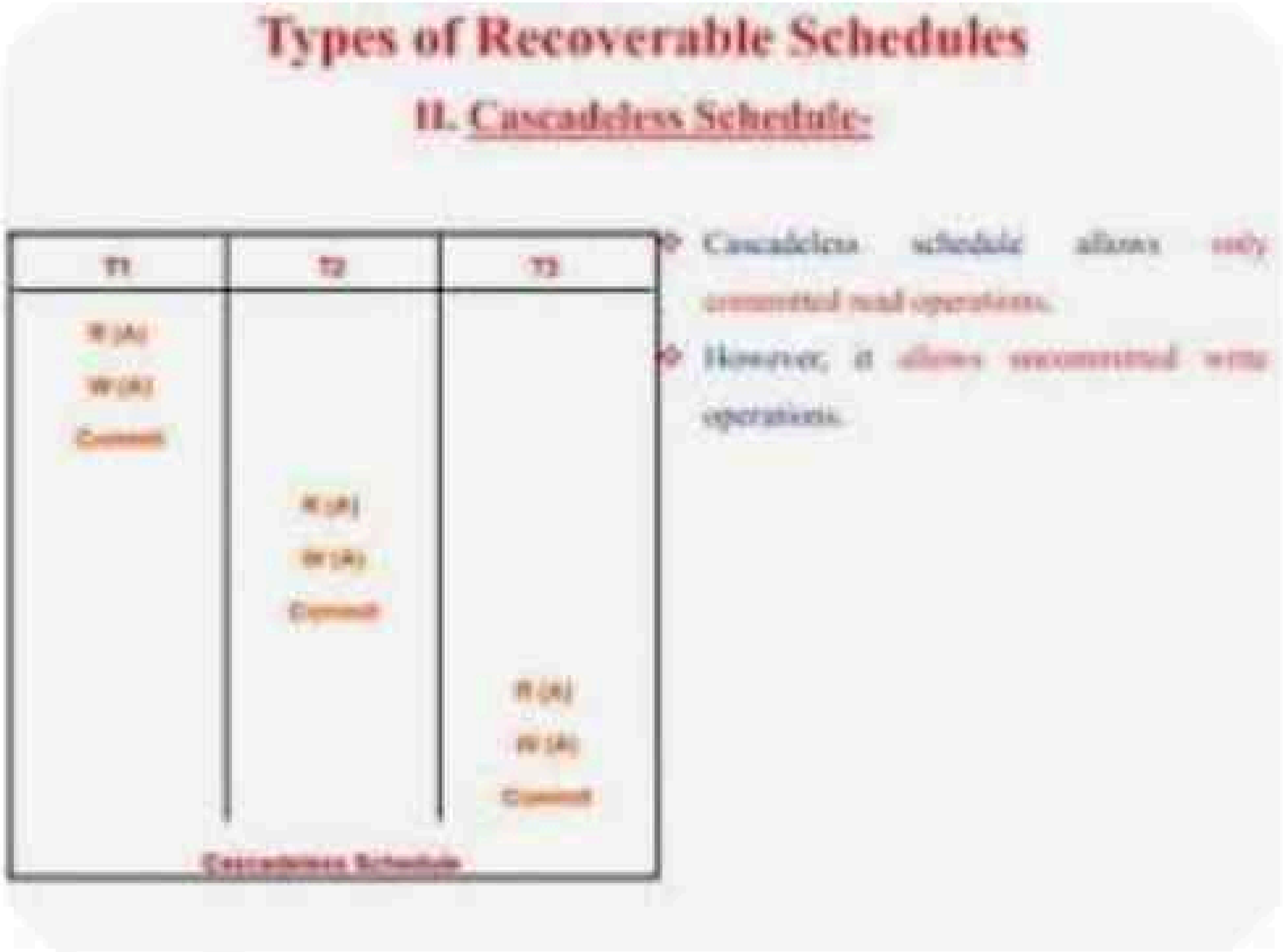
DBMS UNIT IV.pptx

Visit >

Images may be subject to copyright. **Learn more**

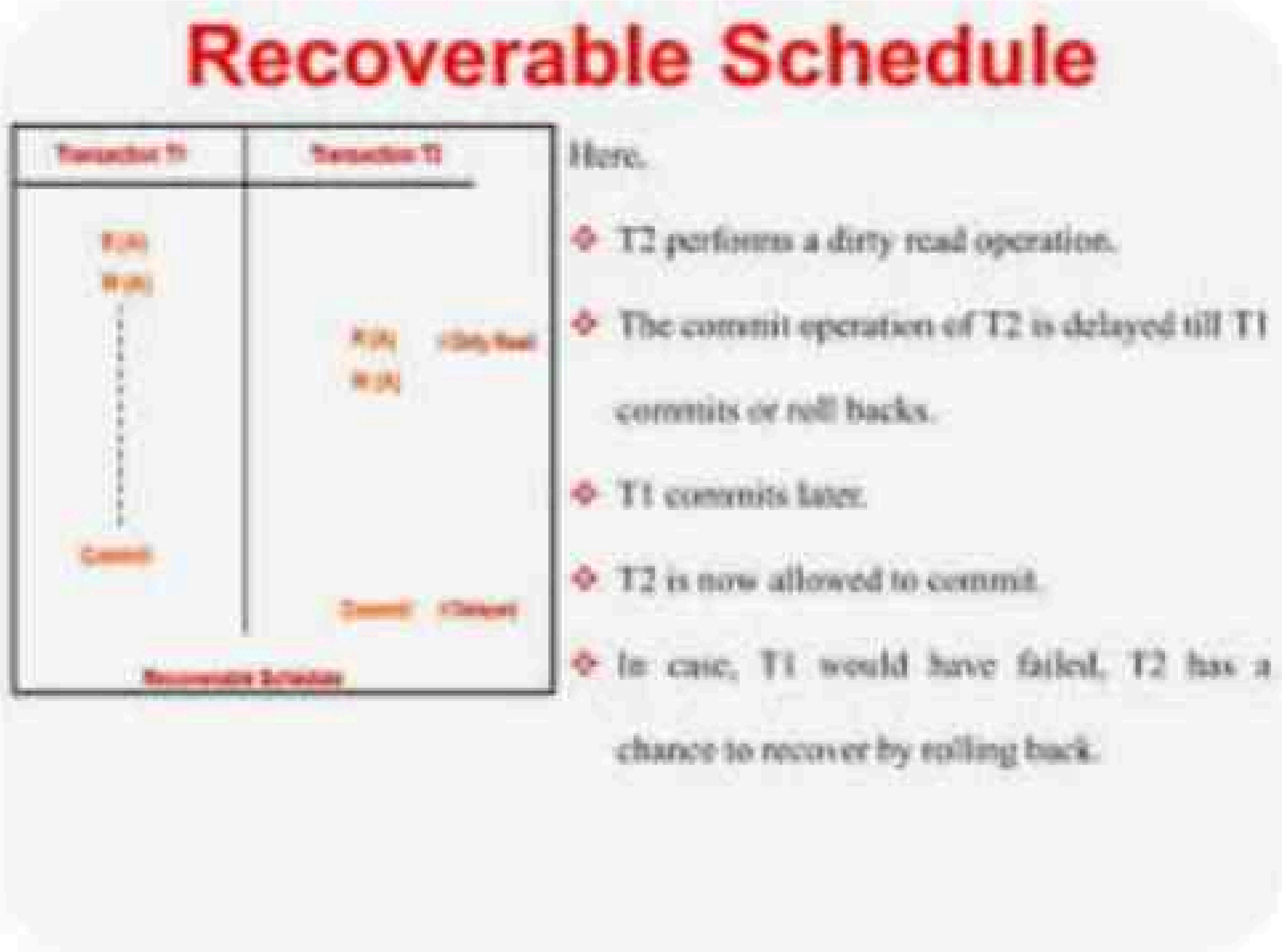
Share

Save



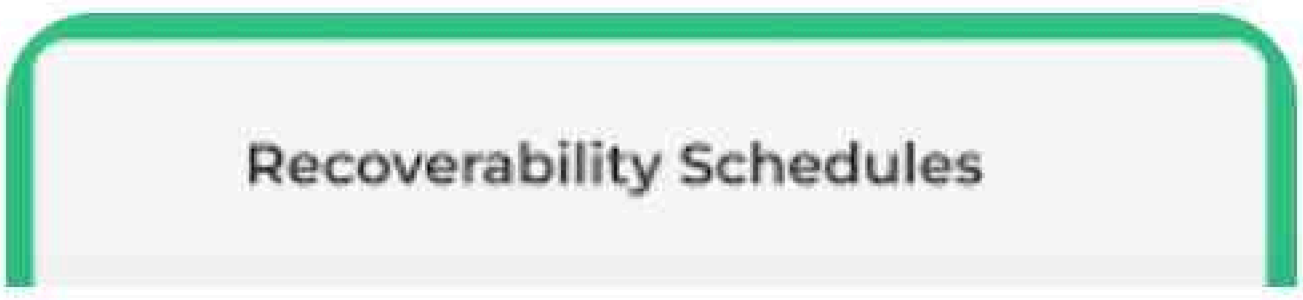
SlideShare

DBMS UNIT IV.pptx



SlideShare

DBMS UNIT IV.pptx



Discover

Search

Saved

22:16

SlideShare

SCHEDULES BASED ON RECOVERY

SCHEDULE D

T1	T2
Read(X)	
Write(X)	
	Read(X)
Read(Y)	
	Write(X)
Write(Y)	
Abort	
	Abort

- The Schedule D is Recoverable.
- T2 reads item X from T1.
- T1 Aborts instead of Committing, Then T2 Should also Abort.
- Because the value of X it read is no longer valid

DBMS UNIT IV.pptx

Visit >

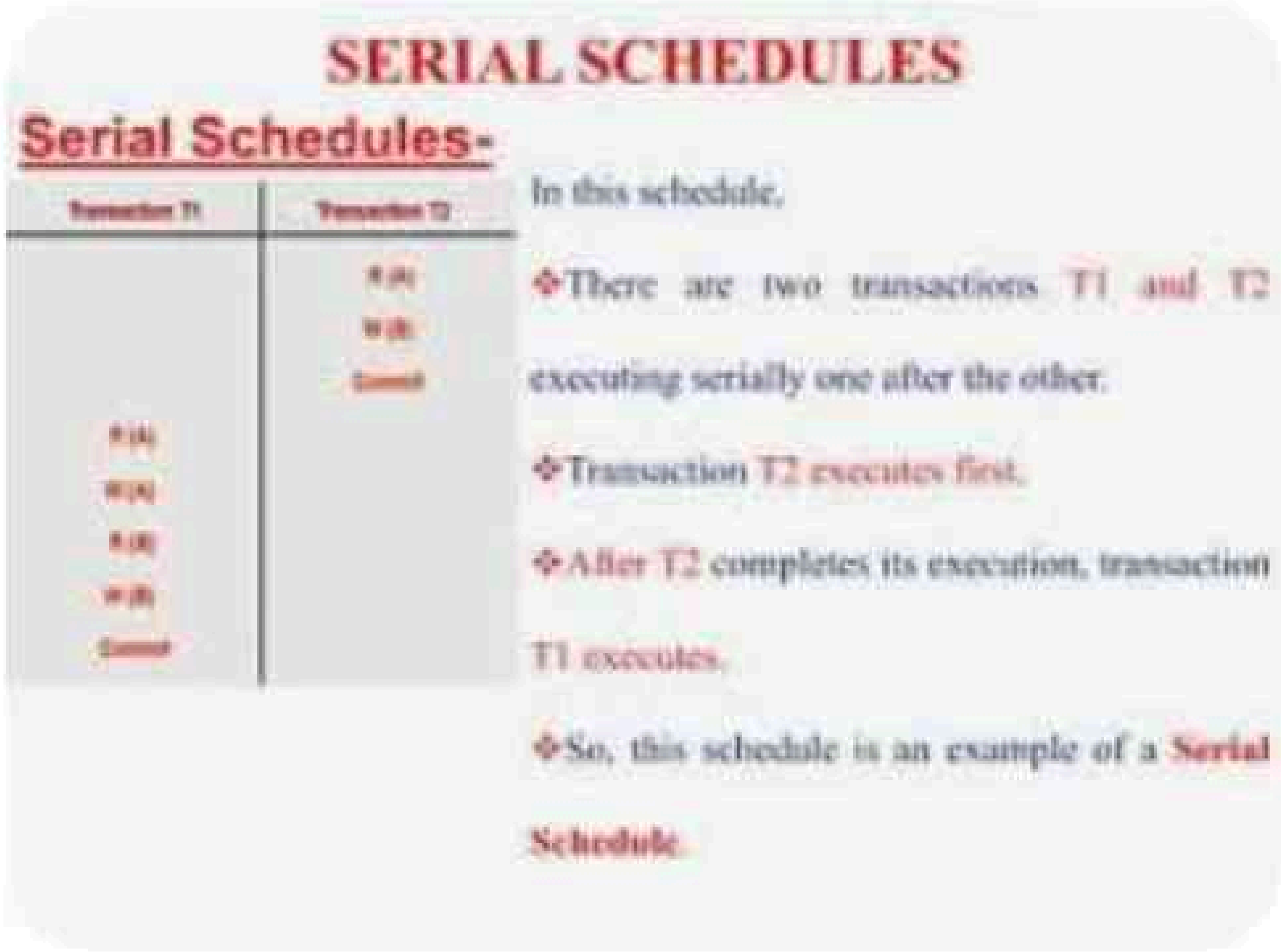
Images may be subject to copyright. **Learn more**

Share

Save

SlideShare

DBMS UNIT IV.pptx



SlideShare

DBMS UNIT IV.pptx

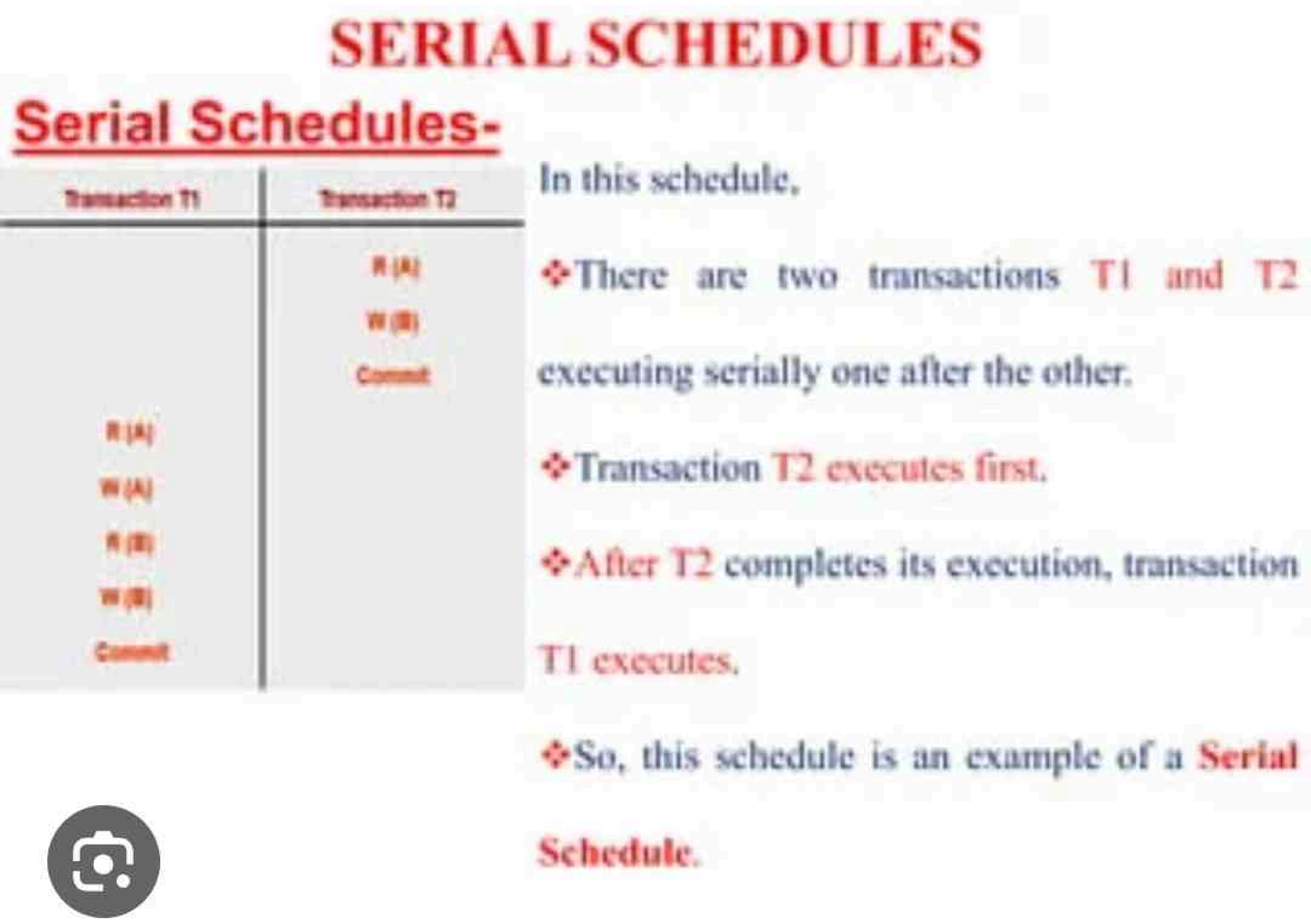
Discover

Search

Saved

22:16

SlideShare



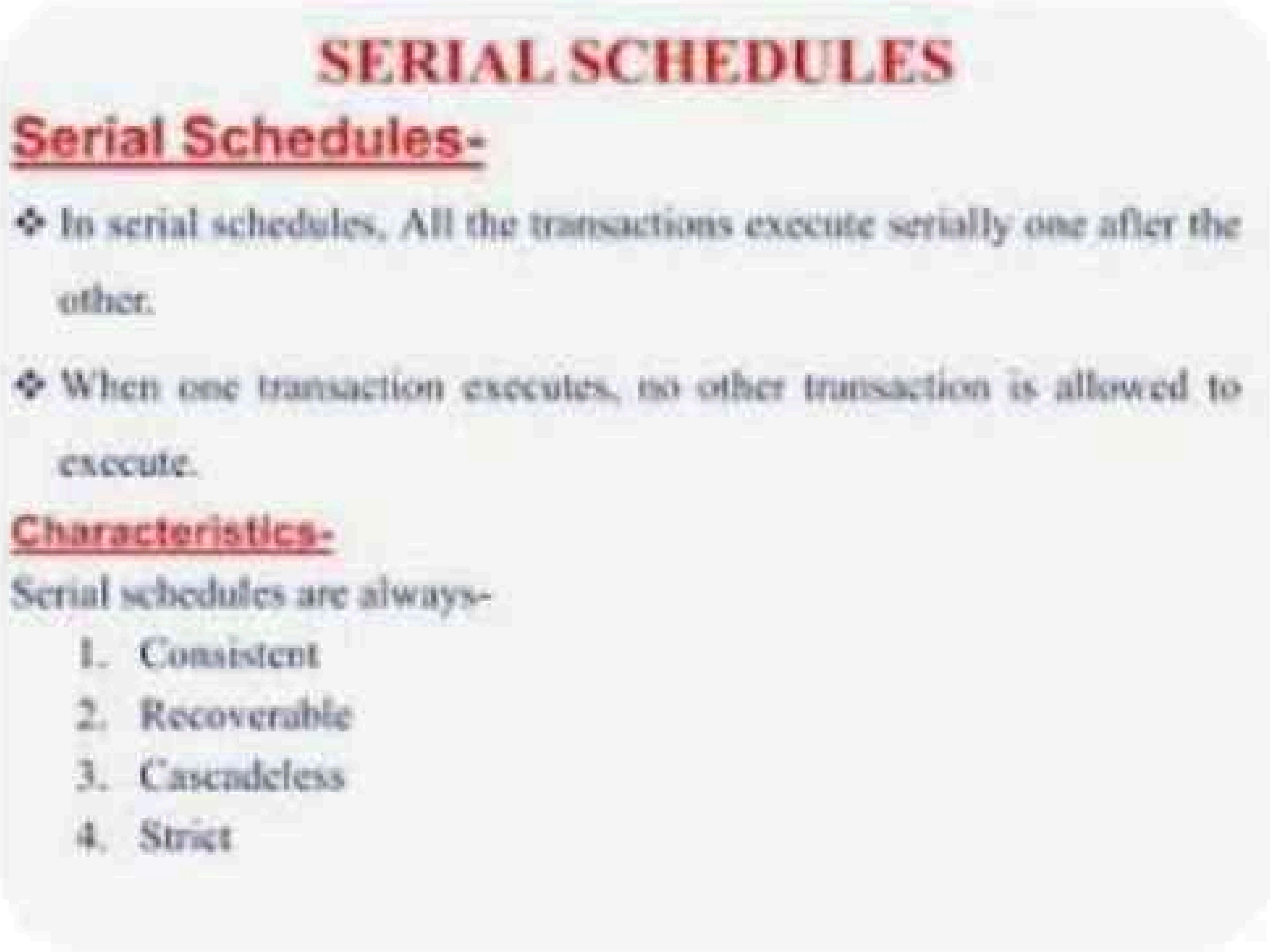
DBMS UNIT IV.pptx

Visit >

Images may be subject to copyright. **Learn more**

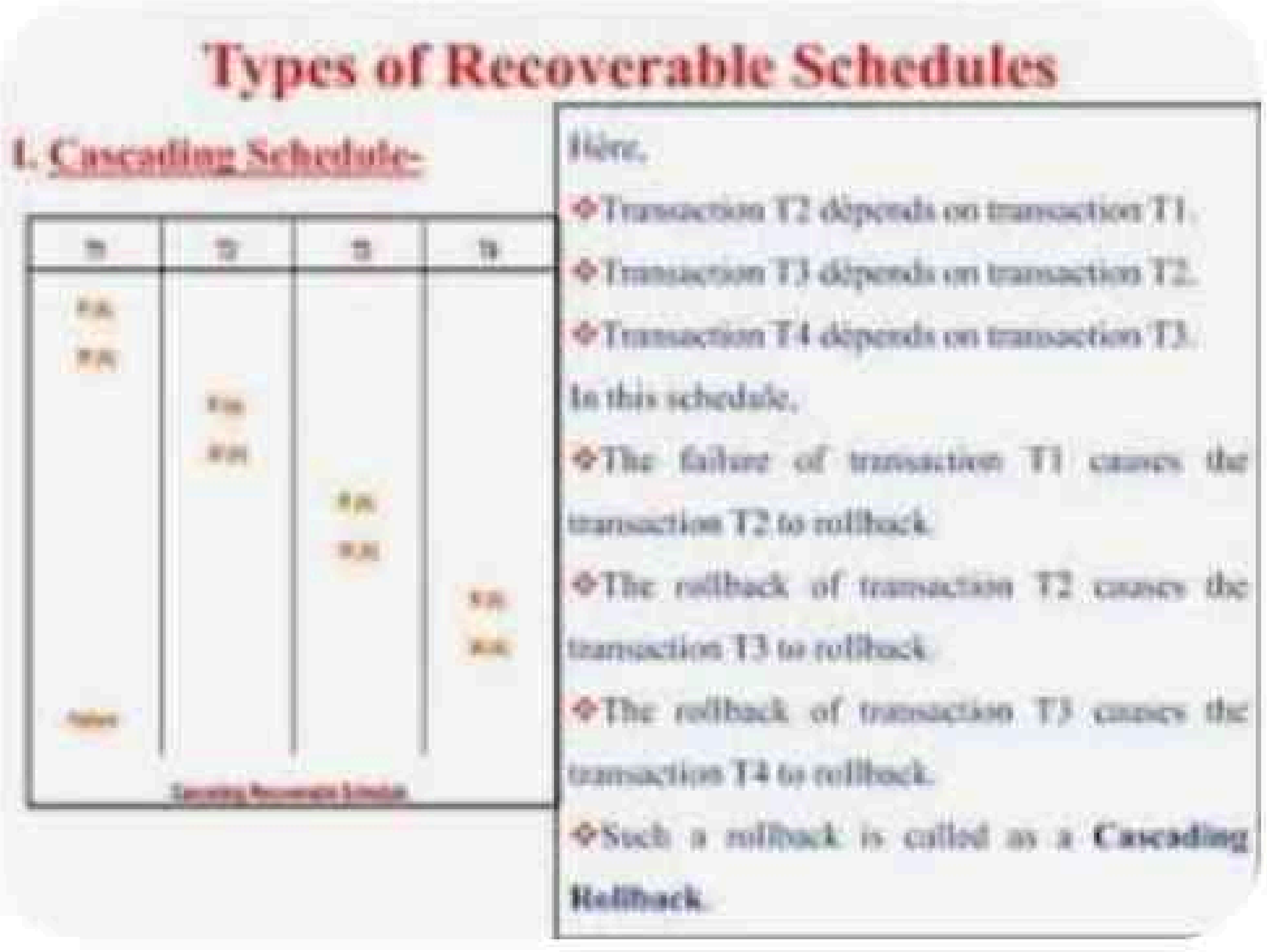
Share

Save



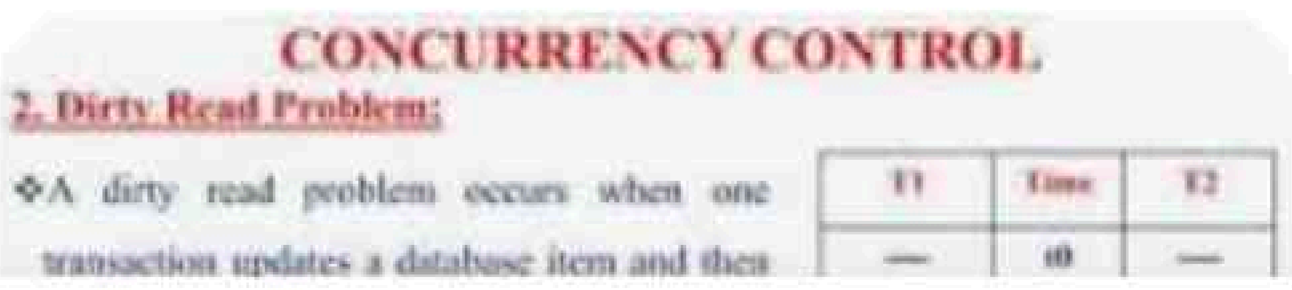
SlideShare

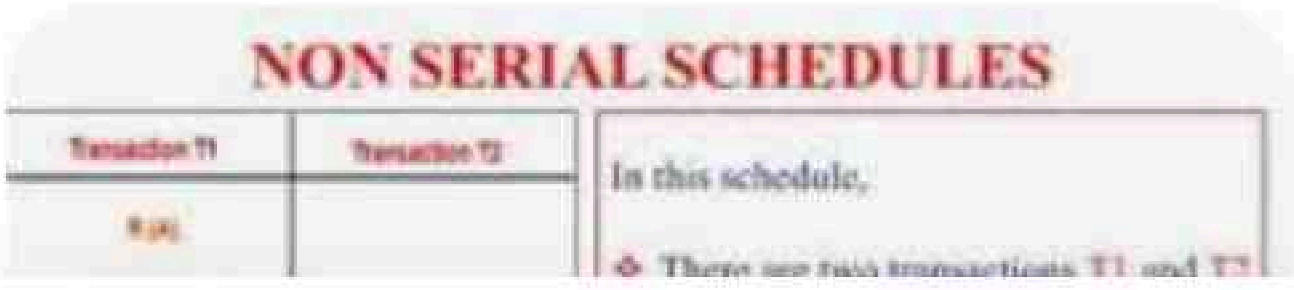
DBMS UNIT IV.pptx



SlideShare

DBMS UNIT IV.pptx





Discover

Search

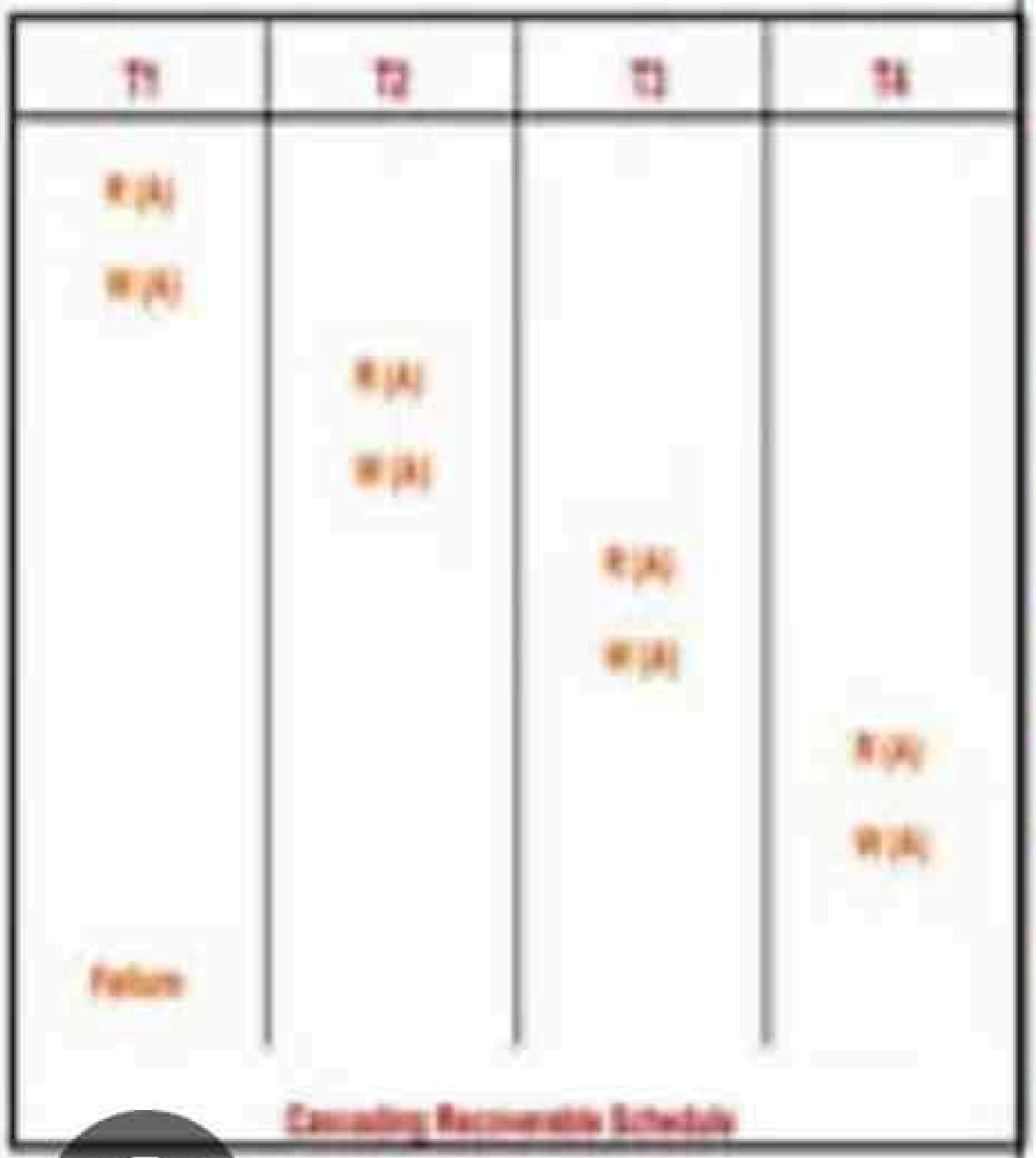
Saved

22:16 85

SlideShare

Types of Recoverable Schedules

I. Cascading Schedule-



Here,

- Transaction T2 depends on transaction T1.
- Transaction T3 depends on transaction T2.
- Transaction T4 depends on transaction T3.

In this schedule,

- The failure of transaction T1 causes the transaction T2 to rollback.
- The rollback of transaction T2 causes the transaction T3 to rollback.
- The rollback of transaction T3 causes the transaction T4 to rollback.
- Such a rollback is called as a **Cascading Rollback**.

DBMS UNIT IV.pptx

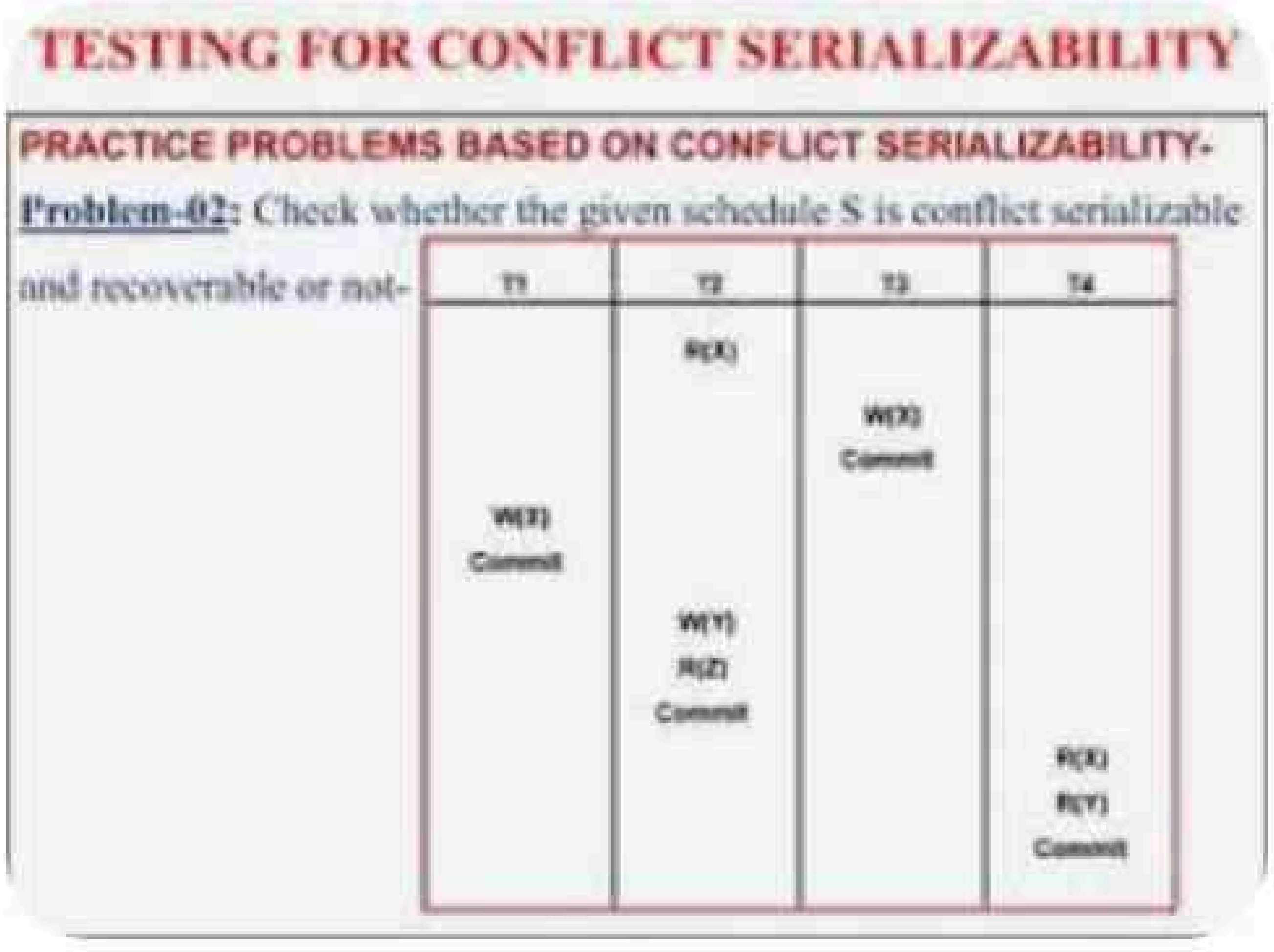
Visit >

Share

Save

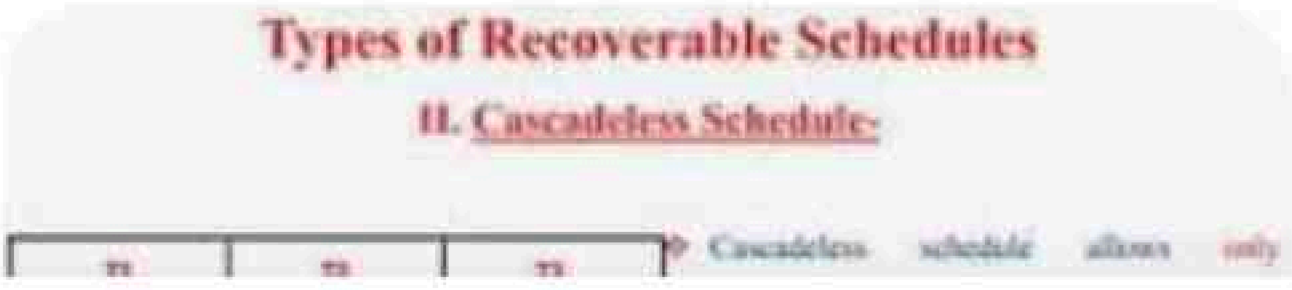
SlideShare

DBMS UNIT IV.pptx



SlideShare

DBMS UNIT IV.pptx



Discover

Search

Saved

DBMS UNIT IV.pptx

Janagi Raman S

TRANSACTION

Basic database access operations:

OPERATIONS	DESCRIPTIONS
Retrieve	To retrieve data stored in a database.
Insert	To store new data in database.
Delete	To delete existing data from database.
Update	To modify existing data in database.
Commit	To save the work done permanently.
Rollback	To undo the work done.

DBMS UNIT IV.pptx

Janagi Raman S

Transaction Execution with SQL

- Transaction support is provided by two SQL statements namely COMMIT and ROLLBACK.
- Transaction sequence is initiated by a user or an application program.
- It must continue through all succeeding SQL statements until one of the following four events occur:

1) A **COMMIT statement** is reached, in which case all changes are permanently recorded within the database.

The COMMIT statement automatically ends the SQL transaction.

The COMMIT operations indicates successful end-of-transaction.

col Projection → π

row selection → δ

join → $\bowtie$

$\pi_{Name}(\delta_{roll=1})(T1))$

Projection by default distinct এ কাজ করে।

1. $\pi_{Course\ Title}(\delta_{D\_C \cap C=3}(course))$

2. i instructor (ID, Name, dept_name salary)

student (ID, name, dep_name, total course)

advisor (S_ID, I_ID)

ins $\bowtie$ Ad $\bowtie$ st

$\pi_{St.ID}(\delta_{ins.Name = Eienstine}(ins \bowtie Adv \bowtie Stu)$

3. 6.12

$G_{max(cot)}(T2)$ 6.1.4.2

Highest, lowest বাদে বাকিগুলো দেখব

$$\pi_{col}\left(T1 - \left(G\,max(col) + G\,min(col)\right)\right)$$

Lecture-
DBMS
22-11-23

→ 2-phase locking — Growing Phase
— Shrinking Phase

(To recoverable)

→ Strict (Until 'Commit', no lock will be unlock)

→ Rigorous (Same as Strict, but ~~Unlock will be after commit~~ work for both X(A), S(A))

$\boxed{\text{Strict 2-phase locking}}$

T_1	T_2
X(A)	
R(A)	
W(A)	
	S(A)
	R(A)
	U(A)
	commit
S(B)	
R(B)	
Commit	
U(A)	
U(B)	

~~So~~ it is recoverable, serializable and cascadeless.

→ To remove deadlock — Prevention (for frequent deadlock)
— Detect and recovery

→ Prevention — Allocating resource before execution
— Resource ordering
— Time-stamp / Preemption

→ Time-stamp method (TS) → Wait-die
→ Wound-wait

Wait-die

if $TS(T_i) < TS(T_j)$

wait T_i

else → (Die situation)

rollback T_i

Wound-wait

if $TS(T_i) > TS(T_j)$

wait T_i

else → (Wound situation)

rollback T_j

→ In upgradle method of TS, we add 'time' to wait stage on of primary key

16+23.5
= 39

Lecture-
DBMS
20-11-23

Chap-15 (Concurrency control)

⇒ Locking Protocol
- Shared [S(A)] → (Only Read)
- Exclusive [X(A)] → Read, write - both work

⇒ Unlock (A)

	S	X
S	Yes	No
X	No	No

Compatibility table

Example:

T_1	T_2
X(B)	
r(B)	
B:= B-50	
w(B)	
Unlock (B)	
	S(A)
	r(A)
	r(B)
	display (A+B)
X(A)	
r(A)	
A := A-90	
w(A)	

Example:

T1:

X(B)
r(B)
B:=B-50
w(B)
Unlock (B)

x(A)
—
— —
Unlock (A)

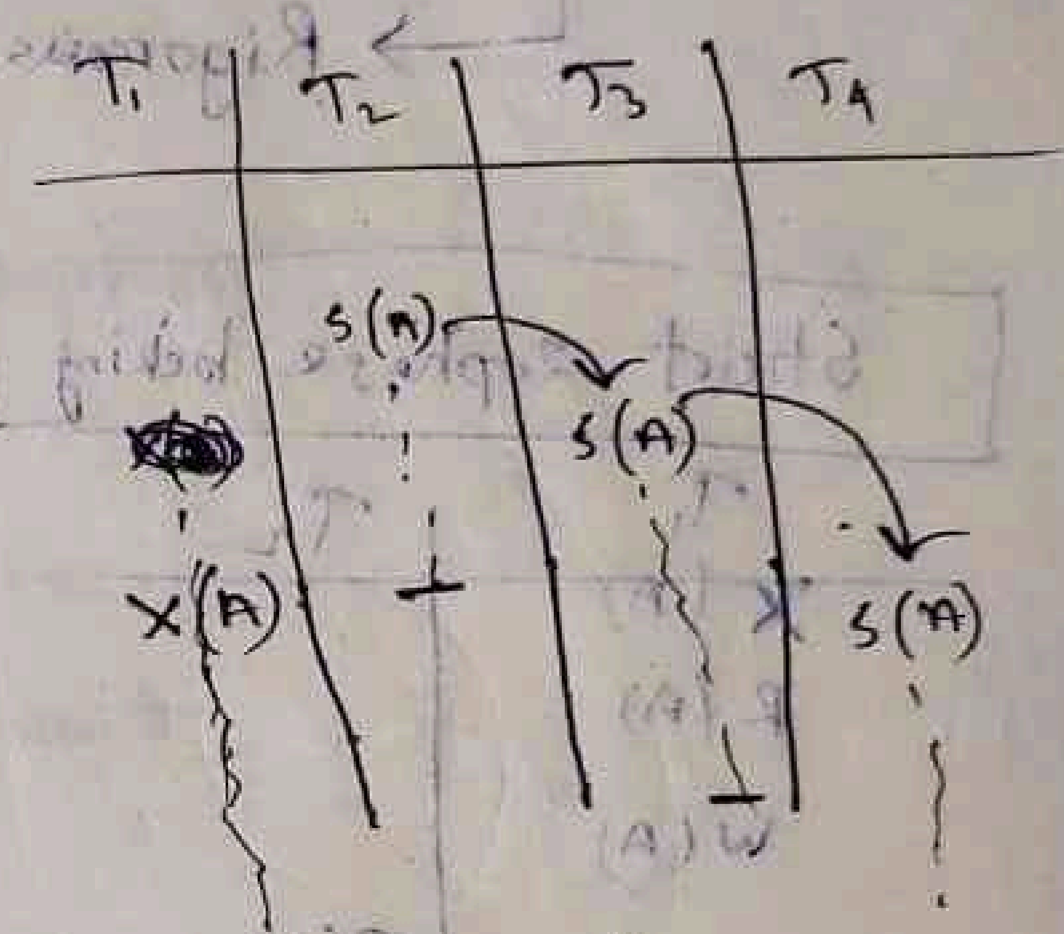
⇒ 'Unlock()' করা না হলে Same resource এ পুনরায় Lock করা যাবে ⇒ শুধুমাত্র S(A) to S(A) এর জন্য।

⇒ Problem of lock protocol

- May not serializable
- May not recoverable
- May be deadlock
- May be starvation
- May not cascadeless

T_1	T_2
X(A)	
	X(B)
	S(A)
S(B)	

Dead-lock



starvation

To serialize:

⇒ 2-phase locking protocol

- Growing phase (apply lock)
- Shrinking phase (apply unlock)

To recoverable:

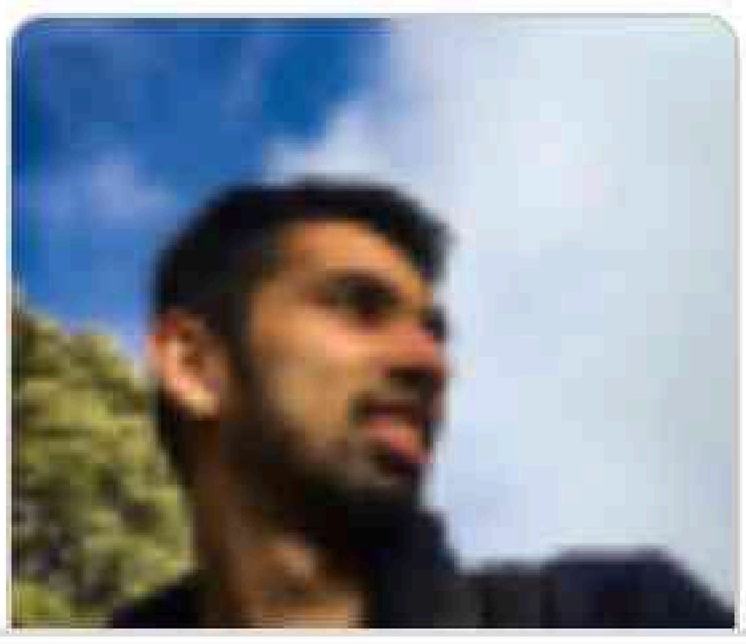
⇒ Be recoverable

- Untill "Commit", no operation (lock, Unlock) in any transaction

Conclusion

- Strict 2PL is the most restrictive form of 2PL and guarantees serializability, but may lead to decreased concurrency and increased contention for resources.
- Rigorous 2PL is similar to strict 2PL but allows for increased concurrency, but does not guarantee serializability and can be more difficult to implement.
- Conservative 2PL is a less restrictive form of 2PL that allows for maximum concurrency but does not guarantee consistency and isolation among transactions, and increases deadlock chances.

It's important to consider the requirements of the system before deciding which type of 2PL to use. It is also important to ensure that the system remains in a consistent state and to take measures to prevent deadlocks while using Conservative 2PL.



Raunak Jain

Types

Two phase locking is of two types −

Strict two phase locking protocol

A transaction can release a shared lock after the lock point, but it cannot release any exclusive lock until the transaction commits. This protocol creates a cascade less schedule.

Cascading schedule: In this schedule one transaction is dependent on another transaction. So if one has to rollback then the other has to rollback.

Rigorous two phase locking protocol

A transaction cannot release any lock either shared or exclusive until it commits.

The 2PL protocol guarantees serializability, but cannot guarantee that deadlock will not happen.

Example

Let T1 and T2 are two transactions.

T1=A+B and T2=B+A

T1	T2
Lock-X(A)	Lock-X(B)
Read A;	Read B;

Example

Let T1 and T2 are two transactions.

T1=A+B and T2=B+A

T1	T2
Lock-X(A)	Lock-X(B)
Read A;	Read B;
Lock-X(B)	Lock-X(A)

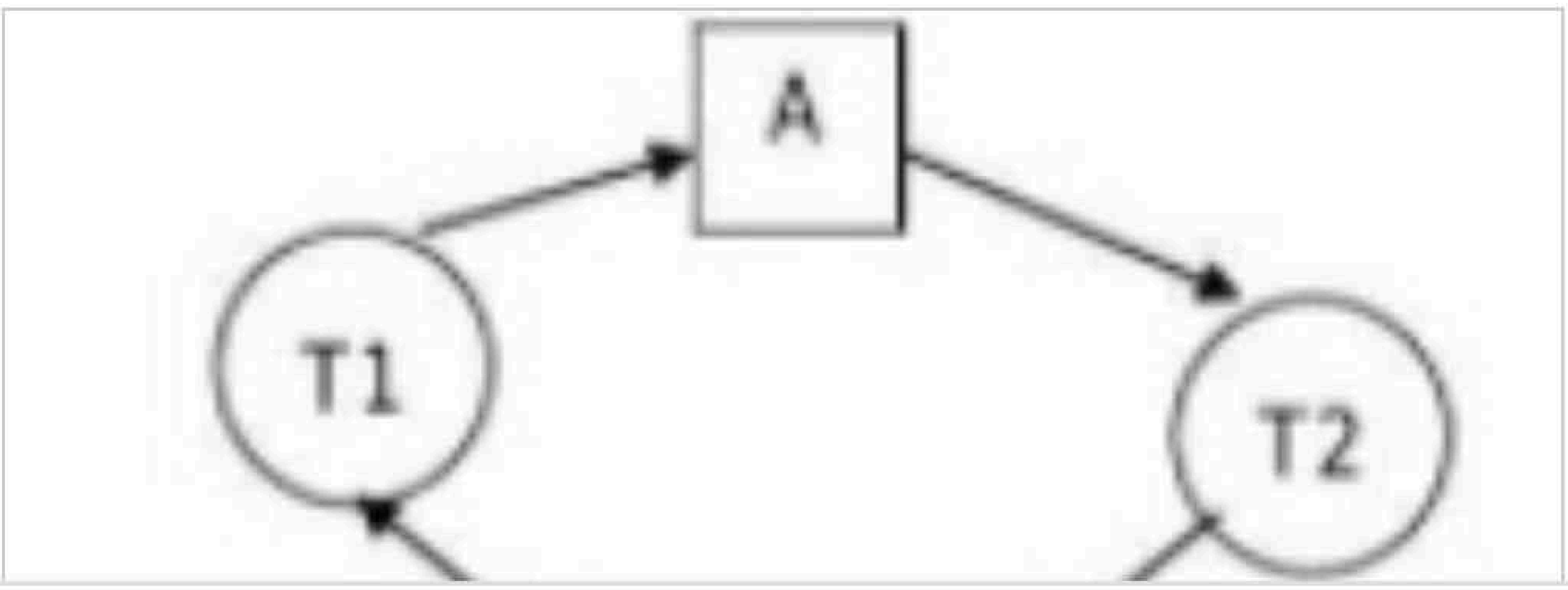
Here,

Lock-X(B) : Cannot execute Lock-X(B) since B is locked by T2.

Lock-X(A) : Cannot execute Lock-X(A) since A is locked by T1.

In the above situation T1 waits for B and T2 waits for A. The waiting time never ends. Both the transaction cannot proceed further at least any one releases the lock voluntarily. This situation is called deadlock.

The wait for graph is as follows −



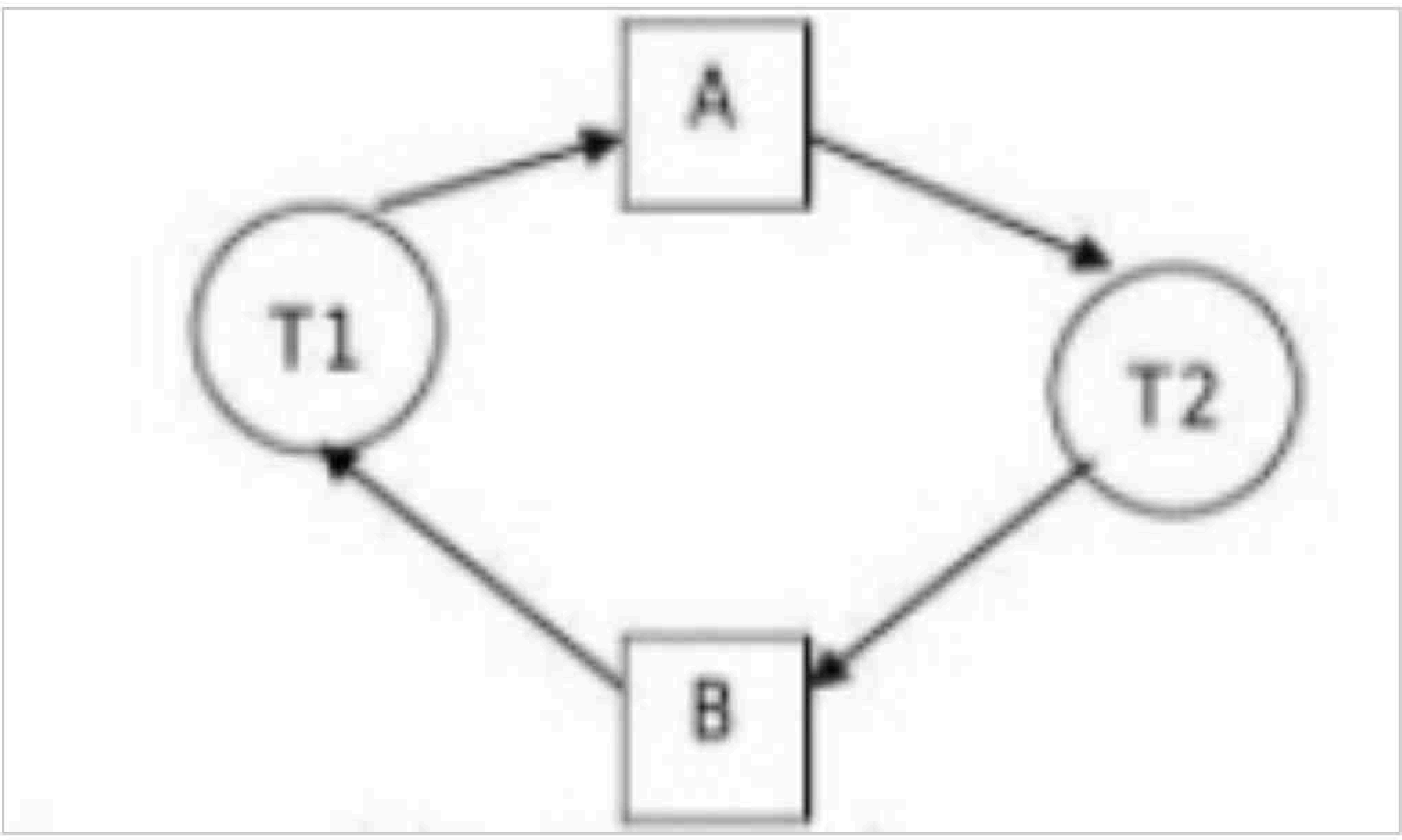
Here,

Lock-X(B) : Cannot execute Lock-X(B) since B is locked by T2.

Lock-X(A) : Cannot execute Lock-X(A) since A is locked by T1.

In the above situation T1 waits for B and T2 waits for A. The waiting time never ends. Both the transaction cannot proceed further at least any one releases the lock voluntarily. This situation is called deadlock.

The wait for graph is as follows −



Wait for graph: It is used in the deadlock detection method, creating a node for each transaction, creating an edge Ti to Tj, if Ti is waiting to lock an item locked by Tj. A cycle in WFG indicates a deadlock has occurred. WFG is created at regular intervals.

is there any algorithm to o...

Search instead for is there any algorithm to optimize disk block access?? if yes,explan algorithm.

বাঙলায় অনুসন্ধান করুন

ডিস্ক ব্লক অ্যাক্সেস অপ্টিমাইজ ...

বাংলায় In English

Optimization techniques besides buffering of blocks in main memory. Scheduling: If several blocks from a cylinder need to be transferred, we may save time by requesting them in the order in which they pass under the heads. A commonly used disk-arm scheduling algorithm is the elevator algorithm.

https://www2.cs.sfu.ca › node6

Optimization of Disk-Block Access

About featured snippets Feedback

People also ask

Optimization of Disk-Block Access

People also ask

What is a disk block in a magnetic disk?

The surface of a platter is organised as a number of concentric tracks. Each track is divided into sectors. The information held in one sector, a block, is **the unit of transfer between the disk and primary memory** (typically 4K bytes or so). The operating system determines where the blocks for each file are placed.

http://www.cs.man.ac.uk › node4

Magnetic disk storage

MORE RESULTS

What is disk block in DBMS?

What is disk block allocation in Unix?

How do I optimize disk block access?

T_1	T_2
X(A)	
R(A)	
W(A)	
U(A)	
	S(A)
	R(A)
	U(A)
S(B)	
R(B)	
U(A)	

→ serializability
→ Recoverable
→ Dead lock
→ Starvation
→ ~~Case cadeless~~
→ Cascadeless

Share and Exclusive
S(A) X(A)

2 Phase Locking → Growing
→ Shrinking

Dead lock → Prevention
→ Detect & Recovery

Exercise - 4

R(A, B, C, D)

FD = {AB → C, AB → D, C → A, D → B}

$A^+ = A$

$B^+ = B$

$C^+ = C$
$= CA$

$D^+ = D$
$= DB$

$AB^+ = AB$
$= ABC$
$= ABCD$

C → A

$CB^+ = CB$
$= CBA$
$= CBAD$

CD → AB

C → A D → B

$CD^+ = CD$
$= CDA$
$= CDAB$

AB → D

$AD^+ = AD$
$= ADB$
$= ADBC$

CK = {AB, CB, CD, AD}

PA = {A, B, C, D}

NPA = {Ø}

2NF: → Already

3NF: AB → C, AB → D, C → A, D → B

L.H.S L.H.S R.H.S. R.H.S. → condition true → Already 3NF

BCNF: C → A, D → B → voilations BCNF

R_1(CA), R_2(BD), R_3(C, D)

$R_1 \cap R_3 = C$; → Lossless

$R_2 \cap R_3 = D$; → Lossless

5.(c) R(A,B,C,D,E,H)

FD{A→B, BC→D, E→C, D→A}

DEH$^+$

D→A → AEH$^+$

⇒ × EH$^+$ = EH
= EHC

✓ AEH$^+$ = AEH
= AEHC
= ABEHC
= ABEHCD

✓ DEH$^+$ = DEH
= DEHA
= DEHAC
= DEHACB

CEH$^+$ BEH$^+$

BC→D → DEH$^+$

✓ BEH$^+$ = BEH
= BEHC
= BEHCD
= BEHCDA

× CEH$^+$ = CEH

CK={AEH, BEH, DEH}

PA={A, B, D, E, H}

NPA={C}

Slide→chapter-8

Exercise: Lossless Join

R{A, B, C, D, E}

FD={A→BC, CD→E, B→D, E→A}

⇒ ✓ A$^+$ = A
= ABC
= ABCD
= ABCDE

✓ E$^+$ = E
= EA
= EABC
= EABCD

CK={A, E}

PA={A, E}

NPA={B, C, D}

× B$^+$ = B
= BD

× C$^+$ = C

× D$^+$ = D

2NF: $A \to BC$
PD $\to R_1(ABC)\ R_2(DEA)$

3NF: $R_1(ABC)\ R_2(DEA) \to R_1 \cap R_2 = A \to$ A is a prime Attribute/ CK/superkey $\to$ Lossless

$A \to BC$, $E \to A$, $B \to D$, $CD \to E$

$\to$ 3NF $\to$ Already exists.

BCNF: $R_1(ABC)\ R_2(DEA)$

$A \to BC$, $E \to A$, $\times\ CD \to E \to R_1(CDE)\ R_2(ABC)$

$R_1 \cap R_2 = C$; C is not super key $\to$ Lossy/ Not lossless

Exercise 3:

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

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

$\Rightarrow B^+ = B$
$= BC$
$= BCA$
$= BCAD$

$CK = \{B\}$

$PA = \{B\}$

$NPA = \{A, C, D\}$

1NF: A relation R is in 1NF because all values are single valued and atomic.

2NF: A relation R is in 1NF and no partial dependency, so it is already 2NF.

3NF: A relation R is in 2NF,
No TD $\to B \to C \to$ Already 3NF, $C \to DA$ [union]

$R_1(BC)\quad R_2(ADC) \to R_1 \cap R_2 = C \to$ Lossy

BCNF: A relation R is in 3NF

$R_1(BC), R_2(CA), R_3(CD)$ $\boxed{B \to C}$ already BCNF

$R_1 \cap R_2 = C \to$ Lossy $\quad R_1 \cap R_3 = C \to$ Lossy

$R_2 \cap R_3 = C \to$ Lossy

18:02 95

jemi39shifa@gmail.com Switch accounts

Not shared

Draft saved

* Indicates required question

Full Name (You will get certificate with this name) *

Habiba Akter

Email *

jemi39shifa@gmail.com

HackerRank User Name *

Habiba Shifa

Your Academic Institution Name (School, College, or University Name) *

Noakhali Science and Technology University

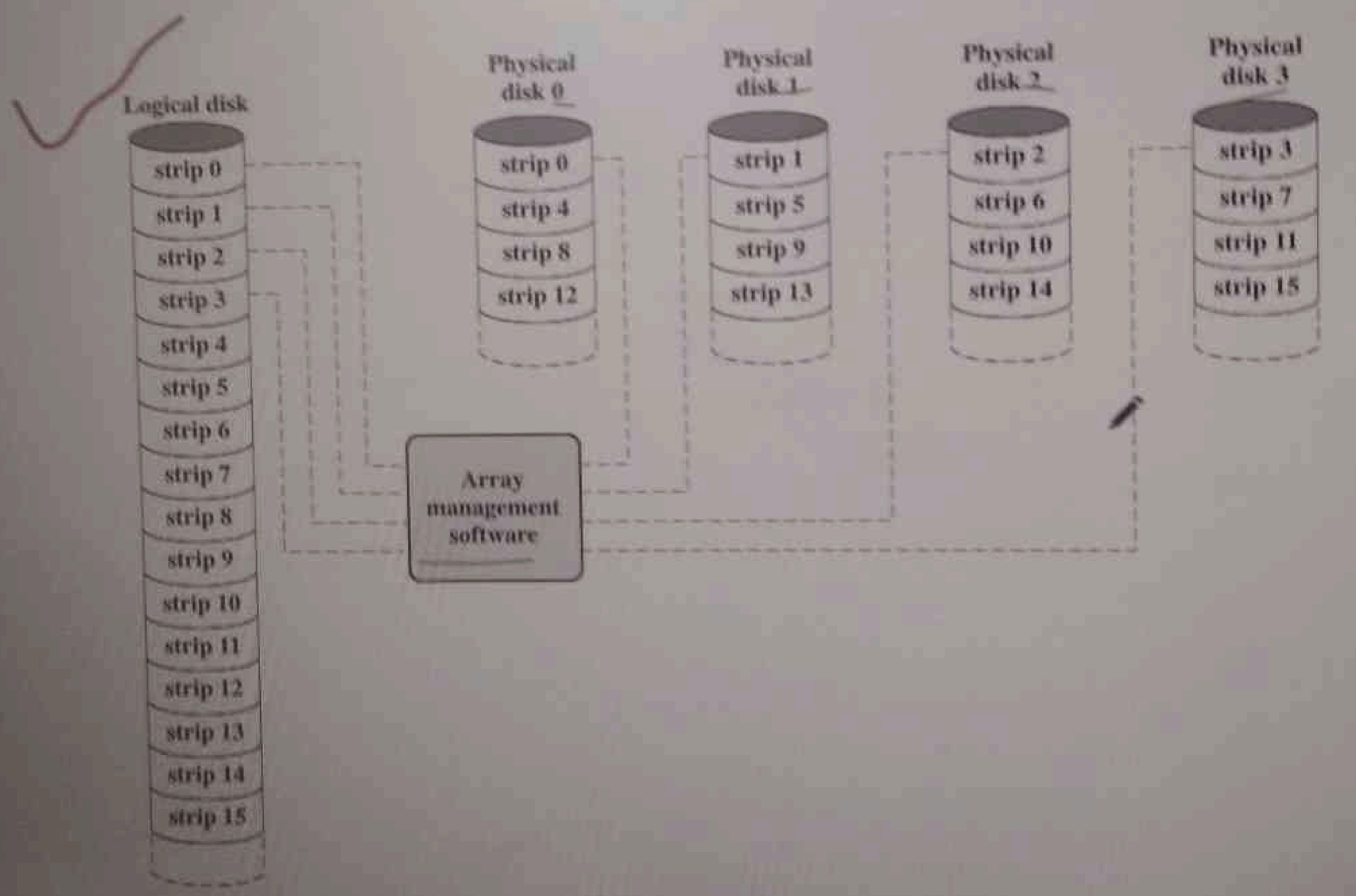


Figure 6.9 Data Mapping for a RAID Level 0 Array

Figure 6.7 Timing of a Disk I/O Transfer

Disk Performance Parameters

The actual details of disk I/O operation depend on the computer system, the operating system, and the nature of the I/O channel and disk controller hardware. A general timing diagram of disk I/O transfer is shown in Figure 6.7.

When the disk drive is operating, the disk is rotating at constant speed. To read or write, the head must be positioned at the desired track and at the beginning of the desired sector on that track. Track selection involves moving the head in a movable-head system or electronically selecting one head on a fixed-head system. On a movable-head system, the time it takes to position the head at the track is known as **seek time**. In either case, once the track is selected, the disk controller waits until the appropriate sector rotates to line up with the head. The time it takes for the beginning of the sector to reach the head is known as **rotational delay**, or *rotational latency*. The sum of the seek time, if any, and the rotational delay equals the **access time**, which is the time it takes to get into position to read or write. Once the head is in position, the read or write operation is then performed as the sector [illegible] under the head; this is the data transfer portion of the operation; the time required for the transfer is the **transfer time**.

In addition to the access time and transfer time, there are several queuing delays normally associated with a disk I/O operation. When a process issues an I/O request, it must first wait in a queue for the device to be available. At that time, the device is assigned to the process. If the device shares a single I/O channel or a set of I/O channels with other disk drives, then there may be an additional wait for the channel to be available. At that point, the seek is performed to begin disk access.

In some high-end systems for servers, a technique known as rotational positional sensing (RPS) is used. This works as follows: When the seek command has been issued, the channel is released to handle other I/O operations. When the seek mon size today is 3.5 inches (8.9 cm), reducing the distance that the arm has to travel. A typical average seek time on contemporary hard disks is under 10 ms.

ROTATIONAL DELAY Disks, other than floppy disks, rotate at speeds ranging from 3600 rpm (for handheld devices such as digital cameras) up to, as of this writing, 20,000 rpm; at this latter speed, there is one revolution per 3 ms. Thus, on the average, the rotational delay will be 1.5 ms.

TRANSFER TIME The transfer time to or from the disk depends on the rotation speed of the disk in the following fashion:

$$\boxed{T = \frac{b}{rN}}$$

where

T = transfer time

b = number of bytes to be transferred

N = number of bytes on a track

r = rotation speed, in revolutions per second

Thus the total average access time can be expressed as

$$\boxed{T_a = T_s + \frac{1}{2r} + \frac{b}{rN}}$$

where T_s is the average seek time. Note that on a zoned drive, the number of bytes per track is variable, complicating the calculation.[2]

A TIMING COMPARISON With the foregoing parameters defined, let us look at two different I/O operations that illustrate the danger of relying on average values. Consider a disk with an advertised average seek time of 4 ms, rotation speed of 15,000 rpm, and 512-byte sectors with 500 sectors per track. Suppose that we wish to read a file consisting of 2500 sectors for a total of 1.28 Mbytes. We would like to estimate the can be executed in parallel if the block of data to be accessed is distributed across multiple disks.

With the use of multiple disks, there is a wide variety of ways in which the data can be organized and in which redundancy can be added to improve reliability. This could make it difficult to develop database schemes that are usable on a number of platforms and operating systems. Fortunately, industry has agreed on a standardized scheme for multiple-disk database design, known as RAID (Redundant Array of Independent Disks). The RAID scheme consists of seven levels,[2] zero through six. These levels do not imply a hierarchical relationship but designate different design architectures that share three common characteristics:

1. RAID is a set of physical disk drives viewed by the operating system as a single logical drive.
2. Data are distributed across the physical drives of an array in a scheme known as striping, described subsequently.
3. Redundant disk capacity is used to store parity information, which guarantees data recoverability in case of a disk failure.

The details of the second and third characteristics differ for the different RAID levels. RAID 0 and RAID 1 do not support the third characteristic.

The term RAID was originally coined in a paper by a group of researchers at the University of California at Berkeley [PATT88].[3] The paper outlined various RAID configurations and applications and introduced the definitions of the RAID levels that are still used. The RAID strategy employs multiple disk drives and distributes data in such a way as to enable simultaneous access to data from multiple drives, thereby improving I/O performance and allowing easier incremental increases in capacity.

The unique contribution of the RAID proposal is to address effectively the need for redundancy. Although allowing multiple heads and actuators to operate

Table 6.3 RAID Levels

Category	Level	Description	Disks Required	Data Availability	Large I/O Data Transfer Capacity	Small I/O Request Rate
Striping	0	Nonredundant	N	Lower than single disk	Very high	Very high for both read and write
Mirroring	1	Mirrored	$2N$	Higher than RAID 2, 3, 4, or 5; lower than RAID 6	Higher than single disk for read; similar to single disk for write	Up to twice that of a single disk for read; similar to single disk for write
Parallel access	2	Redundant via Hamming code	$N + m$	Much higher than single disk; comparable to RAID 3, 4, or 5	Highest of all listed alternatives	Approximately twice that of a single disk
	3	Bit-interleaved parity	$N + 1$	Much higher than single disk; comparable to RAID 2, 4, or 5	Highest of all listed alternatives	Approximately twice that of a single disk
Independent access	4	Block-interleaved parity	$N + 1$	Much higher than single disk; comparable to RAID 2, 3, or 5	Similar to RAID 0 for read; significantly lower than single disk for write	Similar to RAID 0 for read; significantly lower than single disk for write
	5	Block-interleaved distributed parity	$N + 1$	Much higher than single disk; comparable to RAID 2, 3, or 4	Similar to RAID 0 for read; lower than single disk for write	Similar to RAID 0 for read; generally lower than single disk for write
	6	Block-interleaved dual distributed parity	$N + 2$	Highest of all listed alternatives	Similar to RAID 0 for read; lower than RAID 5 for write	Similar to RAID 0 for read; significantly lower than RAID 5 for write

N = number of data disks; m proportional to log N

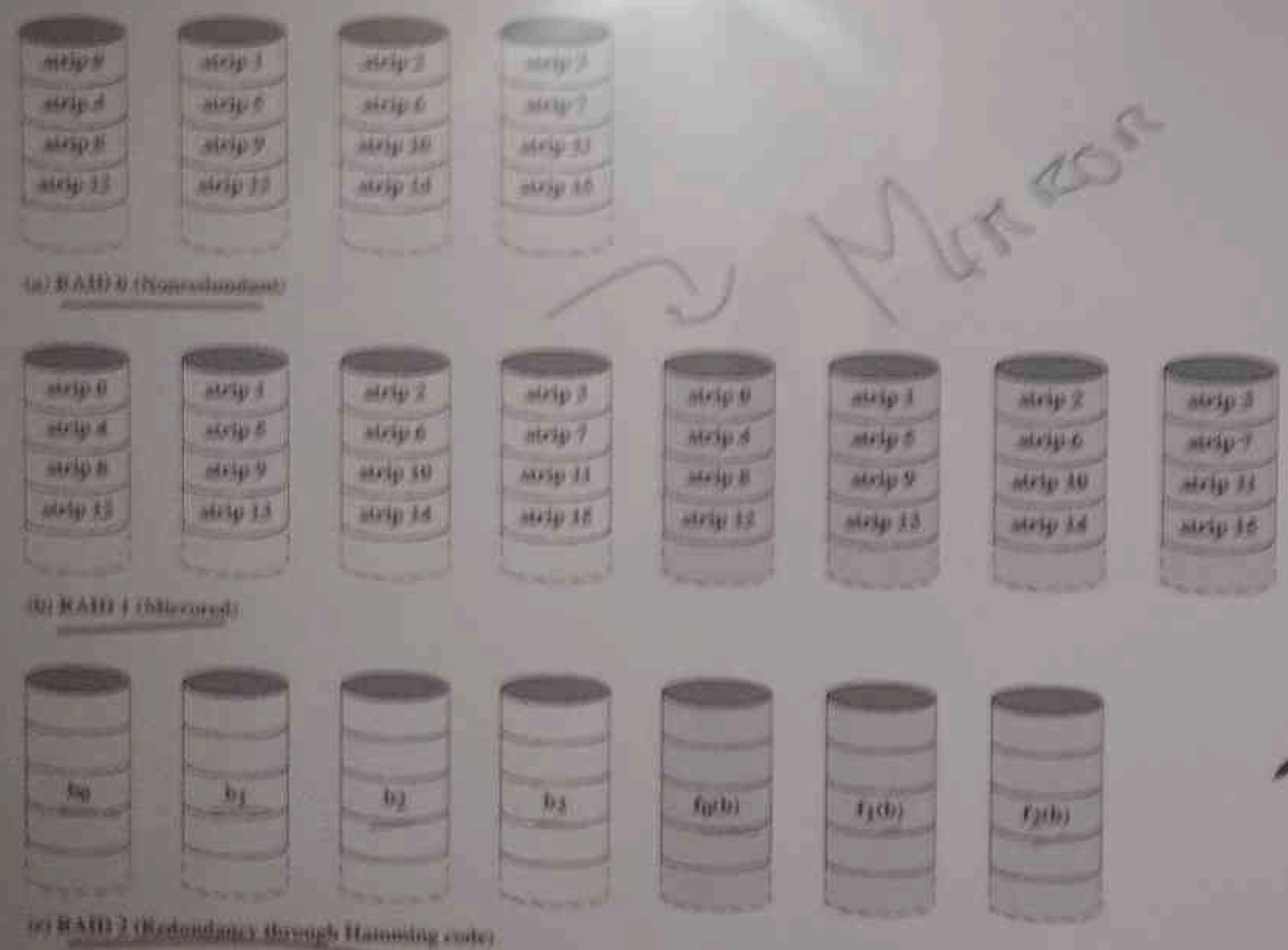
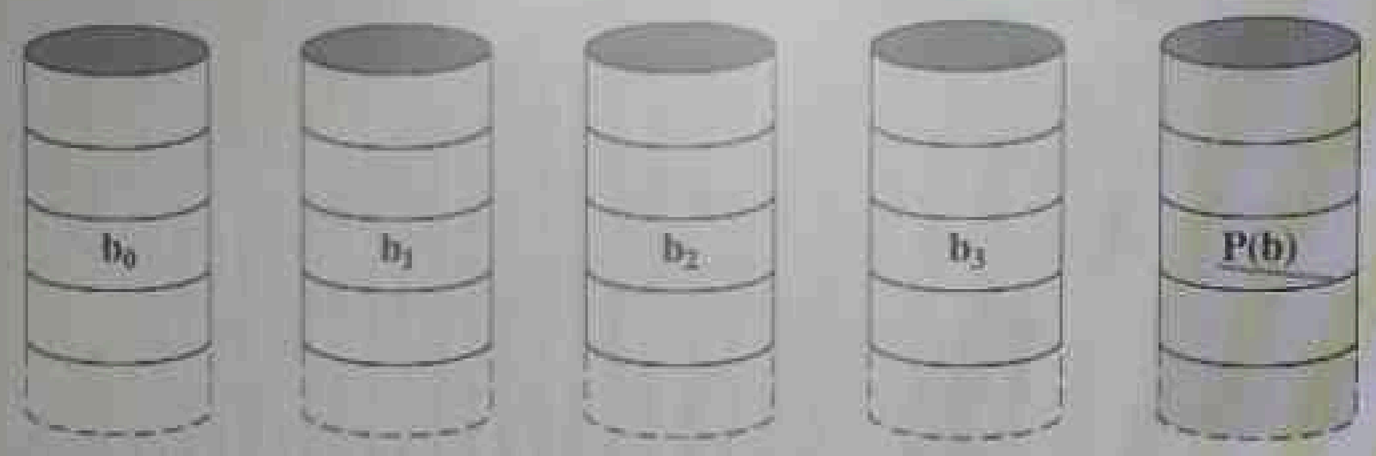
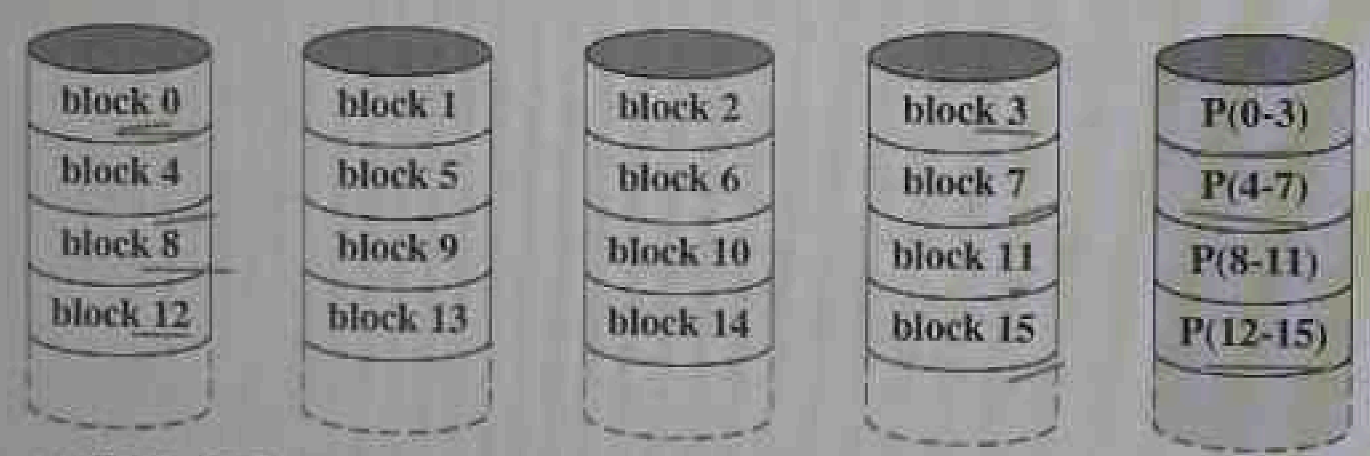


Figure 6.8 RAID Levels

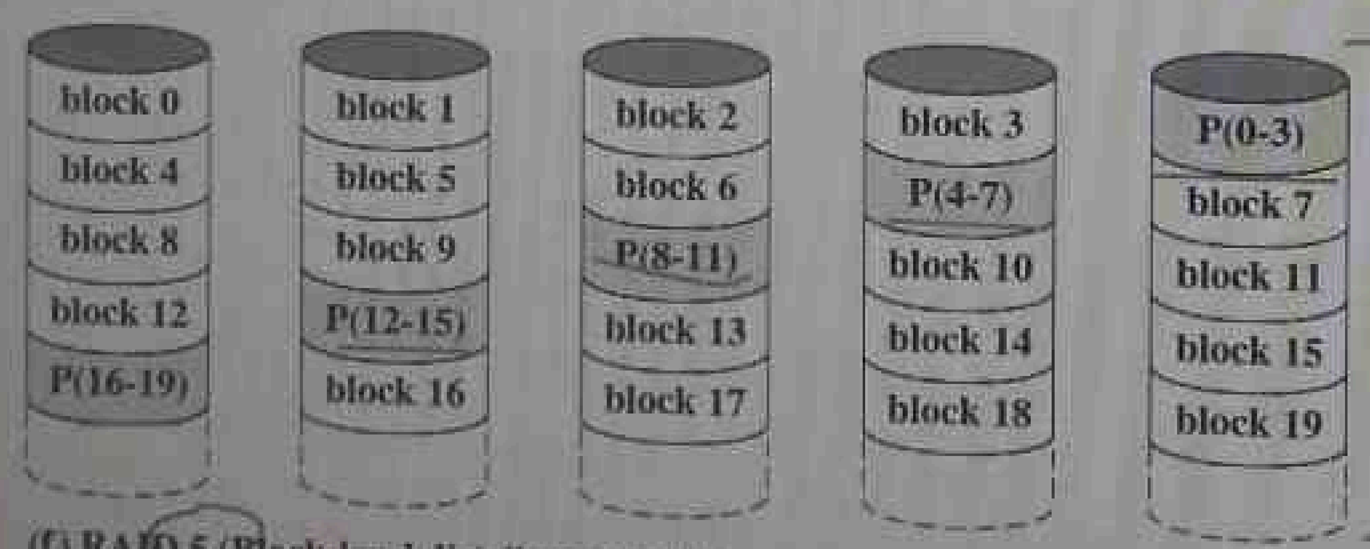
metrics. Each RAID level's strong point is highlighted by darker shading. Figure 6.8



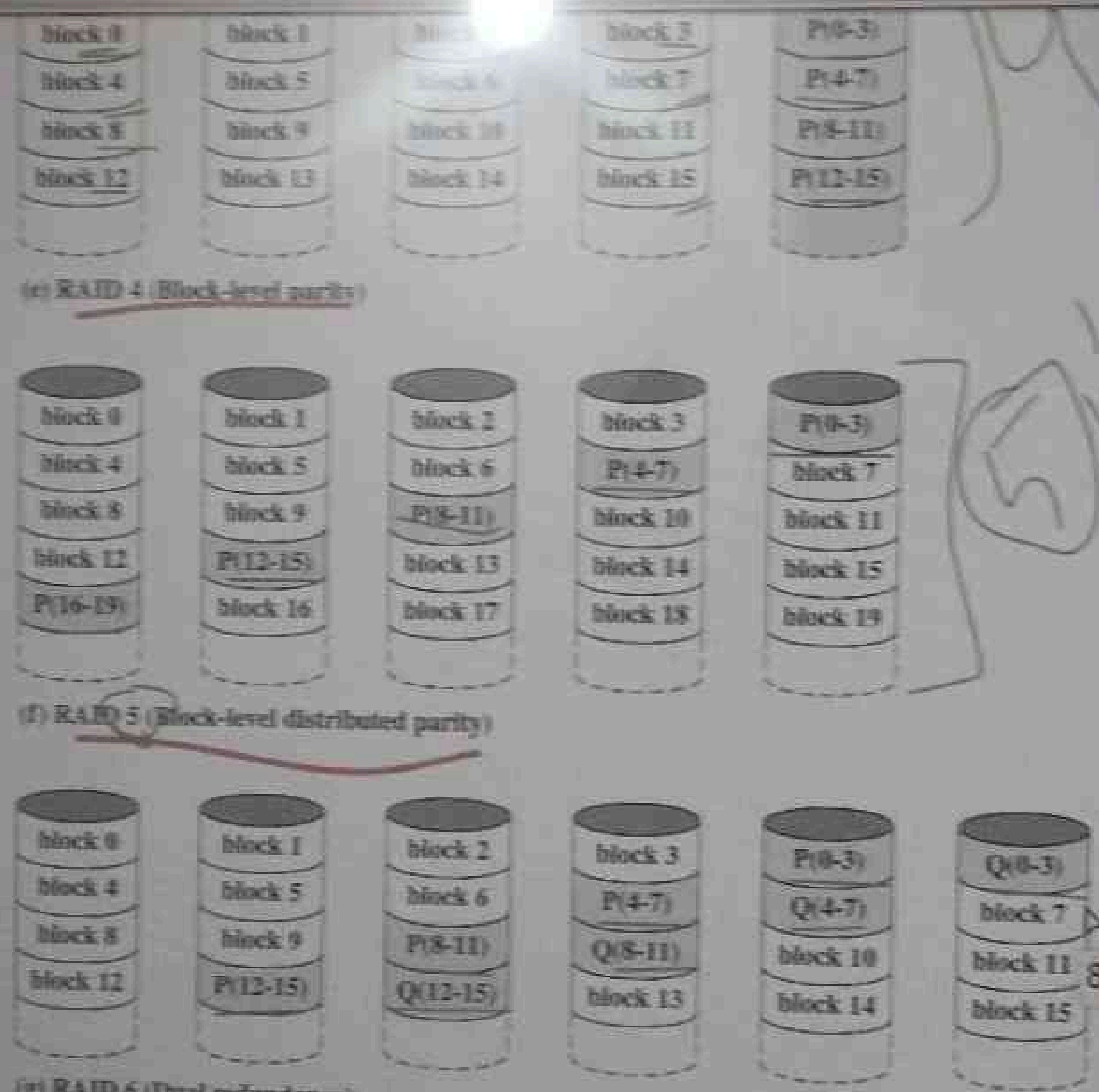
(d) RAID 3 (Bit-interleaved parity)



(e) RAID 4 (Block-level parity)



(f) RAID 5 (Block-level distributed parity)



(e) RAID 4 (Block-level parity)

(f) RAID 5 (Block-level distributed parity)

(g) RAID 6 (Dual redundancy)

Figure 6.8 RAID Levels (continued)

N = number of bytes on a track

r = rotation speed, in revolutions per second

Thus the total average access time can be expressed as

$$\boxed{T_a = T_s + \frac{1}{2r} + \frac{b}{rN}}$$

T

where T_s is the average seek time. Note that on a zoned drive, the number of bytes per track is variable, complicating the calculation.[2]

A TIMING COMPARISON With the foregoing parameters defined, let us look at two different I/O operations that illustrate the danger of relying on average values. Consider a disk with an advertised average seek time of 4 ms, rotation speed of 15,000 rpm, and 512-byte sectors with 500 sectors per track. Suppose that we wish to read a file consisting of 2500 sectors for a total of 1.28 Mbytes. We would like to estimate the total time for the transfer.

First, let us assume that the file is stored as compactly as possible on the disk. That is, the file occupies all of the sectors on 5 adjacent tracks (5 tracks × 500 sectors/track = 2500 sectors). This is known as *sequential organization*. Now, the time to read the first track is as follows:

Average seek	4 ms
Average rotational delay	2 ms
Read 500 sectors	4 ms
	10 ms

[2]Compare the two preceding equations to Equation (4.1).

Suppose that the remaining tracks can now be read with essentially no seek time. That is, the I/O operation can keep up with the flow from the disk. Then, at most, we need to deal with rotational delay for each succeeding track. Thus each successive track is read in 2 + 4 = 6 ms. To read the entire file,

$$\text{Total time} = 10 + (4 \times 6) = 34 \text{ ms} = 0.034 \text{ seconds}$$

Now let us calculate the time required to read the same data using random access rather than sequential access; that is, accesses to the sectors are distributed randomly over the disk. For each sector, we have

Average seek	4 ms
Rotational delay	2 ms
Read 1 sectors	0.008 ms
	6.008 ms

$$\text{Total time} = 2500 \times 6.008 = 15020 \text{ ms} = 15.02 \text{ seconds}$$

It is clear that the order in which sectors are read from the disk has a tremendous effect on I/O performance. In the case of file access in which multiple sectors are read or written, we have some control over the way in which sectors of data are deployed. However, even in the case of a file access, in a multiprogramming environment, there will be I/O requests competing for the same disk. Thus, it is worthwhile to examine ways in which the performance of disk I/O can be improved over that achieved with purely random access to the disk. This leads to a consideration of disk scheduling algorithms, which is the province of the operating system and beyond the scope of this book (see [STAL09] for a discussion).

(c) Explain the partial and total participation constraints with example.

7. (a) Explain the converting procedure of Non-Binary Relationships to Binary Form. 3
(b) Explain Closure of Set of Functional dependency and Closure of Attribute sets 2
(c) Consider the universal relation R = {A,B,C,D,E}. F = {A→BC, CD →E, B →D, E→A}. Is the above decomposition a lossless join or not? 3
(d) What is transitive dependency? 2

8. (a) What is the need of the normalization? Explain the first three steps involed in the normalization. 2
(b) Consider the universal relation R = (A, B, C, D) and the set of functional Dependencies F = {AB→C, AB→D, C→A, D→B}. 4
i) Is R in 3NF, why? If it is not, decompose it into 3NF.
ii) Is R in BCNF, why? If it is not, decompose it into BCNF
(c) What is lossy and lossless decomposition? Explain with example

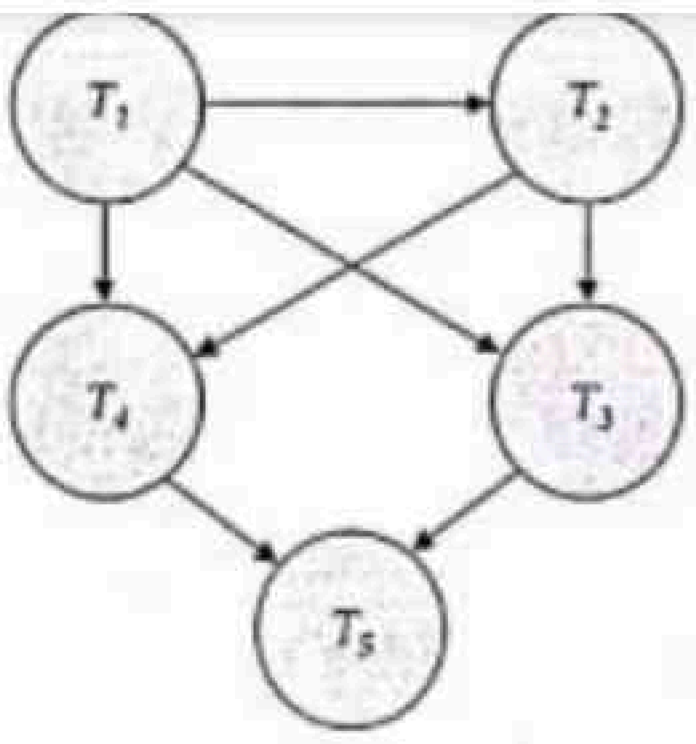
9. (a) Between SAN and NAS, explain which one is comparatively better. 4
(b) Is there any algorithm to optimize Disk-Block Access? If yes, explain the algorithm. 2
(c) Define RAID and describe its all level from 0 to 6. 5

(d) Probably you know Cloud databases and Object-oriented databases. Can you compare these two along with their advantages, disadvantages and techniques? Discuss your answer. 1+3

7. (a) What are the two major pitfalls in designing a database schema and how these can be avoided? 3
(b) Differentiate between the partial and total participation constraints with examples. 3
(c) Discuss different types of the mapping cardinality for a binary relationship set R between entity sets A and B with example. 4

8. (a) Distinguish between 3NF and BCNF with appropriate examples. 2
(b) Consider a relational schema R = {A, B, C, D, X, Y} and the decomposed table R1 = {B, X} and R2 = {A, C, Y, D, X} and FD = {A → B, C → {D, X}, {A, C} → Y}. Decide whether the given decomposition of R, R1 and R2 is lossless or lossy decomposition? 4
(c) Consider a relational schema R = (A, B, C, D) and functional dependencies: F = {C→D, C→A, B→C}. 4
  a) Construct the best normal form that R satisfies.
  b) Decompose R into a set of BCNF relations.

9. (a) Explain variable-length records using slotted page structure. 2
(b) How can we calculate the Access time, Data-transfer rate and Mean time to failure (MTTF) for measuring the disk performance? Implement with example. 3
(c) Define RAID and describe its level 0, 1 and 5. 5



9. a) What is a cascadeless schedule? Why is cascadelessness of schedules desirable? 3

b) Are there any circumstances under which it would be desirable to allow noncascadeless schedules? Explain your answer. 2

c) Consider the following two transactions: 5

$T1$: read(A);
read(B);
if $A = 0$ **then** $B := B + 1$;
write(B).
$T2$: read(B);
read(A);
if $B = 0$ **then** $A := A + 1$;
write(A).

Add lock and unlock instructions to transactions $T1$ and $T2$, so that they observe the two-phase locking protocol. Can the execution of these transactions result in a deadlock?

a. Lock and unlock instructions:

T_{31}: **lock-S**(A)
read(A)
lock-X(B)
read(B)
if $A = 0$
then $B := B + 1$
write(B)
unlock(A)
unlock(B)

T_{32}: **lock-S**(B)
read(B)
lock-X(A)
read(A)
if $B = 0$
then $A := A + 1$
write(A)
unlock(B)
unlock(A)

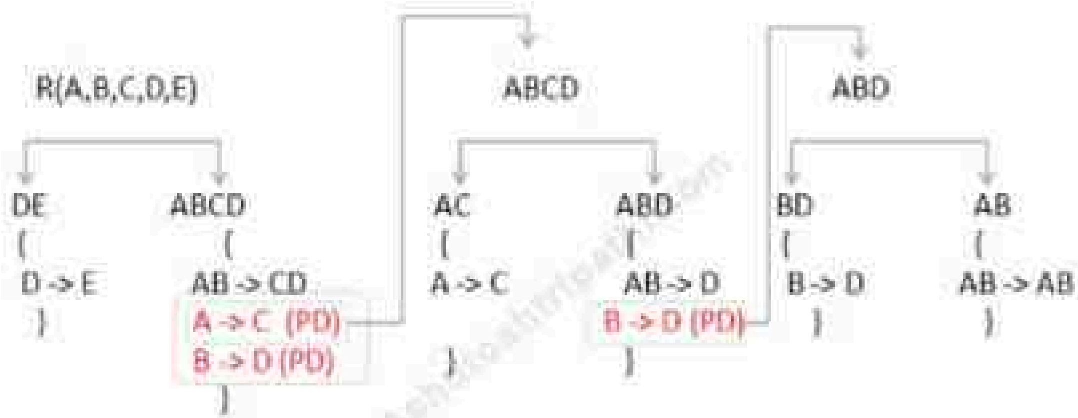
b. Execution of these transactions can result in deadlock. For example, consider the following partial schedule:

T_{31}	T_{32}
lock-S(A)	
	lock-S(B)
	read(B)
read(A)	
lock-X(B)	
	lock-X(A)

The transactions are now deadlocked.

Data Science Duniya

R(A, B, C, D, E)
Functional Dependencies {AB -> CD, D -> E, A -> C,B -> D}



Final BCNF Decomposition will be:
D(DE{D->E} | AC{A->C} | BD{B -> D} | AB{AB - > AB})

*PD : Partial Dependency

3:49 ⋯ 94

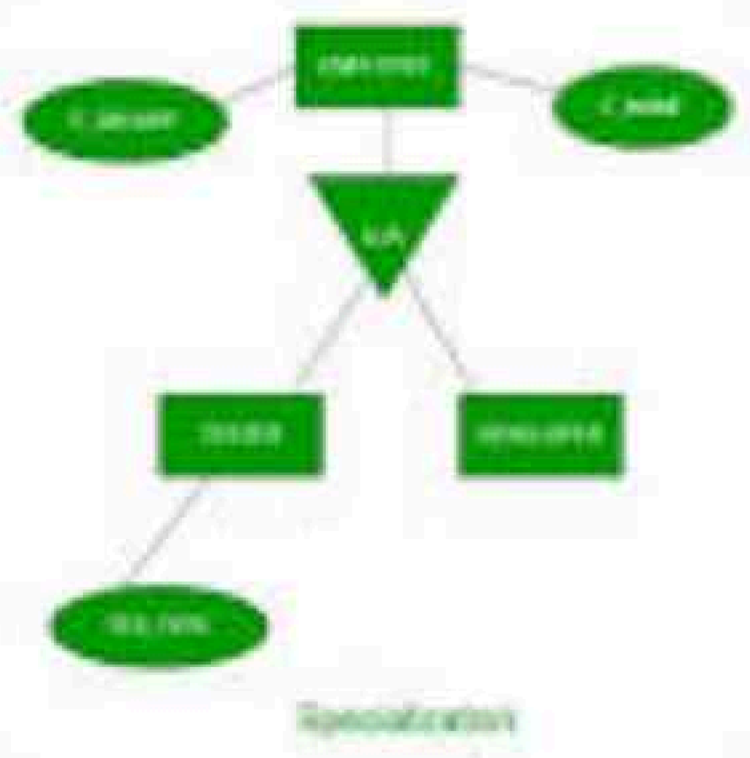
Google

specialization with exampl...

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন ডিবিএমএসে উদাহরণ সহ বিশে..

বাংলায় In English



In specialization, an entity is divided into sub-entities based on its characteristics. It is a top-down approach where the higher-level entity is specialized into two or more lower-level entities. For Example, an EMPLOYEE entity in an Employee management system can be specialized into DEVELOPER, TESTER, etc. Sep 19, 2023

https://www.geeksforgeeks.org › ge...

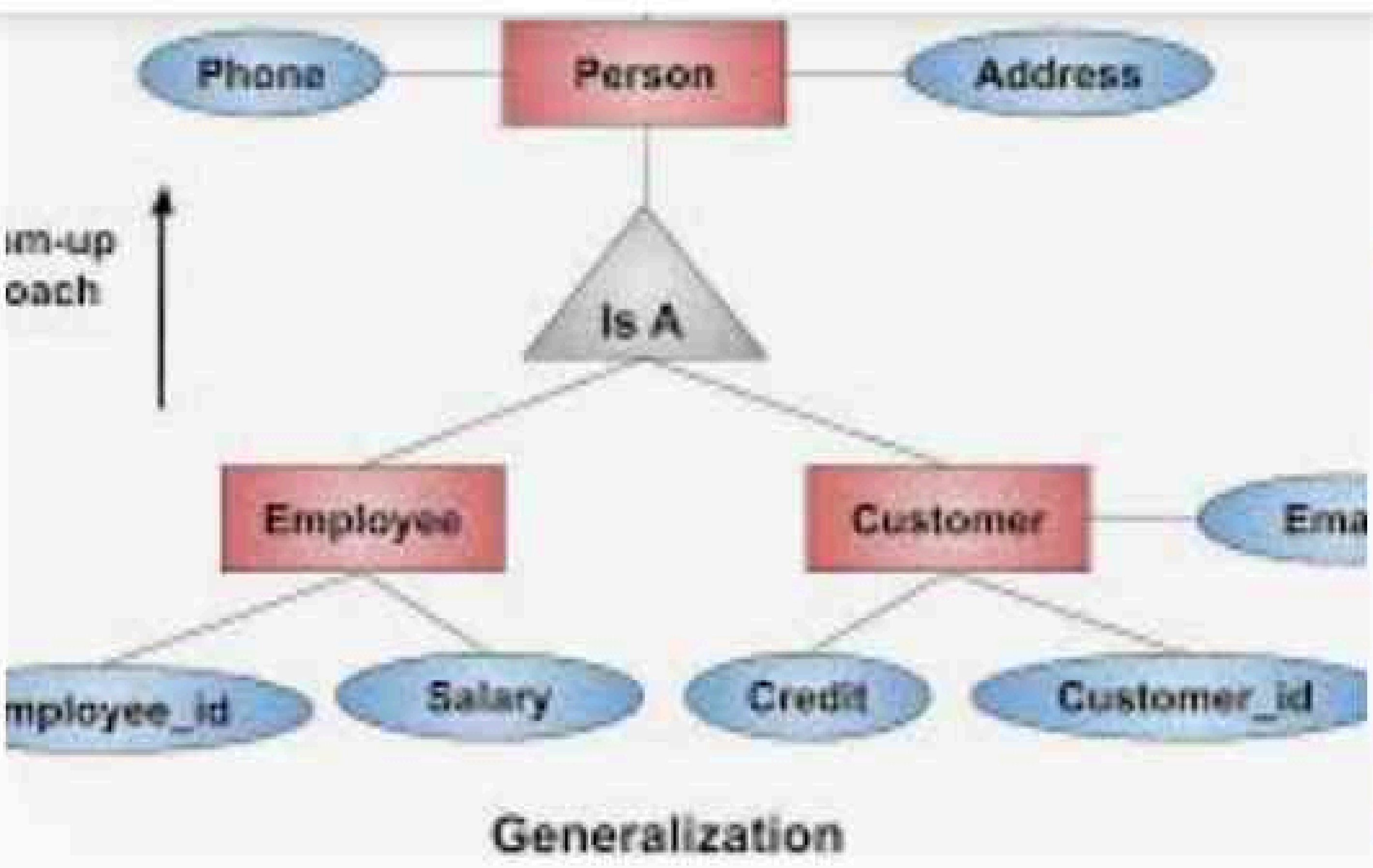
Generalization, Specialization and Aggregation in ER Model

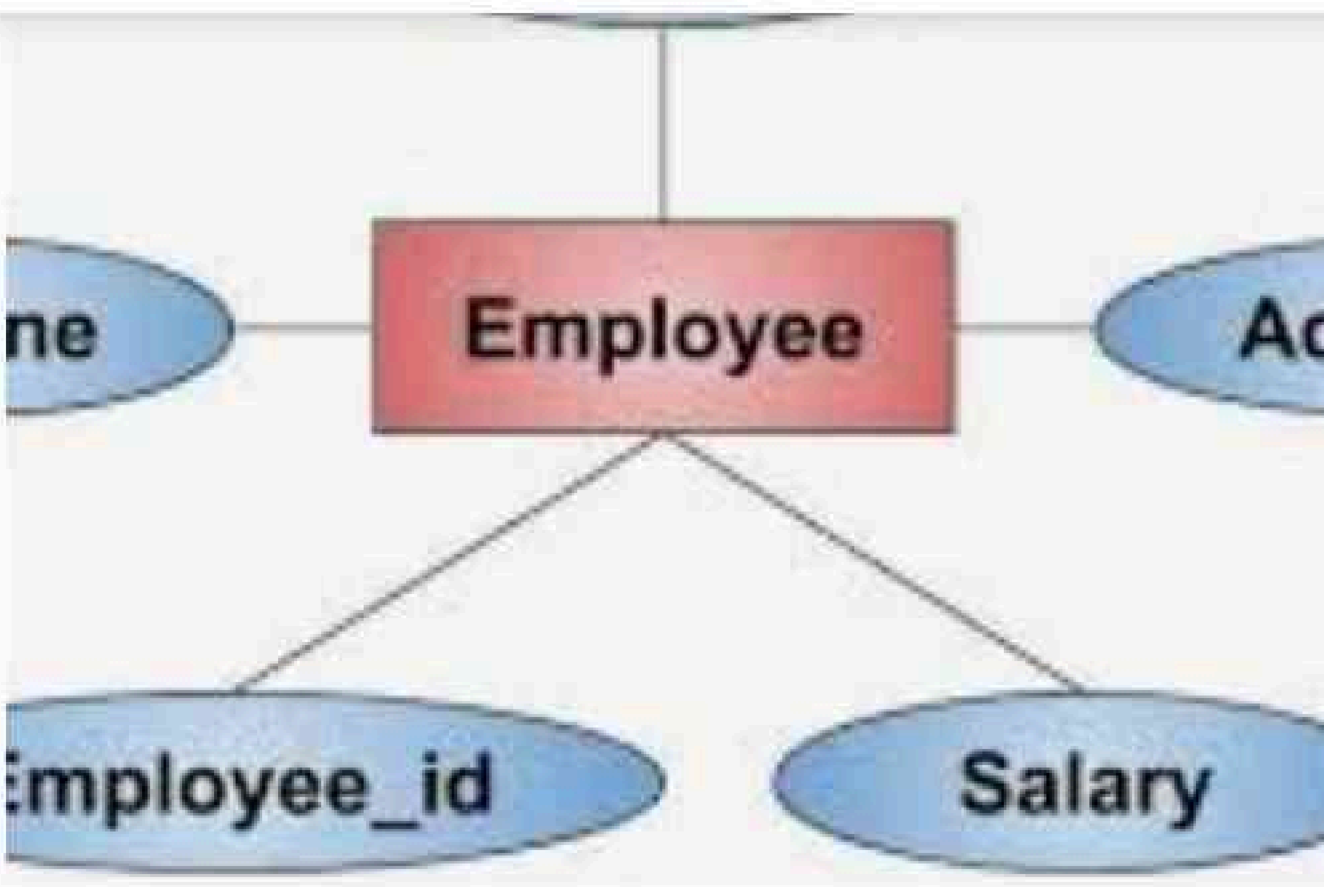
About featured snippets Feedback

Discover Search Saved

3:49

genaraailization with exam...





বাংলায়　In English

It is the reverse process of Specialization. For example, Whales, Sharks, and Dolphins can be generalized as Fish. Similarly, Bicycle, Bike, and Car can be generalized as Vehicles. Example: Suppose we have two entity types, Employee and Customer. Jan 19, 2020

https://afteracademy.com › blog

What is Generalization in DBMS? - AfterAcademy

About featured snippets　Feedback

Javatpoint
https://www.javatpoint.com › dbms-...

Generalization in DBMS

Discover　Search　Saved

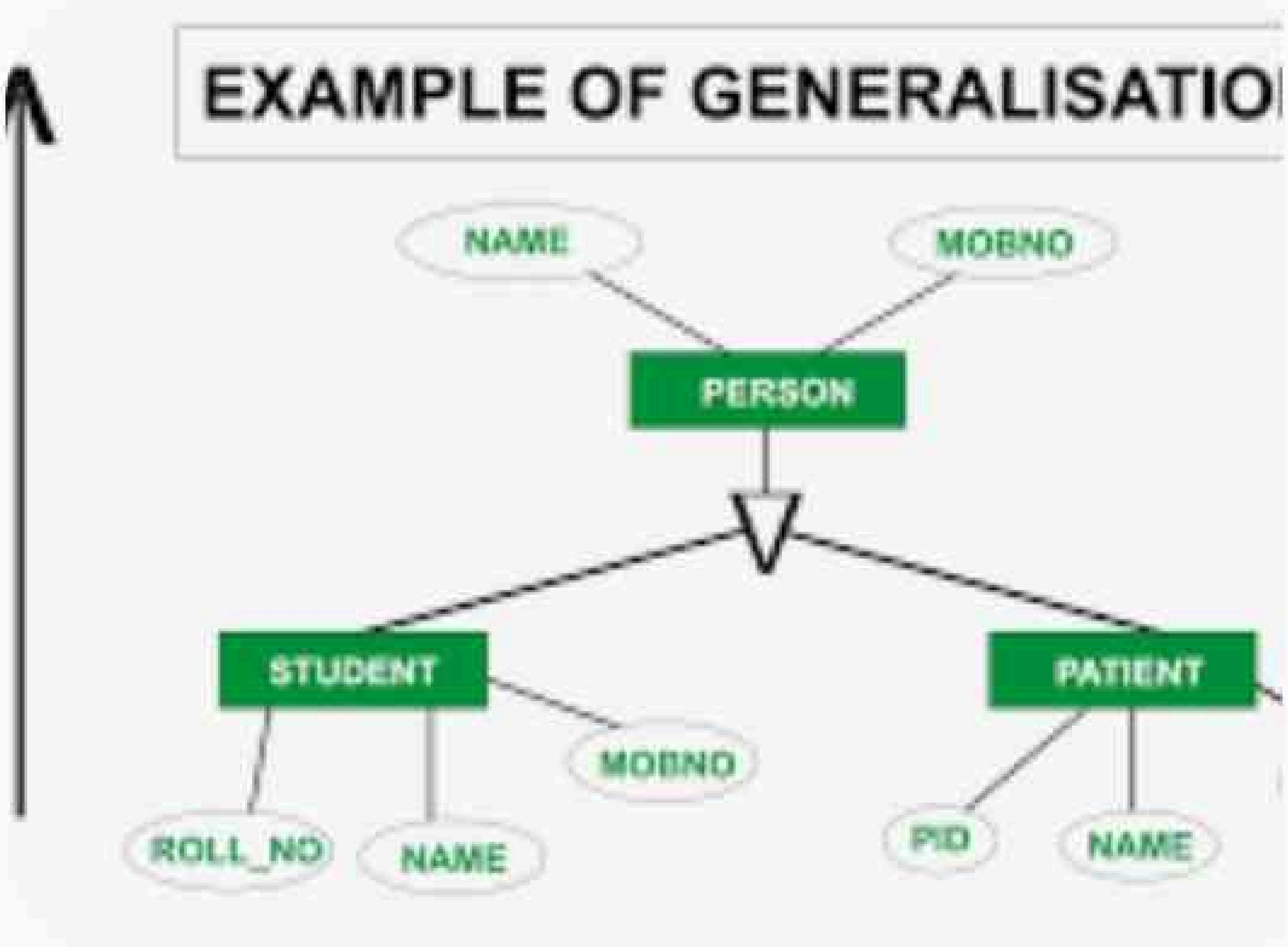
3:50 94

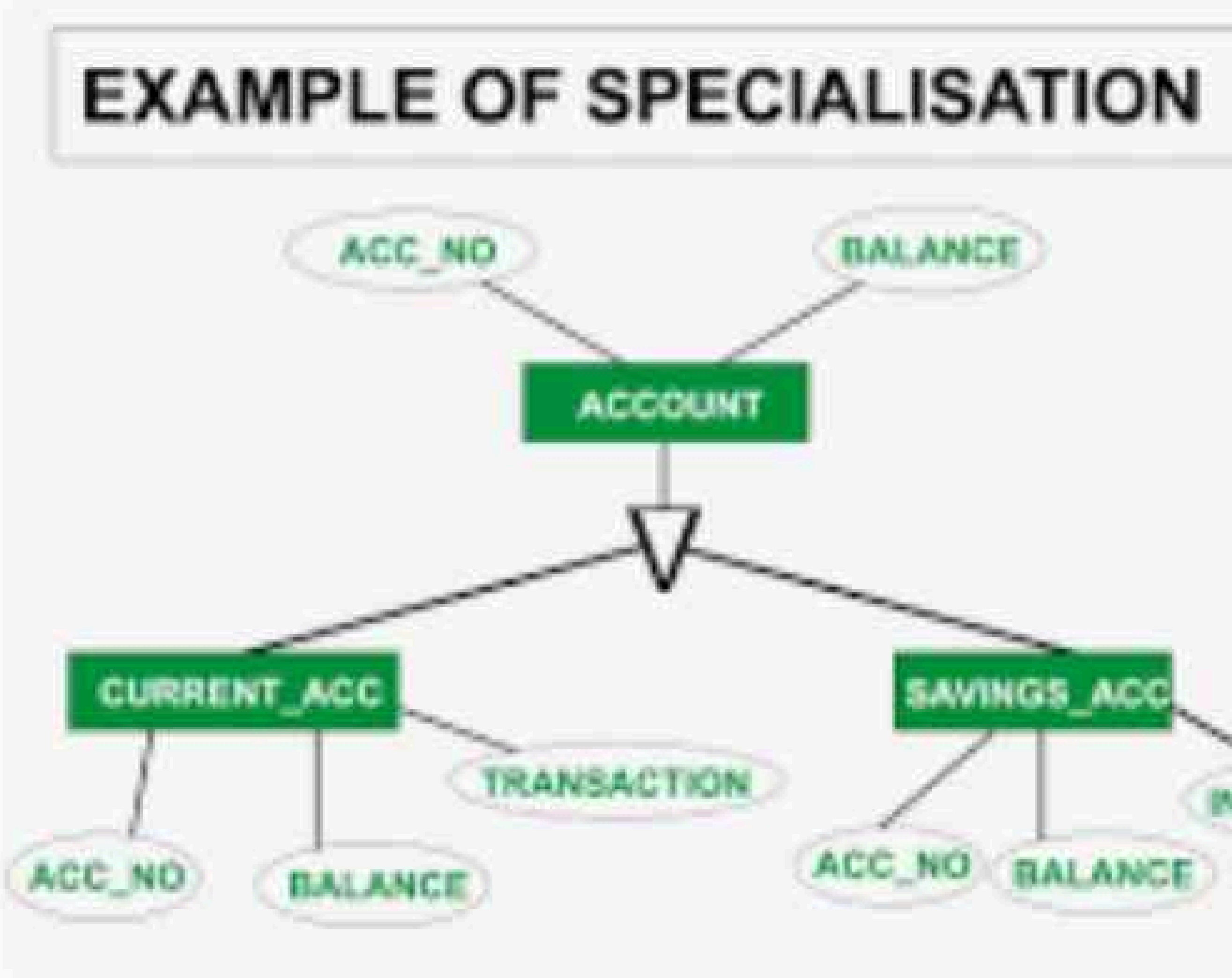
specialization vs generaliz...

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন

বিশেষীকরণ বনাম সাধারণীকরণ





বাংলায় In English

Definition. Specialization and generalization are main principles of database modeling. Specialization is based on a refinement of types or classes to more specific ones. Generalization maps or groups types or classes to more abstract or combined ones.

SN https://link.springer.com › ...

Specialization and Generalization - SpringerLink

Discover Search Saved

8th batch:

Ct:

Information and Communication Engineering
Class Test-1
Marks: 25 Time: 35mins

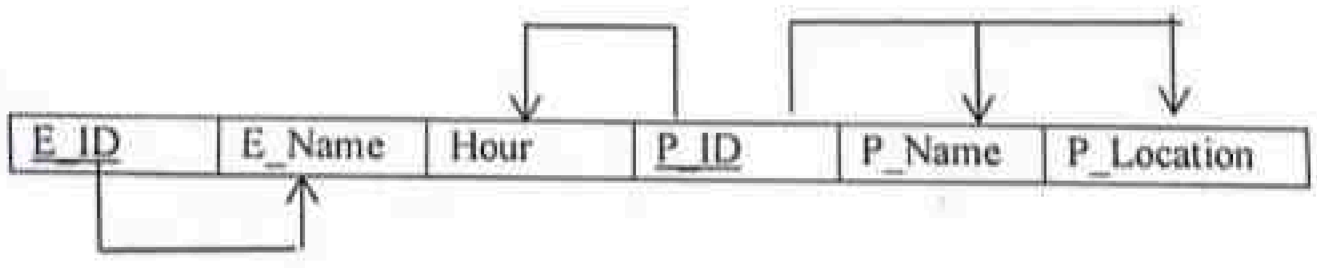
1. Define Database. Briefly explain what are the drawbacks of using a file system over database system? 1+6
2. Suppose you want to design a database for your department. Now write down and elaborate on the process to design the general structure of your database. 8
3. How a query is processed? Explain with an appropriate diagram. 4
4. Differentiate between the Primary key and the Candidate key. 2
5. Suppose a relation (r) is given below- 4

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

Now find out
(i) $\sigma_{A=B \wedge C \leq 5}(r)$
(ii) $\Pi_{A,B}(r)$

Information and Communication Engineering
Database Management System (Class Test-2)
Marks: 25 Time: 35mins

1. Define E-R model. Briefly describe the basic concepts of ER data model with example. 1+3
2. What is mapping cardinality? Explain different types of mapping cardinalities with diagram. 1+4
3. The main entities of the NSTU Library Management System for the (E-R) entity-relationship model are Student (name, s_id, address, mobile_no, date_of_birth, age), Book (book_id, book_name, author, book_price) each entity must have one primary key, the attribute name is complex, the attribute address is composite, the attribute mobile_no is multi-valued, the attribute date_of_birth is stored and the attribute age is derived etc. From the above description construct the E-R diagram. 6
4. How can you convert a non-binary relationship to binary relationship? Elaborate with an example. 4
5. Make use of the normalization process for the following relation up to BCNF 6



Noakhali Science and Technology University
Department of Information and Communication Engineering
Term Final Examination -2022

Course Code: ICE-2207
Session: 2019-2020

Course Title: Database Management System
Semester: 2nd Year 2nd Semester

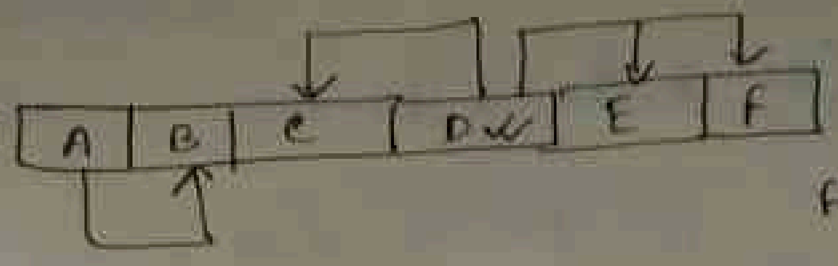
Total Marks: 70
Time: 04 Hours

Answer any seven of the following questions.

1. a) What is database management system? Discuss the significant application of database management systems. 1+2
   b) Explain the advantages of database system over file processing system. 3
   c) What is the concept of physical data independence and its importance in database systems? 2
   d) Discuss the level of database abstraction. 2

2. a) Explain the following with example: 4
   i) Primary key

⑤



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

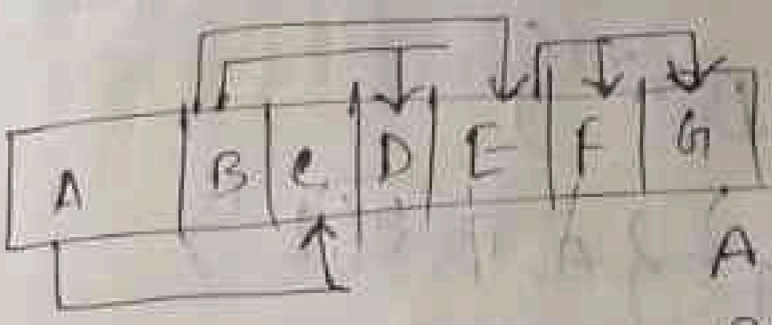
$CK = \{AD\} = \{A, D\}$

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

- → F_1 (AB) :- $A \to B$
- → F_2 (DCEF) :-
  - → DC : $D \to C$
  - → DEF
    - → DE ($D \to E$)
    - → DF ($D \to F$)
- → F_3 (AD)
  ($A \to D$)

2019–2020

2(C)



$A \to C$

$B \to D$

$B \to E$

$E \to F$

$E \to G$

$CK = \{AB\}$

$AB^+ = \{ABCDE\ EFG\}$

- → R_1 (AC)
- → R_2 (BDEFG)
  - → BD
  - → BEFG
    - → BE
    - → EF
    - → EG
- → R_3 (AB)

2.(c)

D-id → D-Name
S-id → S-Name
S-id → C-id
C-id → C-Name
C-id → Credit

R(ABCDEFG)

A → B
C → D
C → E
E → F
E → G

A → B
C → DE
E → FG

$A^+ = AB$
B^+
C^+
D^+
E^+
F^+
G^+

$AB^+ = AB$

$AC^+ = ACB$
$= ACBD$
$= ACBDE$
$= ACBDEF$
$= ACBDEFG$

$AD^+ = AD$
$= ADB$

$BC^+ = BC$
$= BCDEFG$

CK = {AC}
PA = {A, C}
NPA = {B, D, E, F, G}

2NF:

$A \xrightarrow[PD]{} B$, $C \xrightarrow[PD]{} DE$

$R_1(AB)$, $R_2(ACDE)$, $R_3(EFG)$

D-id	D-Name

D-id	S-id	S-Name	C-id

C-id	C-Name	Credit

3NF: Already 3NF

BCNF:

$R_1(AB)$, $R(ACDE)$, $R_3(EFG)$

A → B
C → DE
C → D
C → E
A →

E → FG
E → F
E → G

$R_1(AB)$, $R_2(CD)$, $R_3(CE)$, $R_4(EF)$, $R_5(EG)$, $R_6(AC)$

D-id	D-Name

S-id	S-Name

S-id	C-id

C-id	C-Name

C-id	Credit

D-id	S-id

5.

E_id → E_Name

P_id → Hour

P_id → P_Name

P_id → P_Location

R(ABCDEF)

$A \to B$ $C \to D$ $C \to E$ $C \to F$

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

$A^+ = AB$

$AB^+ = AB$

$AC^+ = AC$
$= ACB$
$= ACBDEF$

$BC^+ = BC$

CK = {AC}

PA = {A, C}

NPA = {B, D, E, F}

2NF: $A \xrightarrow[PD]{} B$, $C \xrightarrow[PD]{} D$, $C \xrightarrow[PD]{} EF$

$R_1(AB)$, $R_2(ACD)$, $R_3(CEF)$

E_id	E_Name

E_id	P_id	Hour

P_id	P_Name	P_Location

3NF:

$R_1(AB)$: $A \to B$

$R_2(ACD)$: $A \to$, $C \to D$

$R_3(CEF)$: $C \to EF$, $C \to E$, $C \to F$

$R_1(AB)$, $R_2(CD)$, $R_3(CE)$, $R_4(CF)$, $R_5(AC)$

E_id	E_Name

P_id	Hour

P_id	P_Name

P_id	P_Location

E_id	P_id

BCNF: Already BCNF.

14:53

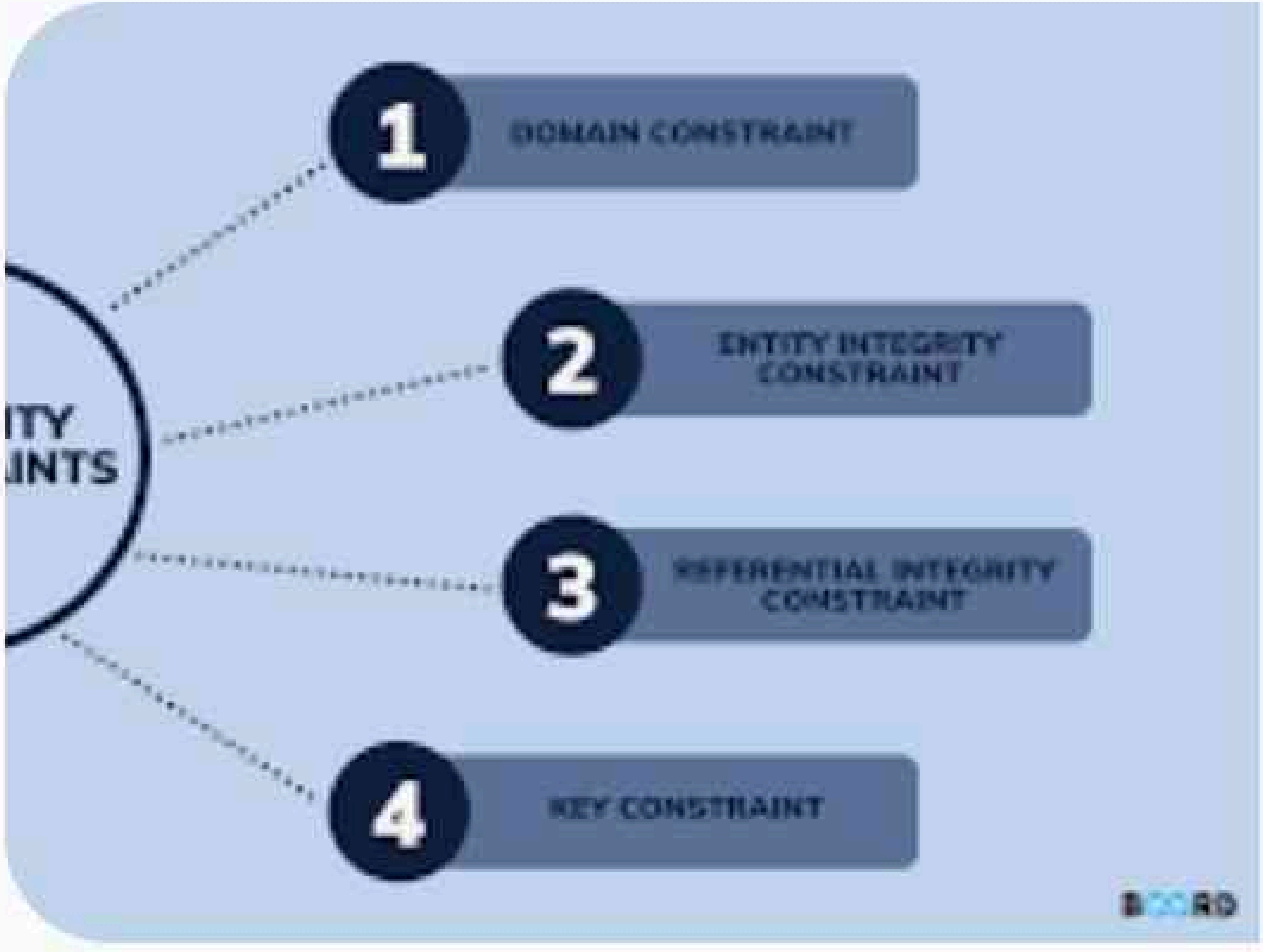
Google

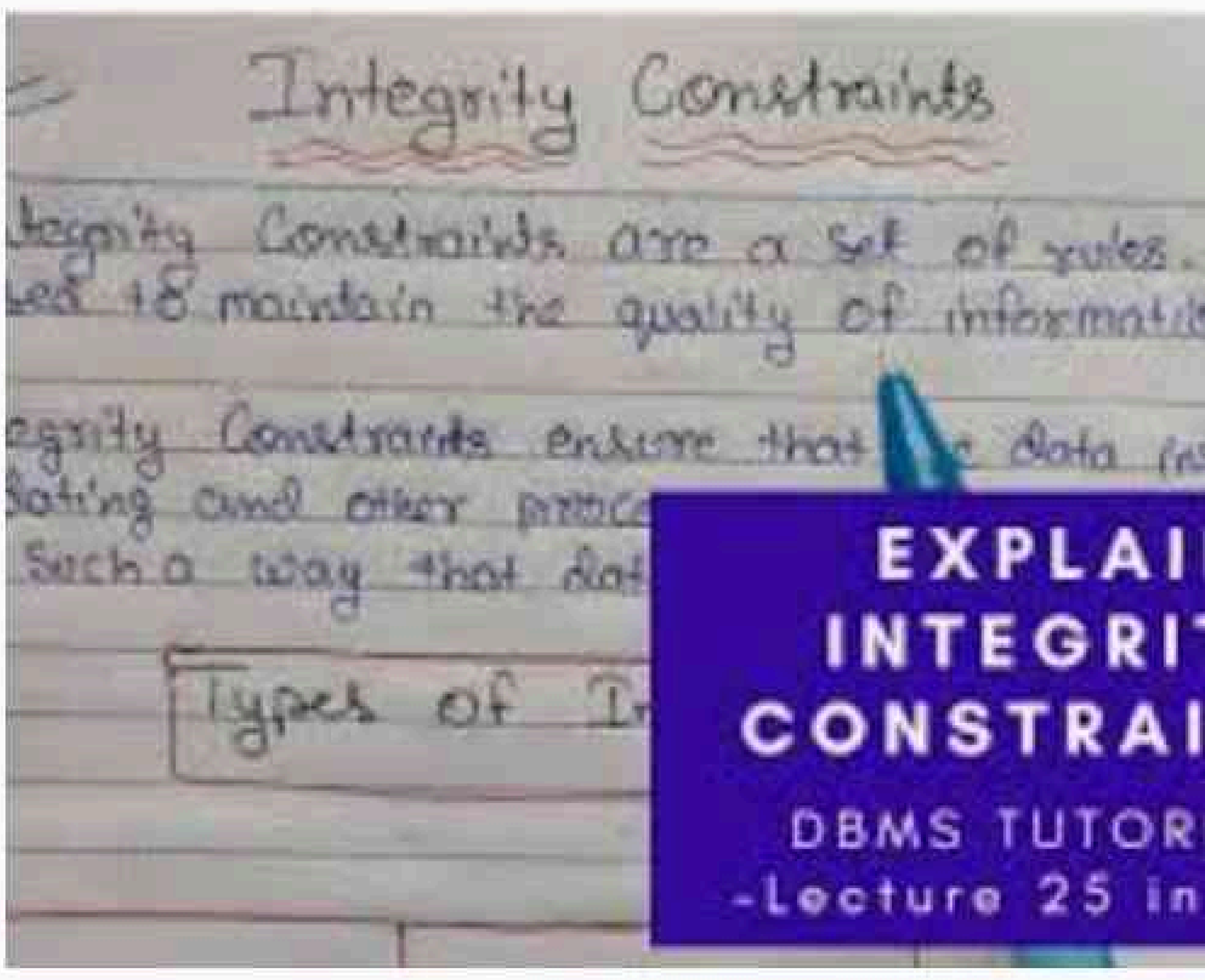
integrity constraints in db...

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন

ডিবিএমএসে অখণ্ডতার সীমাবদ্ধ





বাংলায় In English

Integrity constraints in DBMS are used to ensure that data is consistent and accurate. There are four main types of integrity constraints: domain, entity, referential, and key. Here, we'll take a closer look & explain the types of integrity constraints along with some examples. Sep 5, 2023

https://www.knowledgehut.com › in...

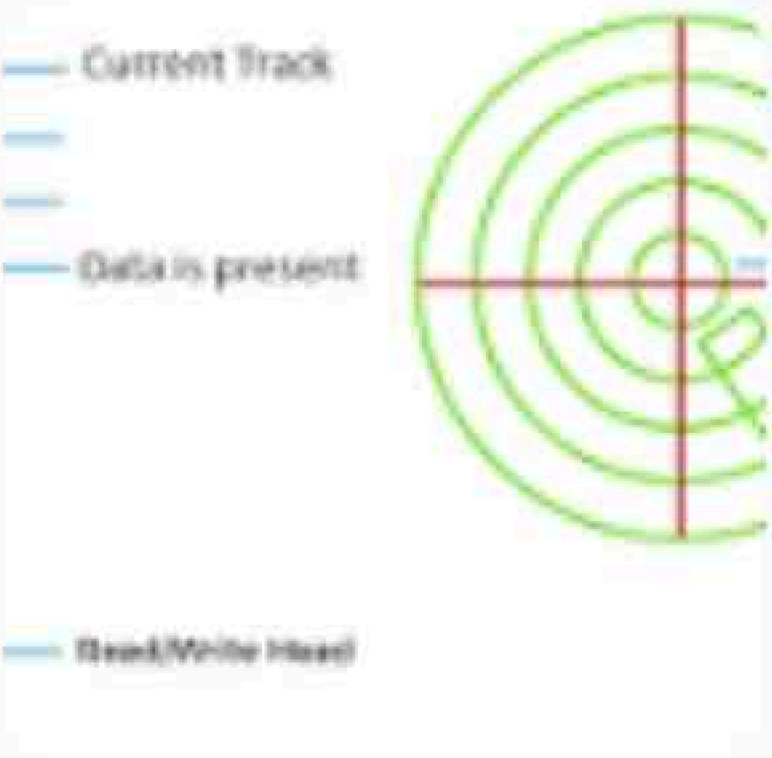
Discover Search Saved

https://dictionary.cambridge.org › a...

access time collocation | meaning and examples of use

MORE RESULTS

What is seek time and transfer time in access time?



Seek time is the time taken by the head to move from the current track to the one where data is present. Transfer time is the time taken to transfer data from the disk to the host system. Seek time can vary a lot depending upon the distance between the current and final position and how it has been instructed to go.

https://www.javatpoint.com › seek-t...

Difference between Seek Time and Transfer Time in Disk Scheduling

MORE RESULTS

What are the three access time delays in hard disk access?

Accessing the Disk

access time = seek time + rotational delay + transfer time +other delay

- Other Delays:
  - CPU time to issue I/O
  - Contention for controller
    - Different programs can be using the disk
  - Contention for bus, memory
    - Different programs can be transferring data
  - These delays are negligible compared to `Seek time + rotational delay + transfer time`
  - "Typical" Value: 0

Accessing the Disk

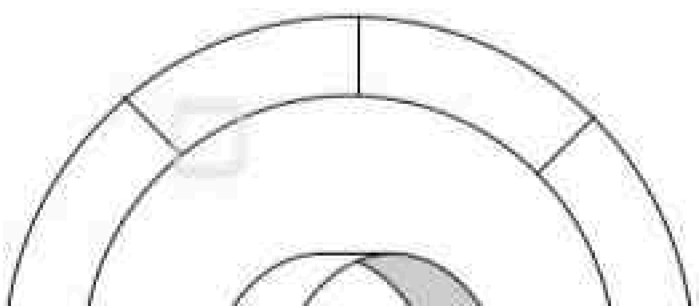
access time = seek time + rotational delay + transfer time

- Seek time: time to move the arm to position disk head on the right track (position the read/write head at the proper cylinder)
- Seek time can be 0 if the heads happen already to be at the proper cylinder.
- If not, the heads require some minimum time to start moving and to stop again, plus additional time that is roughly proportional to the distance traveled.
- The average seek time is often used as a way to characterize the speed of the disk.

Accessing the Disk

access time = seek time + rotational delay + transfe

- rotational delay: time to wait for sector to rotate under the
- i.e., wait for the beginning of the block

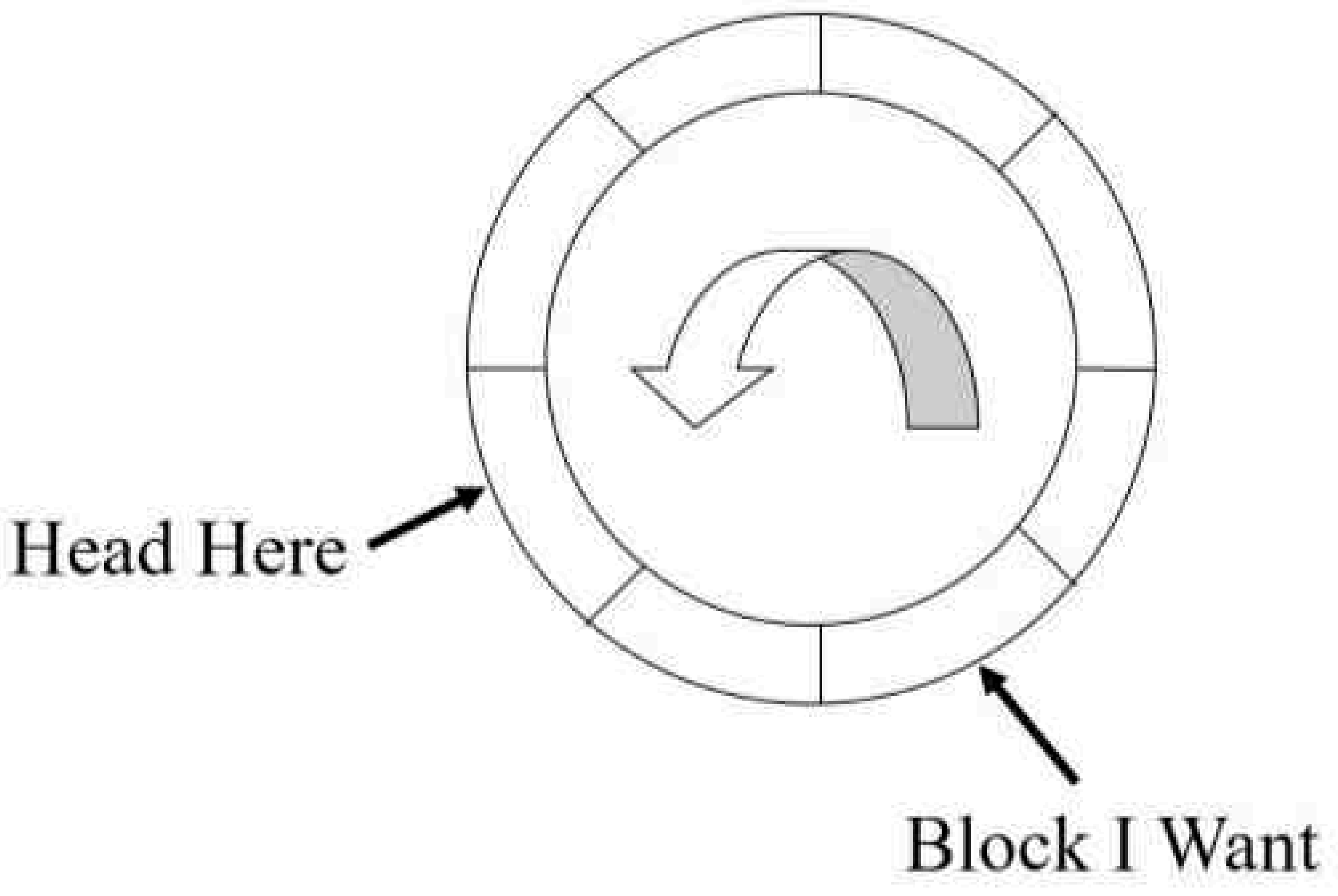


- The average seek time is often used as a way to characterize the speed of the disk.

Accessing the Disk

access time = seek time + rotational delay + transfer time

- rotational delay: time to wait for sector to rotate under the disk head
- i.e., wait for the beginning of the block



Average Rotational Delay

- On the average, the desired sector will be about half way around the circle when the heads arrive at its cylinder.
- Average rotational delay is time for $\frac{1}{2}$ revolution
- Example: Given a total revolution of 7200 RPM
  - One rotation $= \frac{60s}{7200} = 8.33$ ms
  - Average rotational latency $= 4.16$ ms

Accessing the Disk

Accessing the Disk

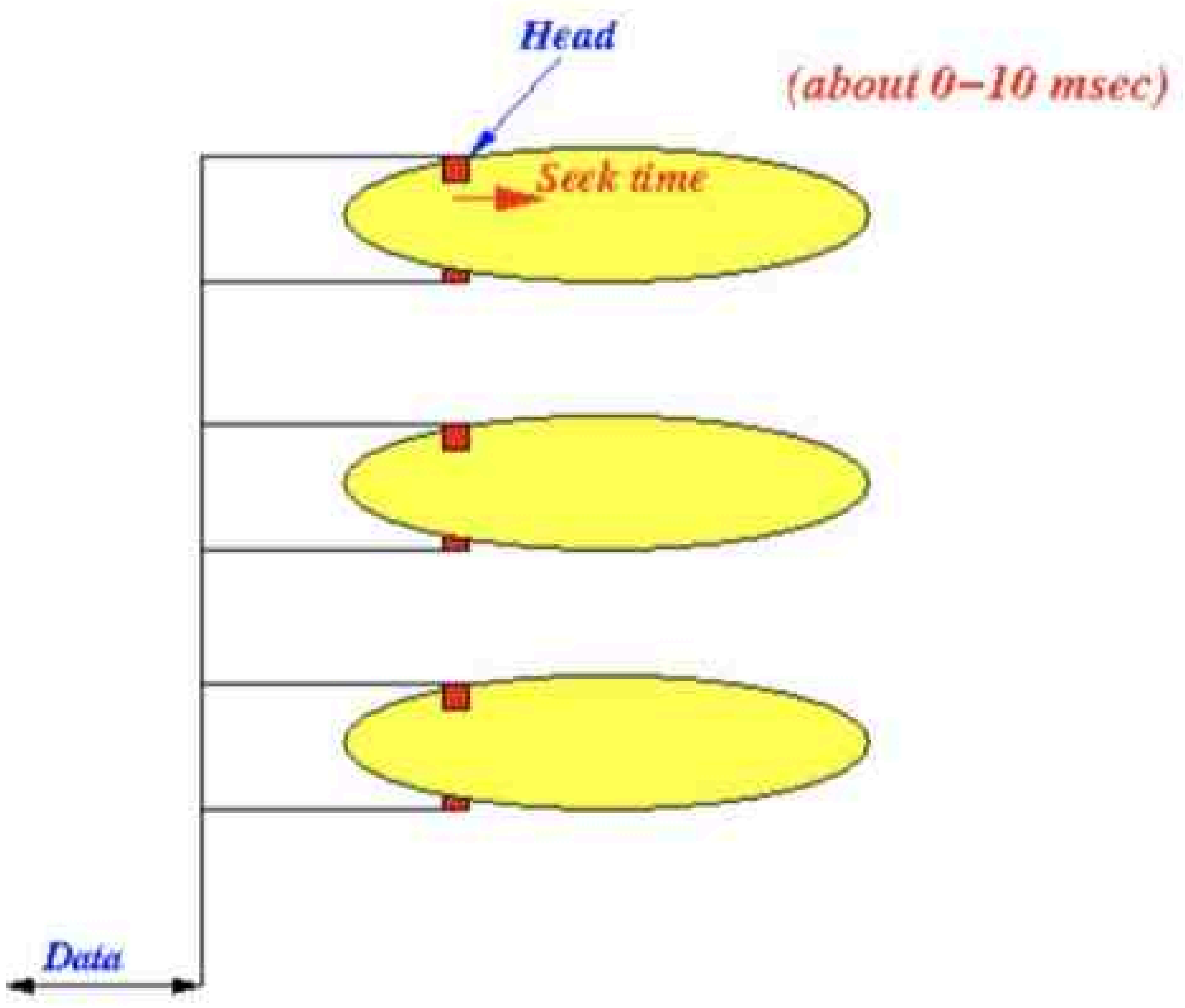
access time = seek time + rotational delay + transfer time

- data transfer time: time to move the data to/from the disk surface
- Transfer time is the time it takes the sectors of the block and any gaps between them to rotate past the head.
- Given a transfer rate, the transfer time=$\frac{Amount\ data\ transferred}{transfer\ rate}$
- Transfer Rate: # bits transferred/sec

Steps to access data on a disk

1. Move the disk heads to the desired cylinder

- Time to seek a cylinder = seek time



Steps to access data on a disk

2. Wait for the desired sector to arrive under the disk head

- Time to wait for a sector = rotational delay

14:59

Part-07: Data Transfer Rate-

Number of rotations in one second

= (3600 / 60) rotations/sec

= 60 rotations/sec

Now, Data transfer rate

= Number of heads x Capacity of one track x Number of rotations in one second

= 16 x (256 x 512 bytes) x 60

= $2^4 \times 2^8 \times 2^9 \times 60$ bytes/sec

= 60×2^{21} bytes/sec

= 120 MBps

Part-08: Average Access Time-

Time taken for one full rotation

= (60 / 3000) sec

= (1 / 50) sec

= 0.02 sec

= 20 msec

Part-08: Average Access Time-

Time taken for one full rotation

= (60 / 3000) sec

= (1 / 50) sec

= 0.02 sec

= 20 msec

Average rotational delay

= 1/2 x Time taken for one full rotation

= 1/2 x 20 msec

= 10 msec

Now, average access time

= Average seek time + Average rotational delay + Other factors

= 11.5 msec + 10 msec + 0

= 21.5 msec

Problem-02:

What is the average access time for transferring 512 bytes of data with the following specifications-

Problem-02:

What is the average access time for transferring 512 bytes of data with the following specifications-

- Average seek time = 5 msec
- Disk rotation = 6000 RPM
- Data rate = 40 KB/sec
- Controller overhead = 0.1 msec

Solution-

Given-

- Average seek time = 5 msec
- Disk rotation = 6000 RPM
- Data rate = 40 KB/sec
- Controller overhead = 0.1 msec

Time Taken For One Full Rotation-

Time taken for one full rotation

= (60 / 6000) sec

= (1 / 100) sec

= 0.01 sec

= 10 msec

= 10 msec

Average Rotational Delay-

Average rotational delay

= 1/2 x Time taken for one full rotation

= 1/2 x 10 msec

= 5 msec

Transfer Time-

Transfer time

= (512 bytes / 40 KB) sec

= 0.0125 sec

= 12.5 msec

Average Access Time-

Average access time

= Average seek time + Average rotational delay + Transfer time + Controller overhead + Queuing delay

= 5 msec + 5 msec + 12.5 msec + 0.1 msec + 0

= 22.6 msec

Problem-03:

A certain moving arm disk storage with one head has the following specifications-

- Number of tracks per surface = 200
- Disk rotation speed = 2400 RPM
- Track storage capacity = 62500 bits
- Average latency = P msec
- Data transfer rate = Q bits/sec

What is the value of P and Q?

Solution-

Given-

- Number of tracks per surface = 200
- Disk rotation speed = 2400 RPM
- Track storage capacity = 62500 bits

Time Taken For One Full Rotation-

Time taken for one full rotation

= (60 / 2400) sec

= (1 / 40) sec

= 0.025 sec

= 25 msec

= 0.025 sec

= 25 msec

Average Latency-

Average latency or Average rotational latency

= 1/2 x Time taken for one full rotation

= 1/2 x 25 msec

= 12.5 msec

Data Transfer Rate-

Data transfer rate

= Number of heads x Capacity of one track x Number of rotations in one second

= 1 x 62500 bits x (2400 / 60)

= 2500000 bits/sec

= 2.5 x 10^6 bits/sec

Thus, P = 12.5 and Q = 2.5 x 10^6

Problem-04:

(1) How query is processed? Explain with diagram.

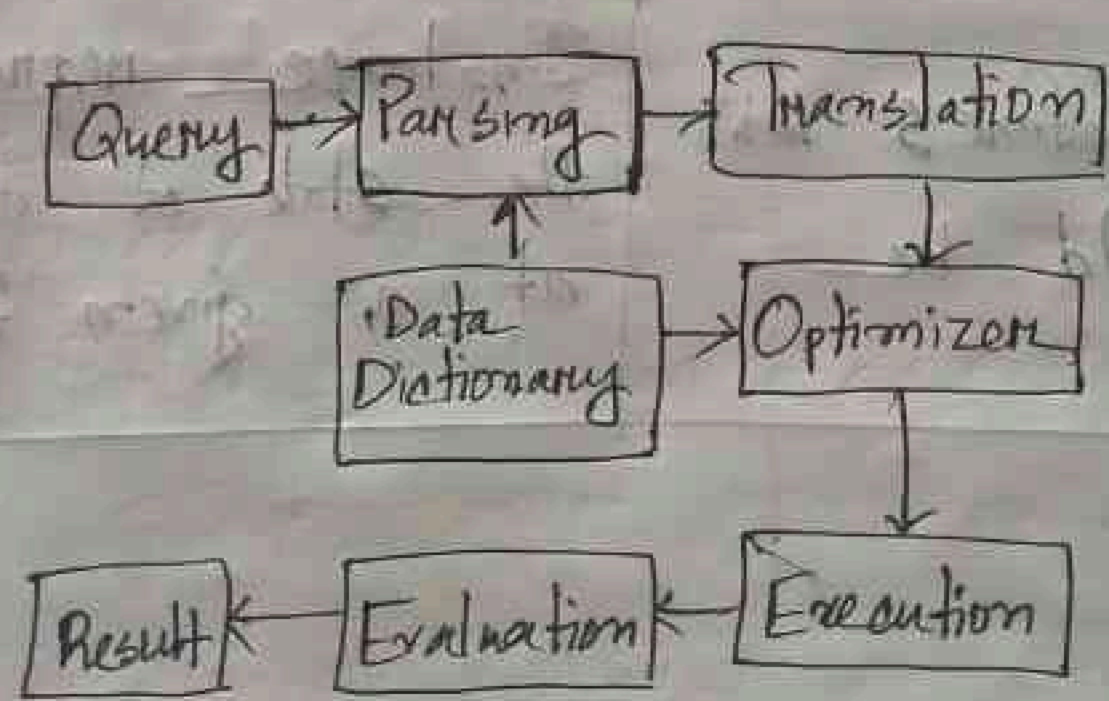
⟹ Query is processed in 3 steps—

1. Parsing
2. Optimization
3. Execution

Parsing: In this step, a query is checked for errors.

Optimization: Find the most efficient way to execute the query.

Excecution: After finding the most efficient way, the query starts executing.



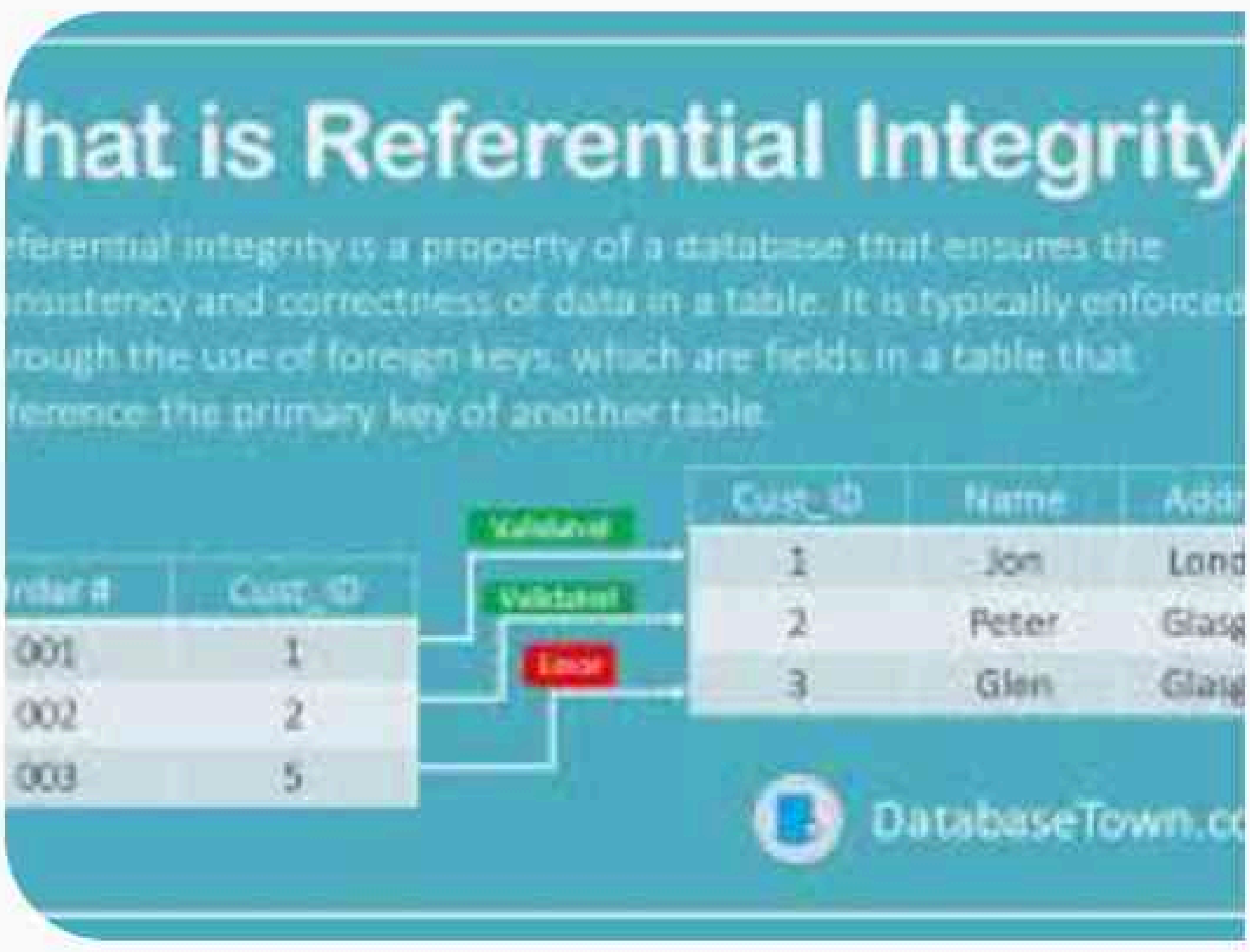
17:04

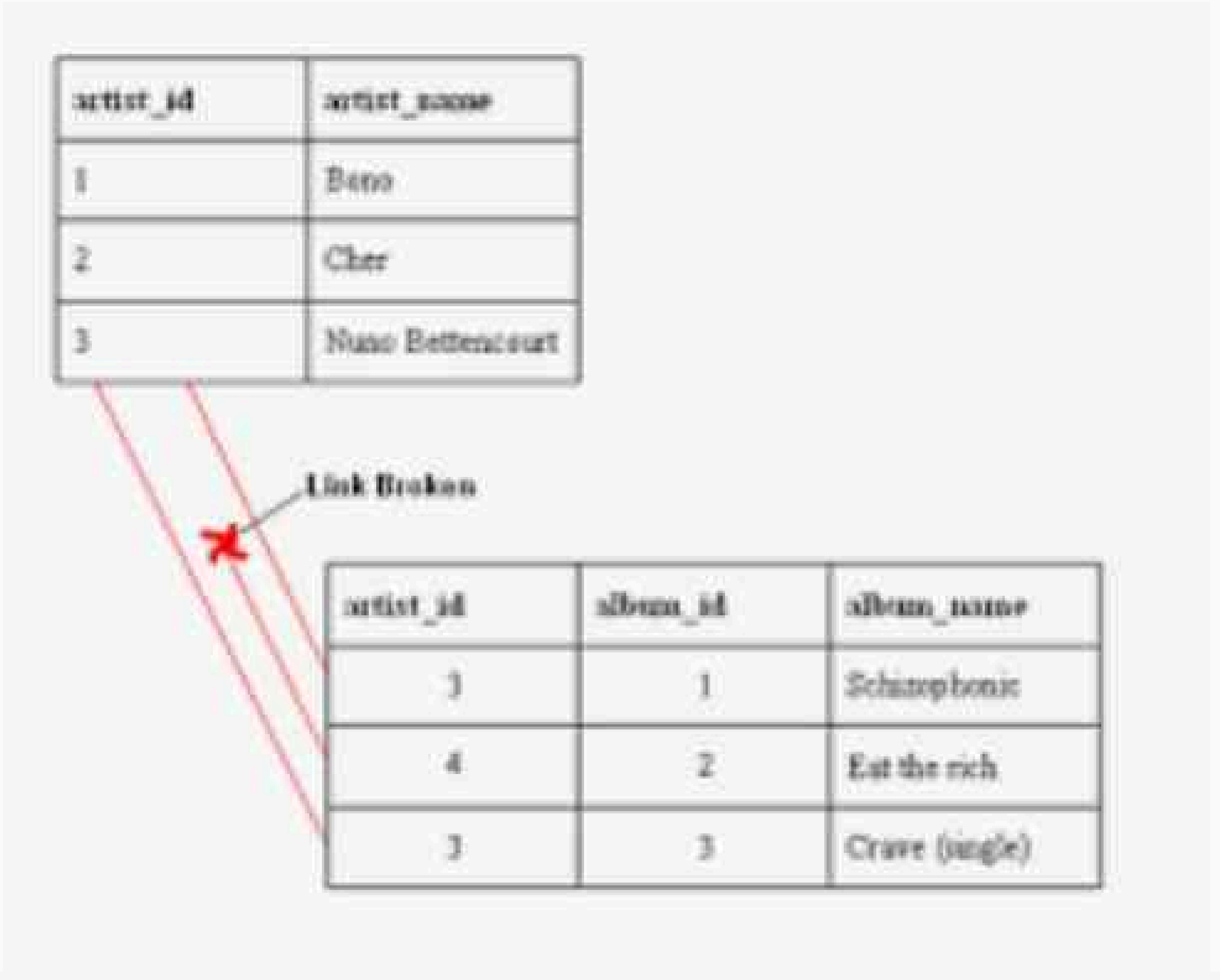
referential integrity in dbms

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন

ডিবিএমএসে রেফারেন্সিয়াল অ...





বাংলায় In English

Referential integrity **refers to the relationship between tables**. Because each table in a database must have a primary key, this primary key can appear in other tables because of its relationship to data within those tables. When a primary key from one table appears in another table, it is called a foreign key .

Discover Search Saved

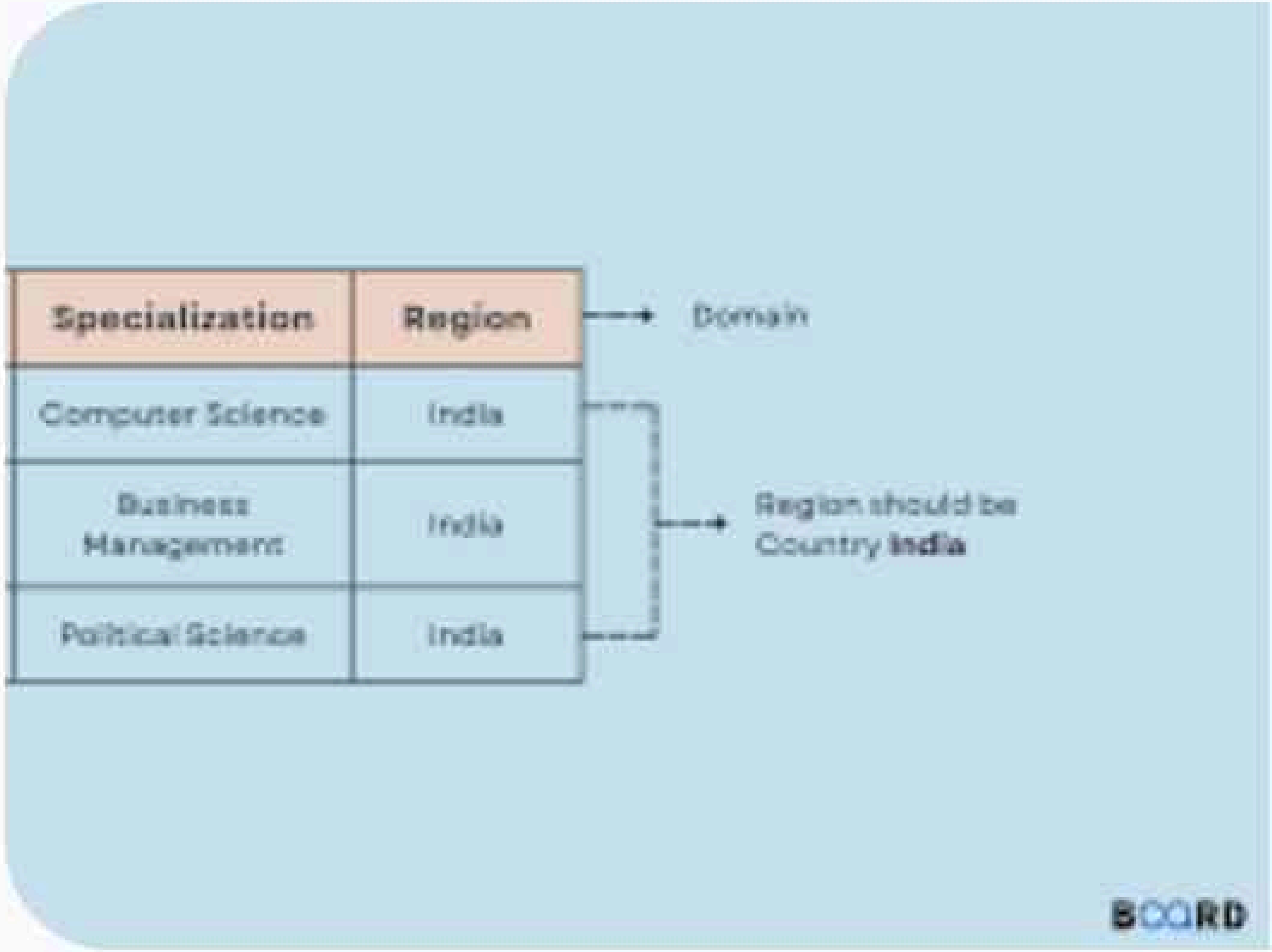
Google

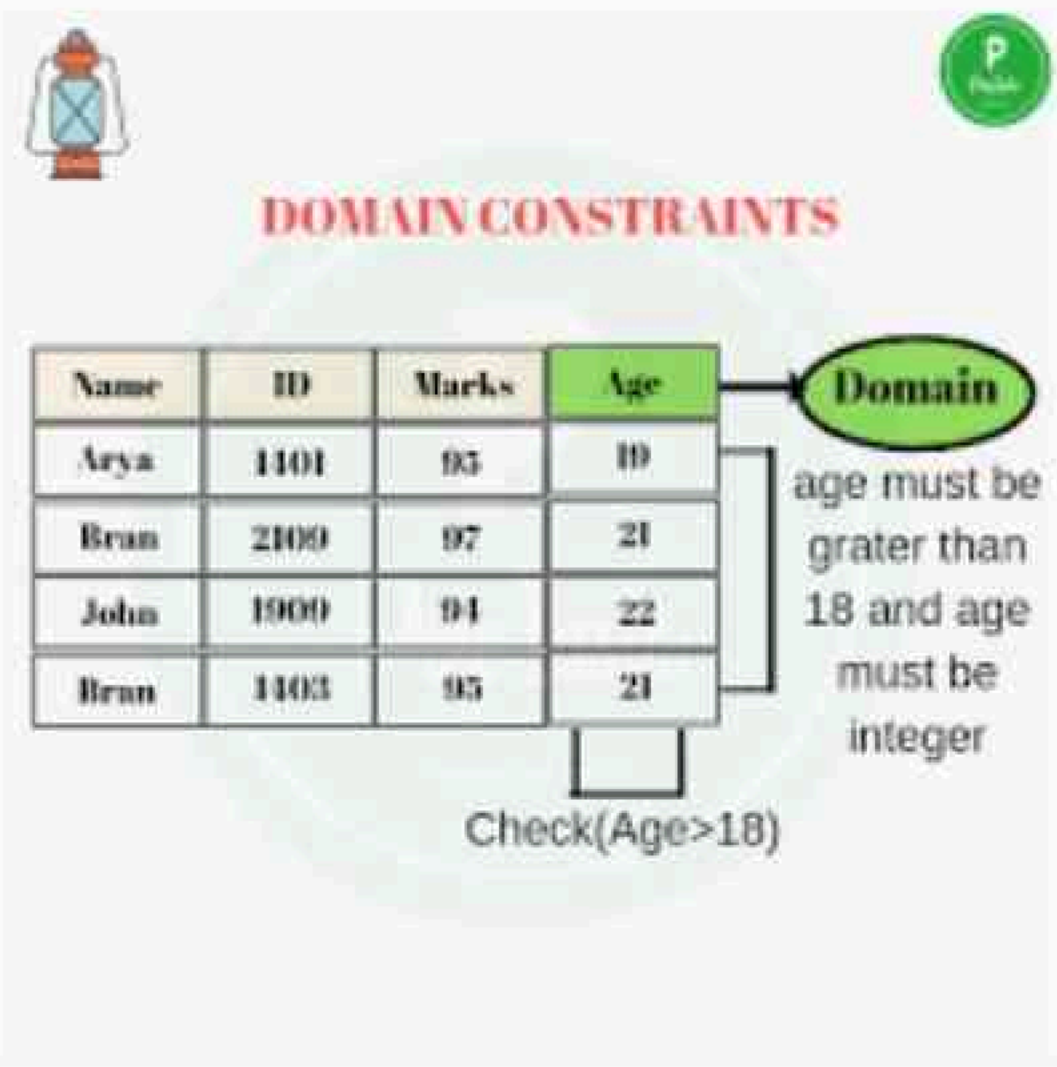
domain constraints in dbms

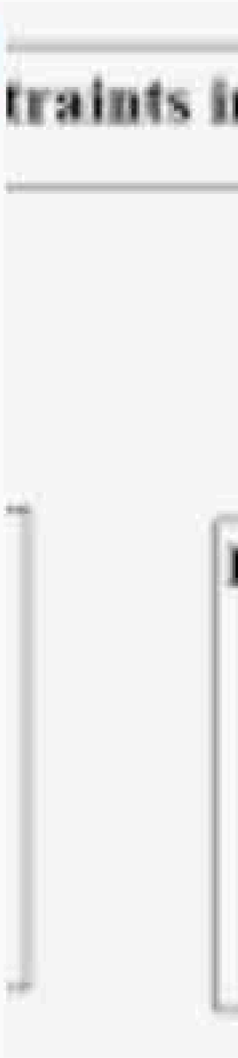
All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন

dbms এ ডোমেনের সীমাবদ্ধতা







বাংলায় In English

Domain constraints in DBMS are **the set of rules which defines what kind of attributes can be stored in an entity** (a table that stores data). Domain Constraints help us to enter the data into the table according to the particular data type.

https://www.boardinfinity.com › blog

Domain Constraints in DBMS - Board

18:58 93

Google

how query is procced? in d...

All Videos Images News Shopping Books

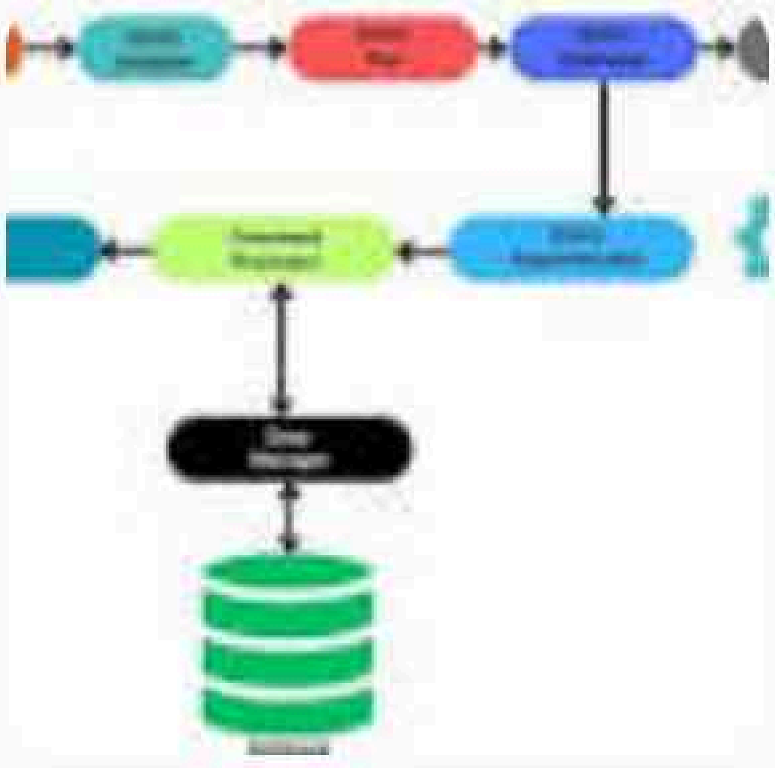
Showing results for how query is ***processed***? in dbms

Search instead for how query is procced? in dbms

বাঙলায় অনুসন্ধান করুন

dbms-এ প্রশ্ন কীভাবে প্রক্রিয়া ক

বাংলায় In English



A DBMS processes queries in three phases that are parsing (checks syntax), optimization (finds efficient plan), and execution (retrieves and manipulates data). Oct 18, 2023

https://www.codingninjas.com › qu...

Query Processing in DBMS - Coding Ninjas

About featured snippets Feedback

Discover Search Saved

Evaluation

For this, with addition to the relational algebra translation, it is required to annotate the translated relational algebra expression with the instructions used for specifying and evaluating each operation. Thus, after translating the user query, the system executes a query evaluation plan.

Query Evaluation Plan

expression which is saved with the name X.

- **Example-1:** Query to rename the relation Student as Male Student and the attributes of Student – RollNo, SName as (Sno, Name).

Sno	Name
2600	Ronny
2655	Raja

ρ MaleStudent(Sno, Name) $\pi_{\text{RollNo, SName}}(\sigma_{\text{Condition}}(\text{Student}))$

- **Example-2:** Query to rename the attributes Name, Age of table Department to A,B.

ρ (A, B) (Department)

- **Example-3:** Query to rename the name Project to Pro and its attrib to P, Q, R.

- **Example-2:** Query to rename the attributes Name, Age of table Department to A,B.

```
ρ (A, B) (Department)
```

- **Example-3:** Query to rename the table name Project to Pro and its attributes to P, Q, R.

```
ρ Pro(P, Q, R) (Project)
```

- **Example-4:** Query to rename the first attribute of the table Student with attributes A, B, C to P.

```
ρ (P, B, C) (Student)
```

commit/abort. In this protocol transactions can be serialized in the order in which they commit.

Much deeper :

Strict 2PL :

Strict 2PL	
T1	T2
s-lock(A)	
read(A)	
	s-lock(A)
x-lock(B)	
unlock(A)	
read(B)	
write(B)	
	read(A)
	unlock(A)
commit	
unlock(B)	
	s-lock(B)
	read(B)
	unlock(B)
	commit

Same as 2PL but Hold all exclusive locks until the transaction has already successfully committed or aborted. –It guarantees cascadeless recoverability

Same as 2PL but Hold all exclusive locks until the transaction has already successfully committed or aborted. –It guarantees cascadeless recoverability

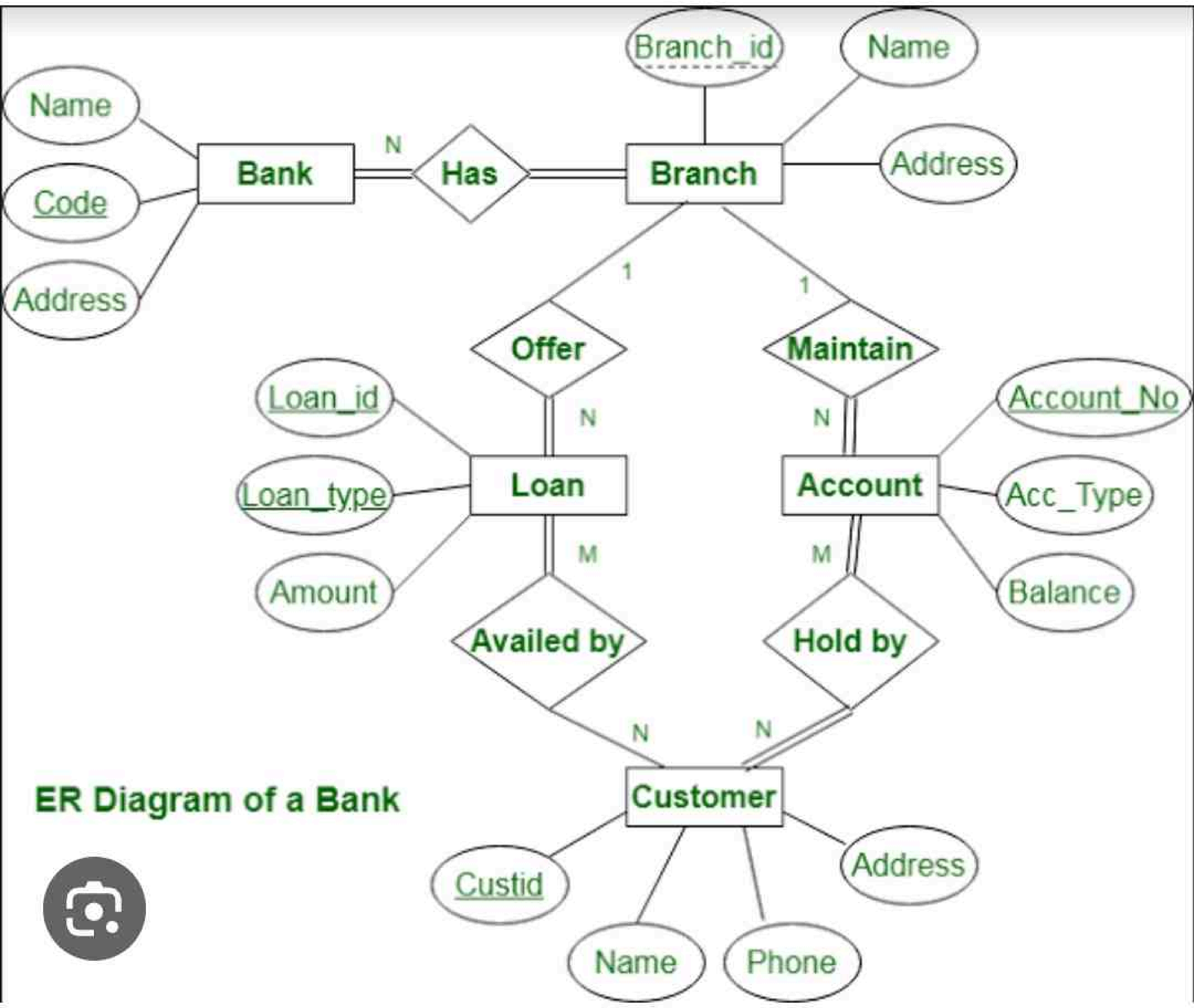
Rigorous 2PL :

Rigorous 2PL	
T1	T2
s-lock(A)	
read(A)	
	s-lock(A)
x-lock(B)	
	read(A)
read(B)	
write(B)	
commit	
unlock(B)	
	s-lock(B)
	read(B)
unlock(A)	
	commit
	unlock(A)
	unlock(B)

Same as Strict 2PL but Hold all locks until the transaction has already successfully committed or aborted. –It is used in dynamic environments where data access

21:49 ··· 84

GeeksforGeeks



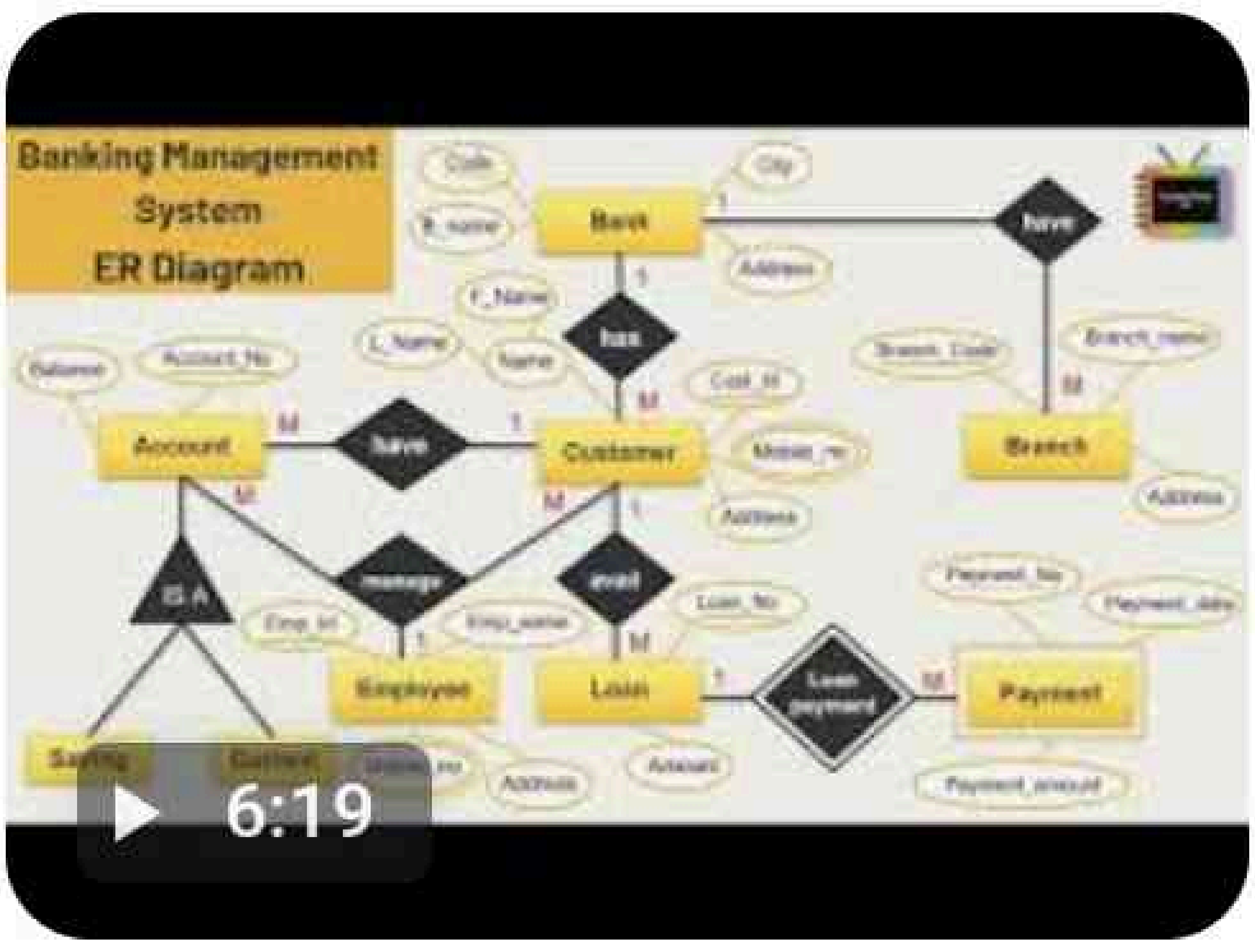
ER diagram of Bank Management System -...

Visit >

Images may be subject to copyright. **Learn more**

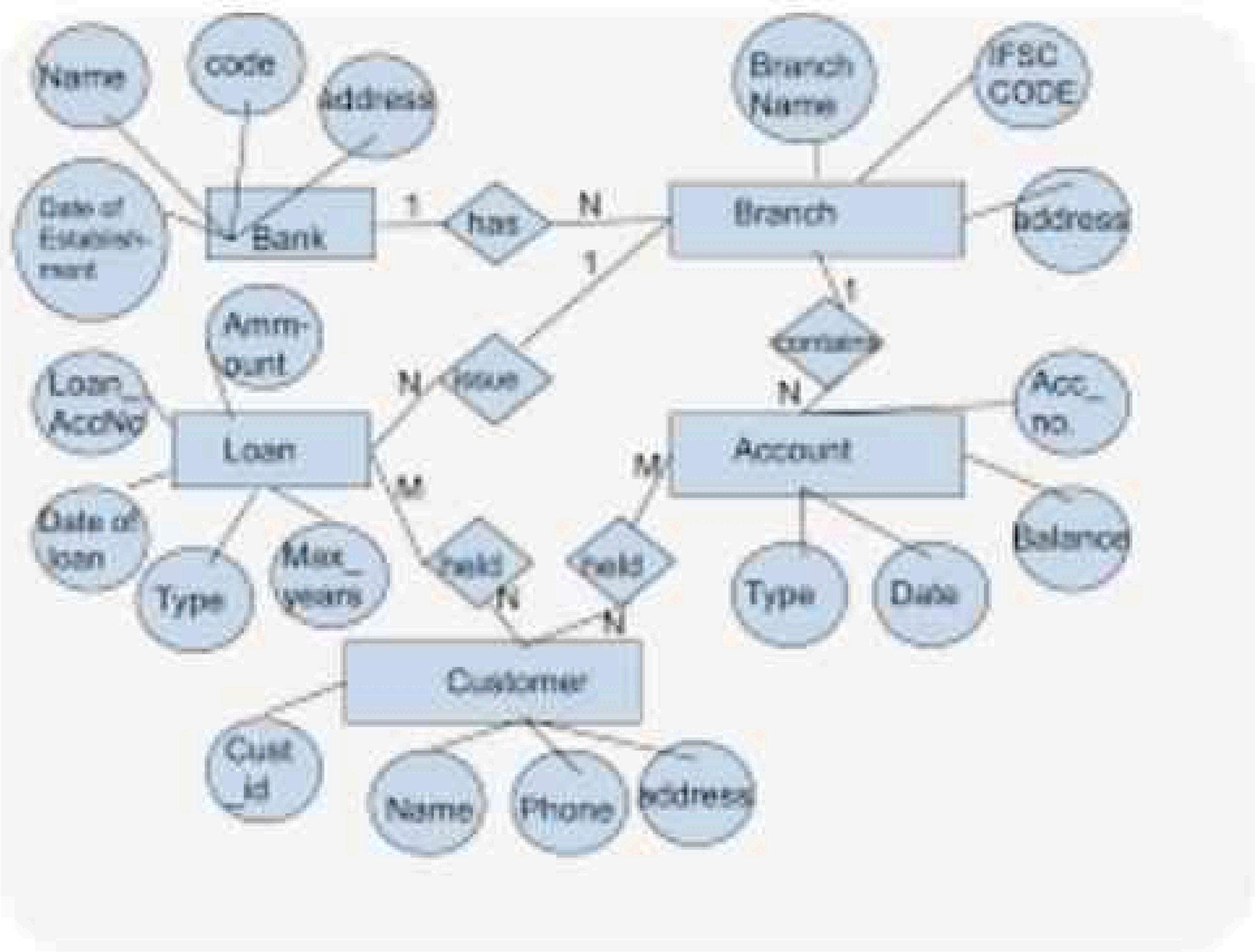
Share

Save



YouTube

Banking Management sy...



A Tutorial Website with Re...

ER diagram of the Bank ...

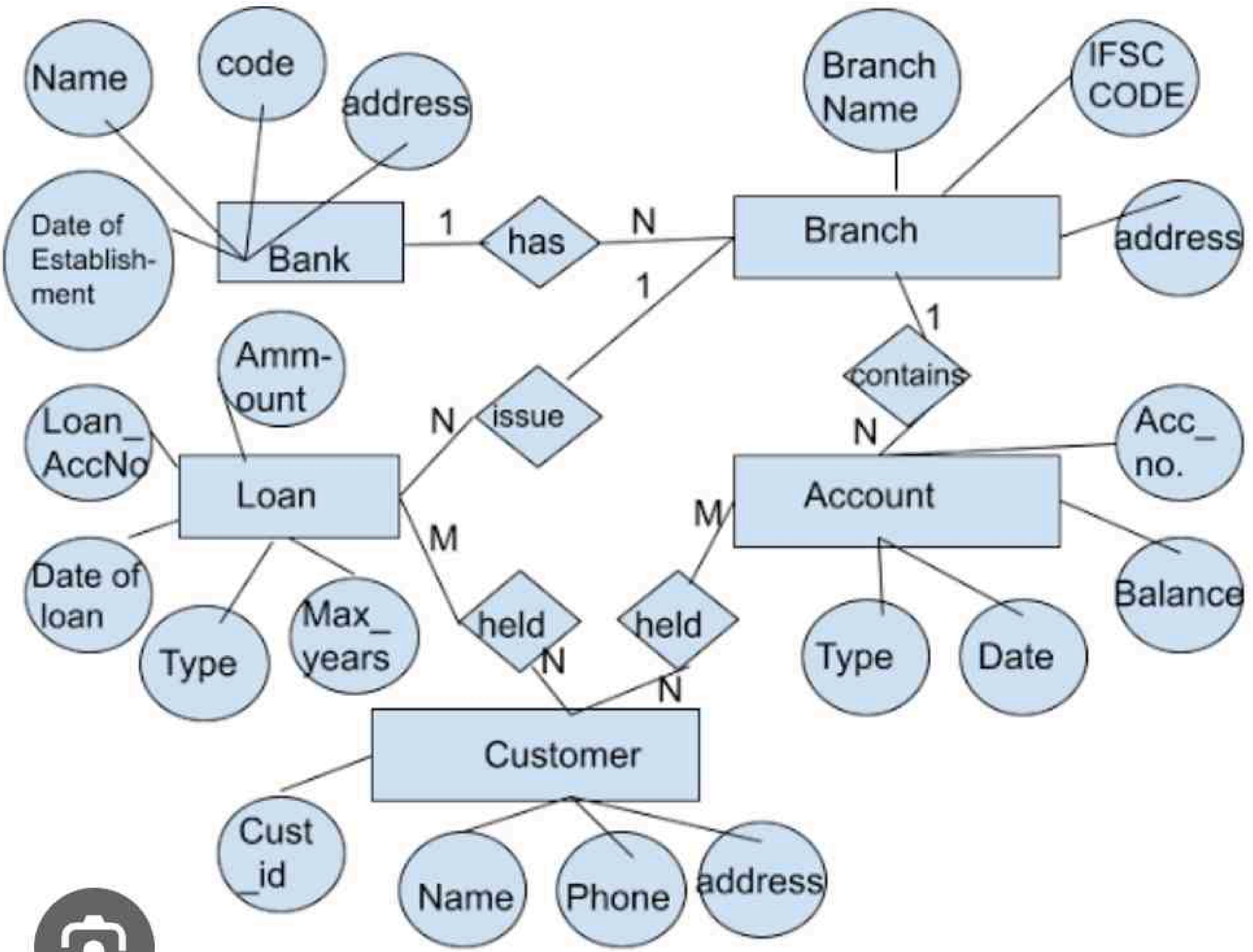
Discover

Search

Saved

21:49 ··· 84

A Tutorial Website with Real Ti...



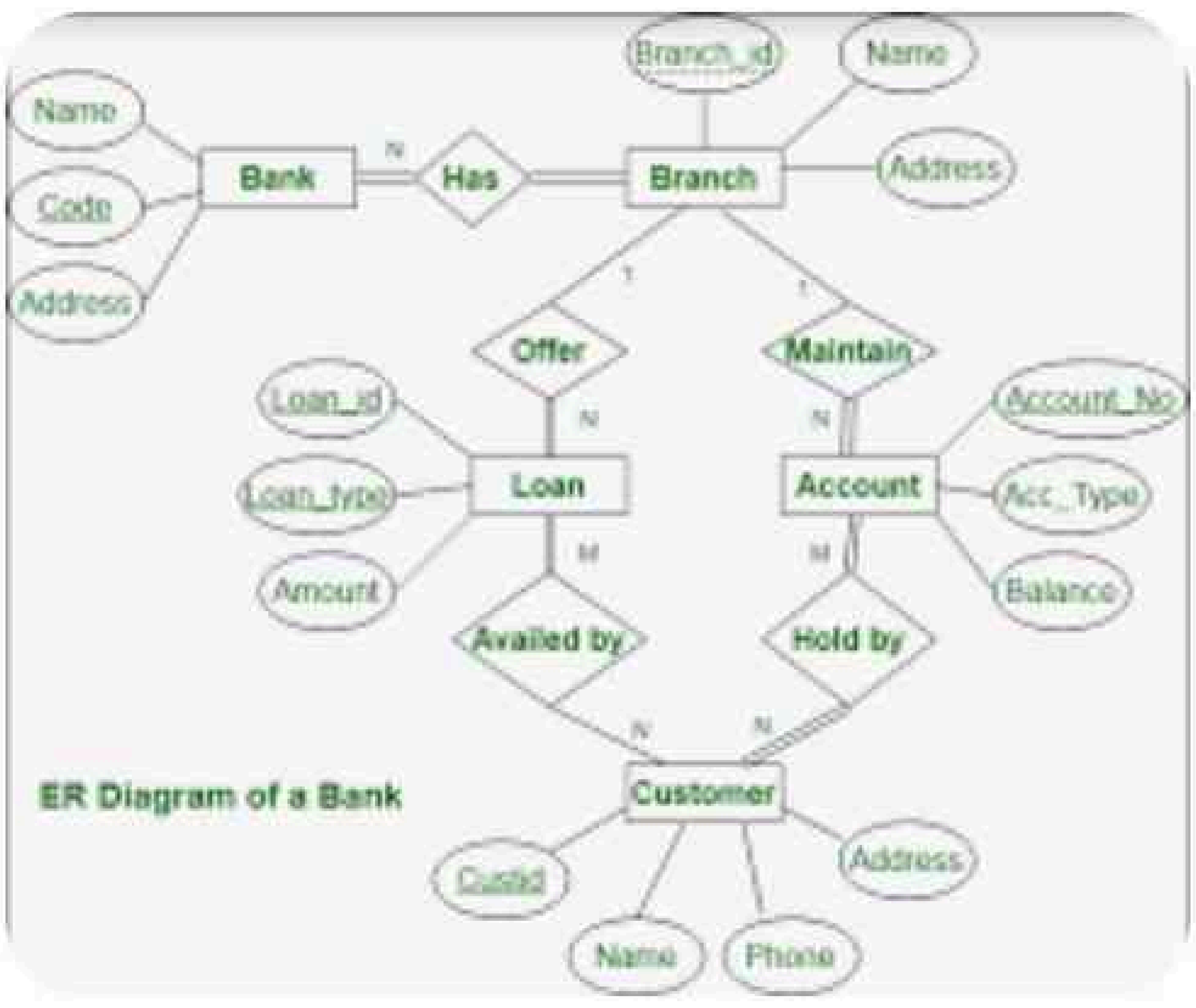
ER diagram of the Bank Management System in pytho...

Visit >

Images may be subject to copyright. **Learn more**

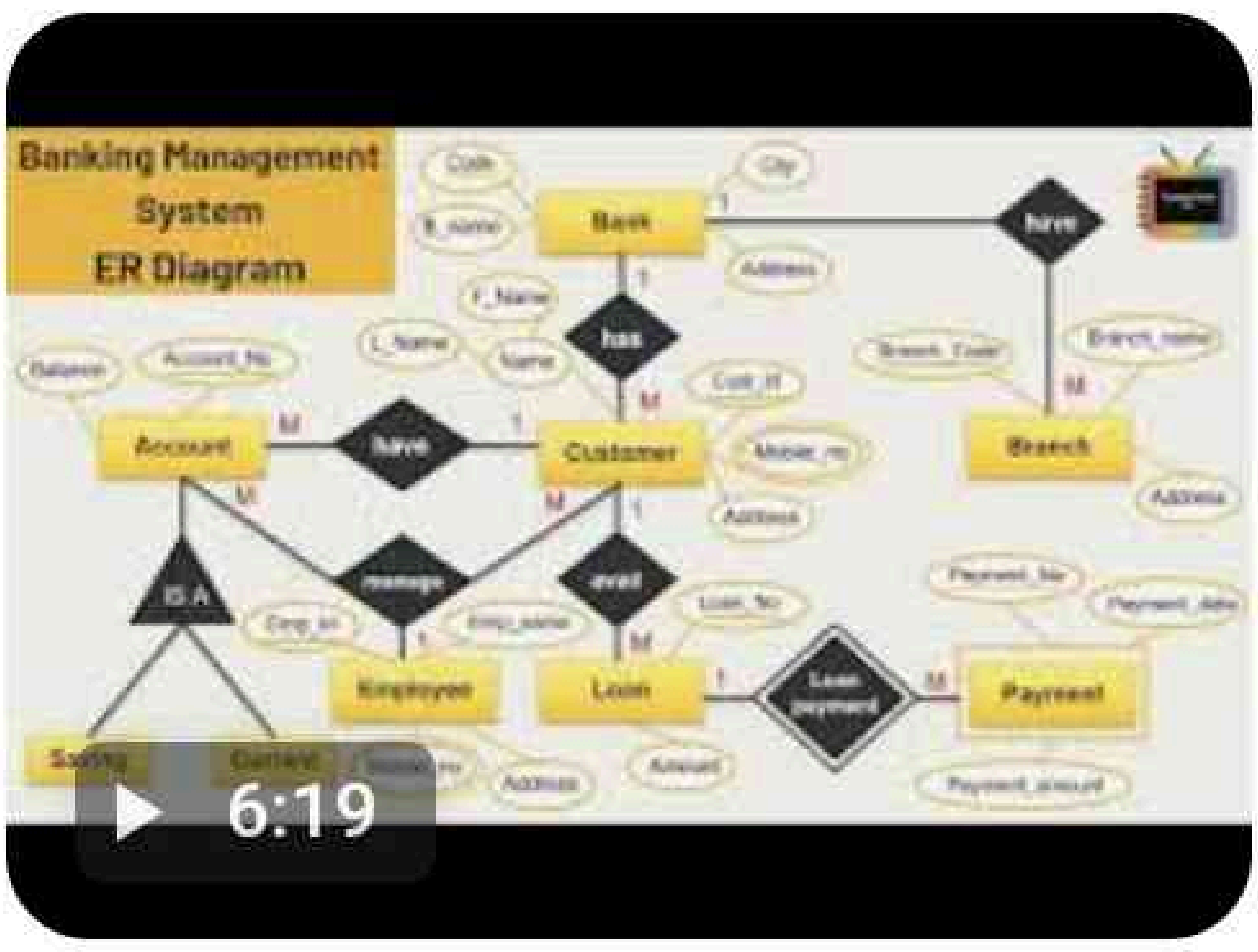
Share

Save



GeeksforGeeks

ER diagram of Bank Man...



YouTube

Banking Management sy...

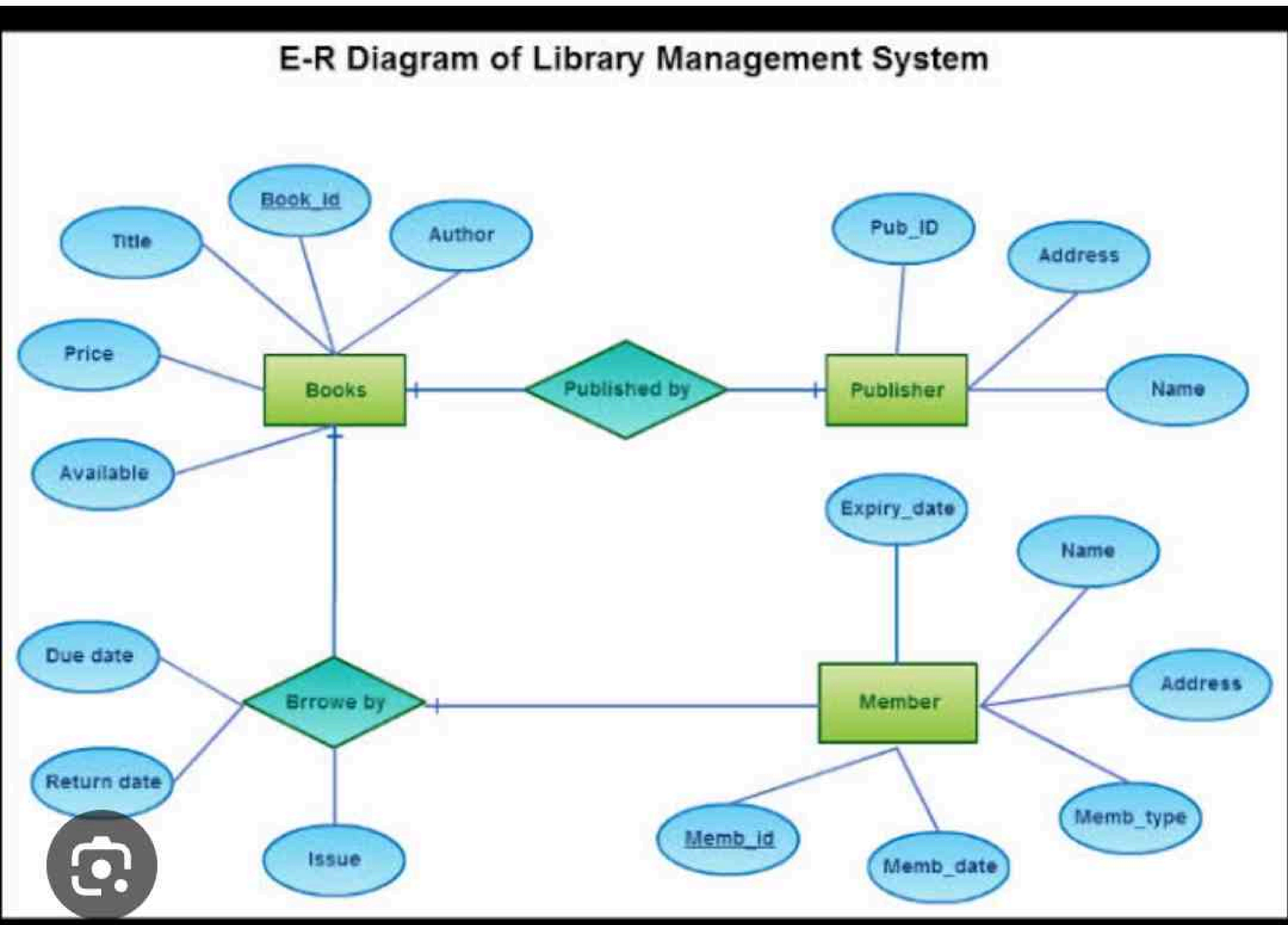
Discover

Search

Saved

21:50 84

Computers Hub!

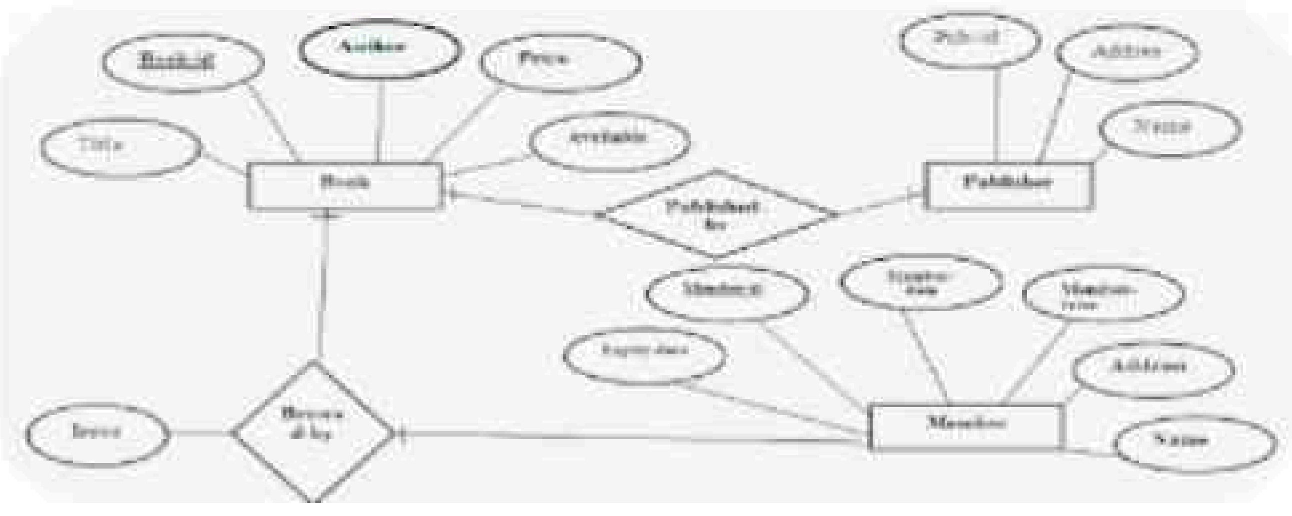


E-R Diagram of Library Management System -...

Visit >

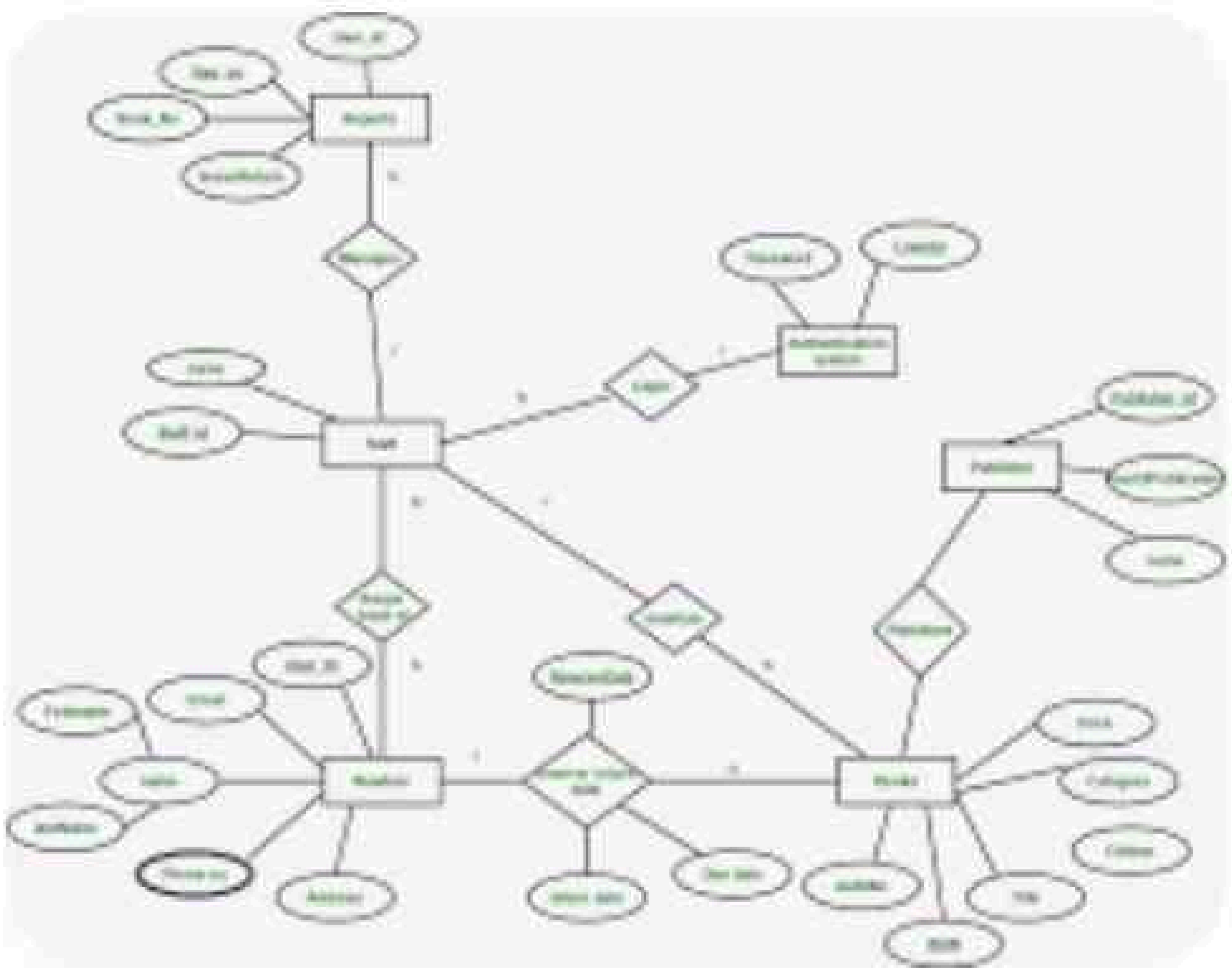
Share

Save



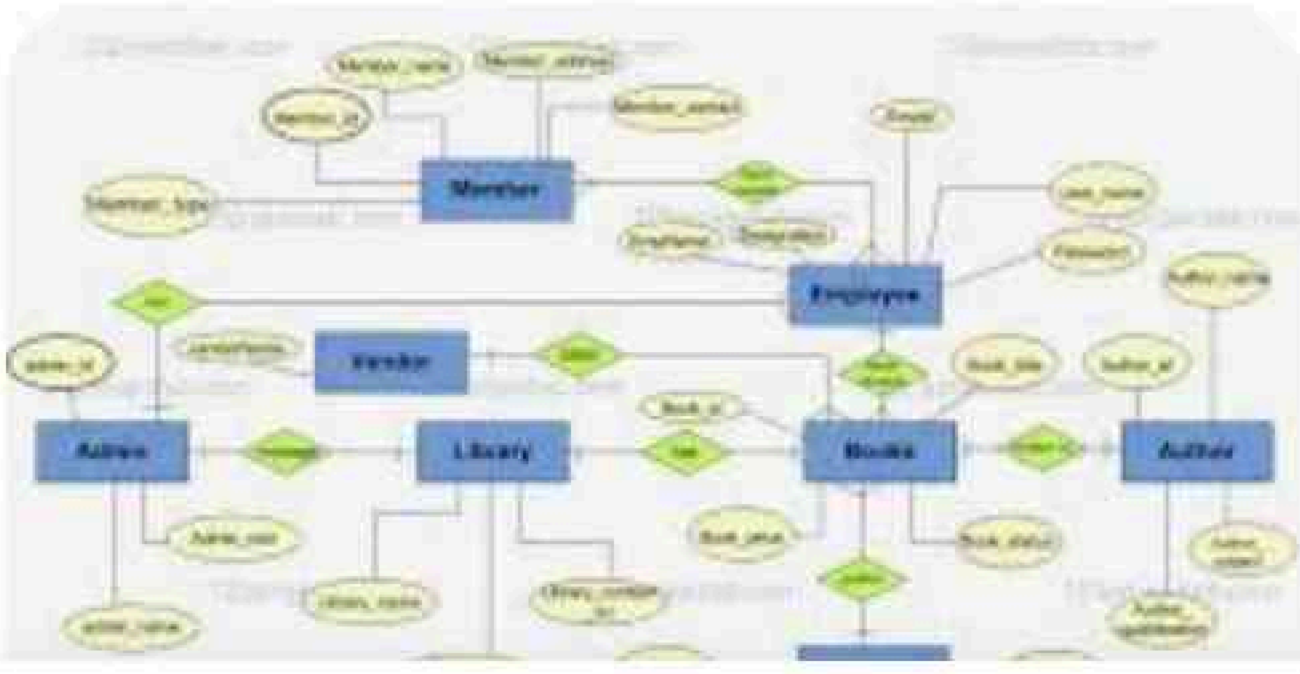
ResearchGate

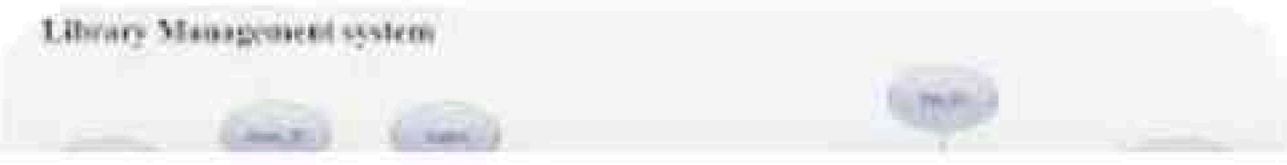
E-R Diagram for Library M...



GeeksforGeeks

ER diagram of Library Ma...





Discover

Search

Saved

21:52 83

kelatebaru

T_1	T_2
read(A)	
write(A)	
	read(A)
	write(A)
read(B)	
write(B)	
	read(B)
	write(B)

Before swapping

T_1	T_2
read(A)	
write(A)	
read(B)	
write(B)	
	read(A)
	write(A)
	read(B)
	write(B)

After swapping

Conflict Serializability in database

Visit >

Images may be subject to copyright. **Learn more**

Share   Save

Transaction 1	Transaction 2
Write(A)	
	Read(A)
	Read(B)
Write(B)	

Serializable schedule

dipelle.com.br

DBMS Conflict Serializabl...

Consider the following schedule:

Time	T1	T2	T3	T4
t1	begin			
t2	read(x)			begin
t3	write(x)			
t4				read(x)
t5	read(z)		begin	
t6			read(y)	
t7			write(y)	
t8		begin		read(y)
t9		read(z)		Commit
t10		write(z)		
t11		commit	read(z)	
t12	write(z)		Commit	
t13	commit			

Table 2: Schedule S_3

Stack Overflow

database - Is this conflict...

T1	T2	T3
Read(A)		
	Write(A)	
Write(A)		
		Write(A)

Javatpoint

DBMS View Serializability...

Non-Serial Schedule

Time	T1	T2
	READ (A); WRITE (A)	
		READ (A); WRITE (A)
	READ (B); WRITE (B);	
		READ (B);

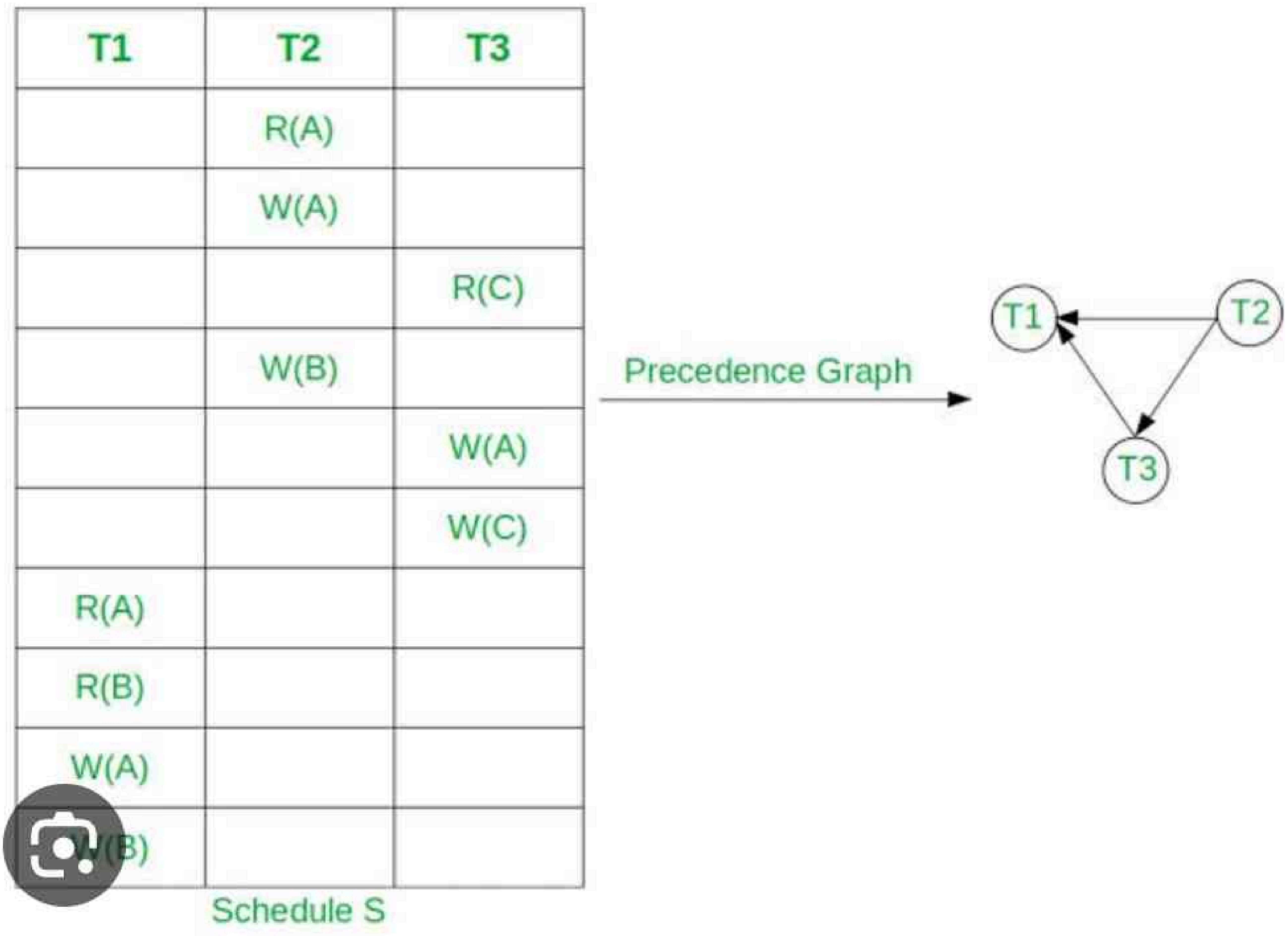
Serial Schedule

Time	T1	T2
	READ (A); WRITE (A); READ (B); WRITE (B);	
		READ (A); WRITE (A); READ (B);

Discover   Search   Saved

21:53 83

GeeksforGeeks



T1	T2	T3
	R(A)	
	W(A)	
		R(C)
	W(B)	
		W(A)
		W(C)
R(A)		
R(B)		
W(A)		
W(B)		

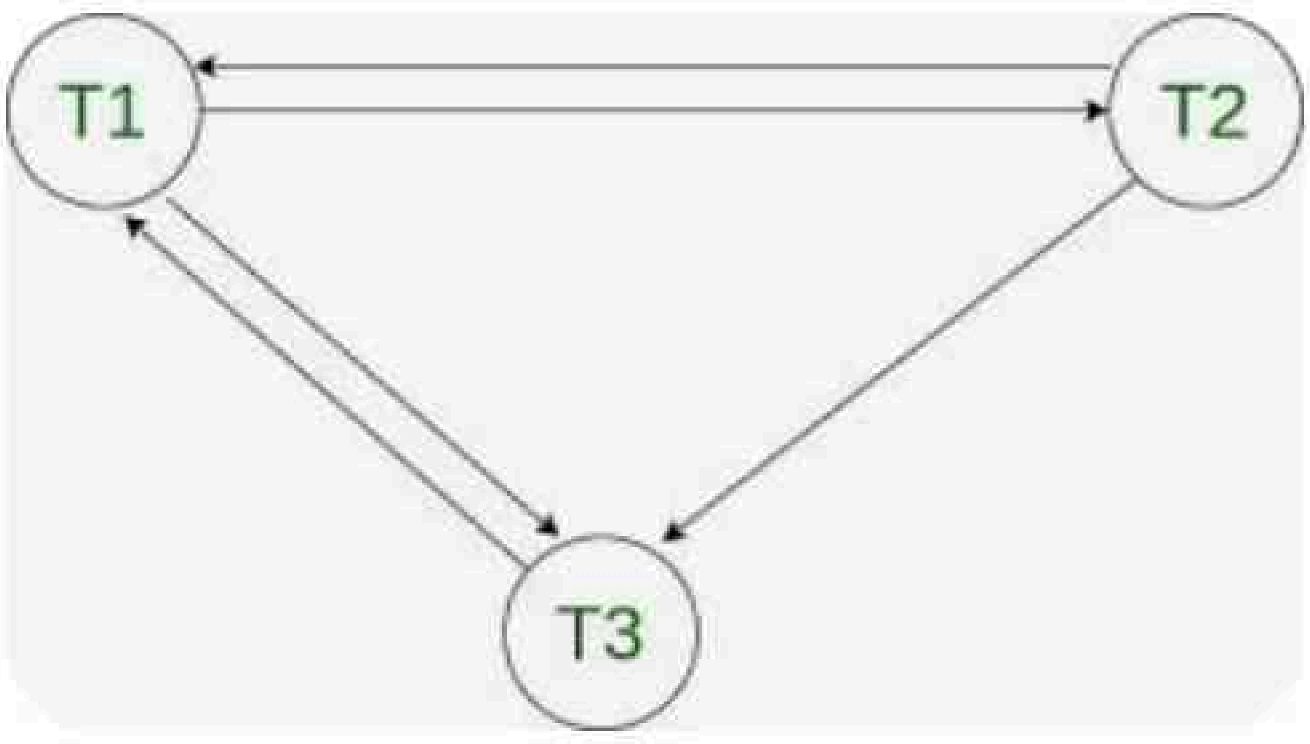
Equivalent Serial Schedule of Conflict Serializable Schedule...

Visit >

Images may be subject to copyright. **Learn more**

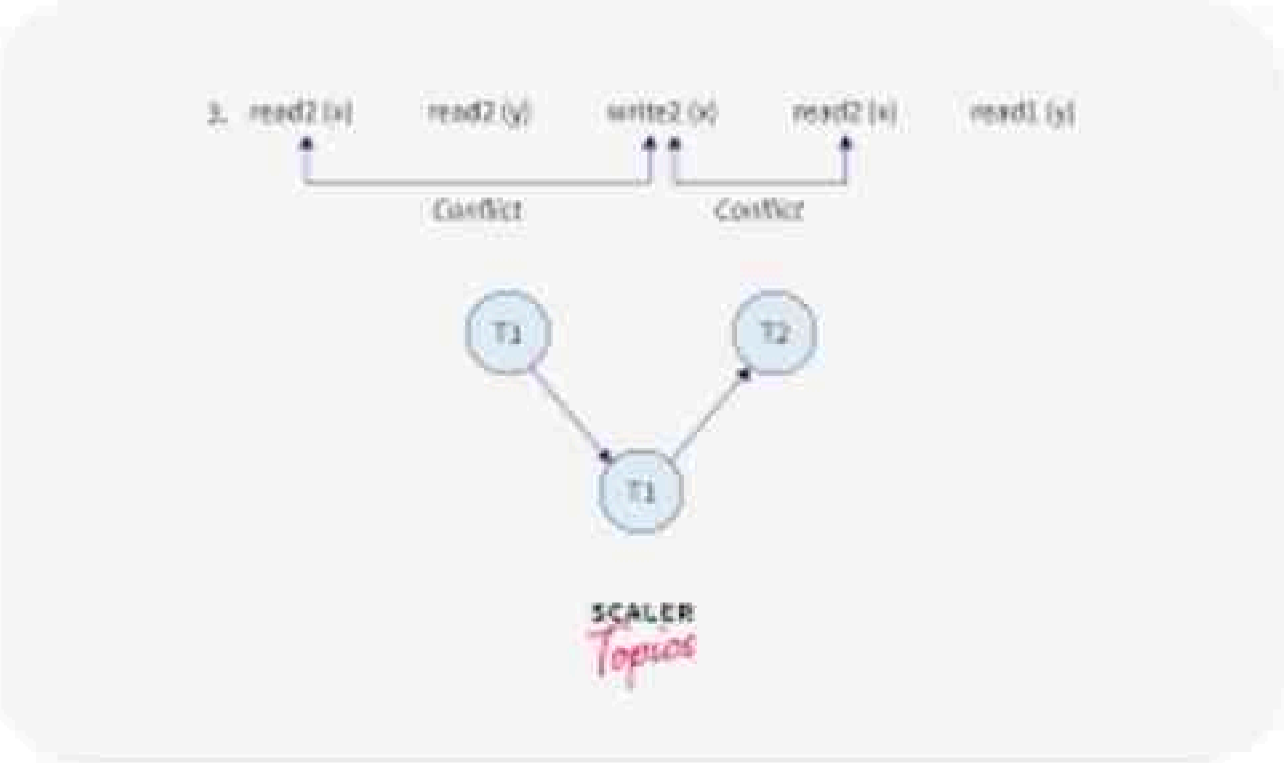
Share

Save



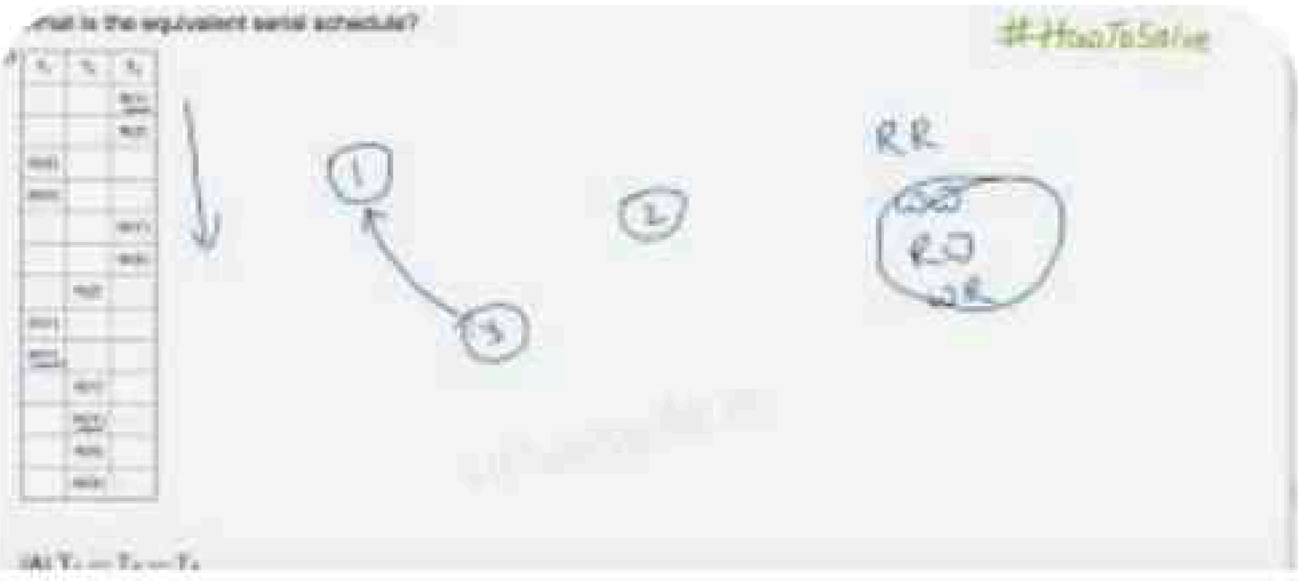
GeeksforGeeks

Difference between Confl...



Scaler

Conflict Serializability in ...



T1	T2
R(A)	
A = A-10	

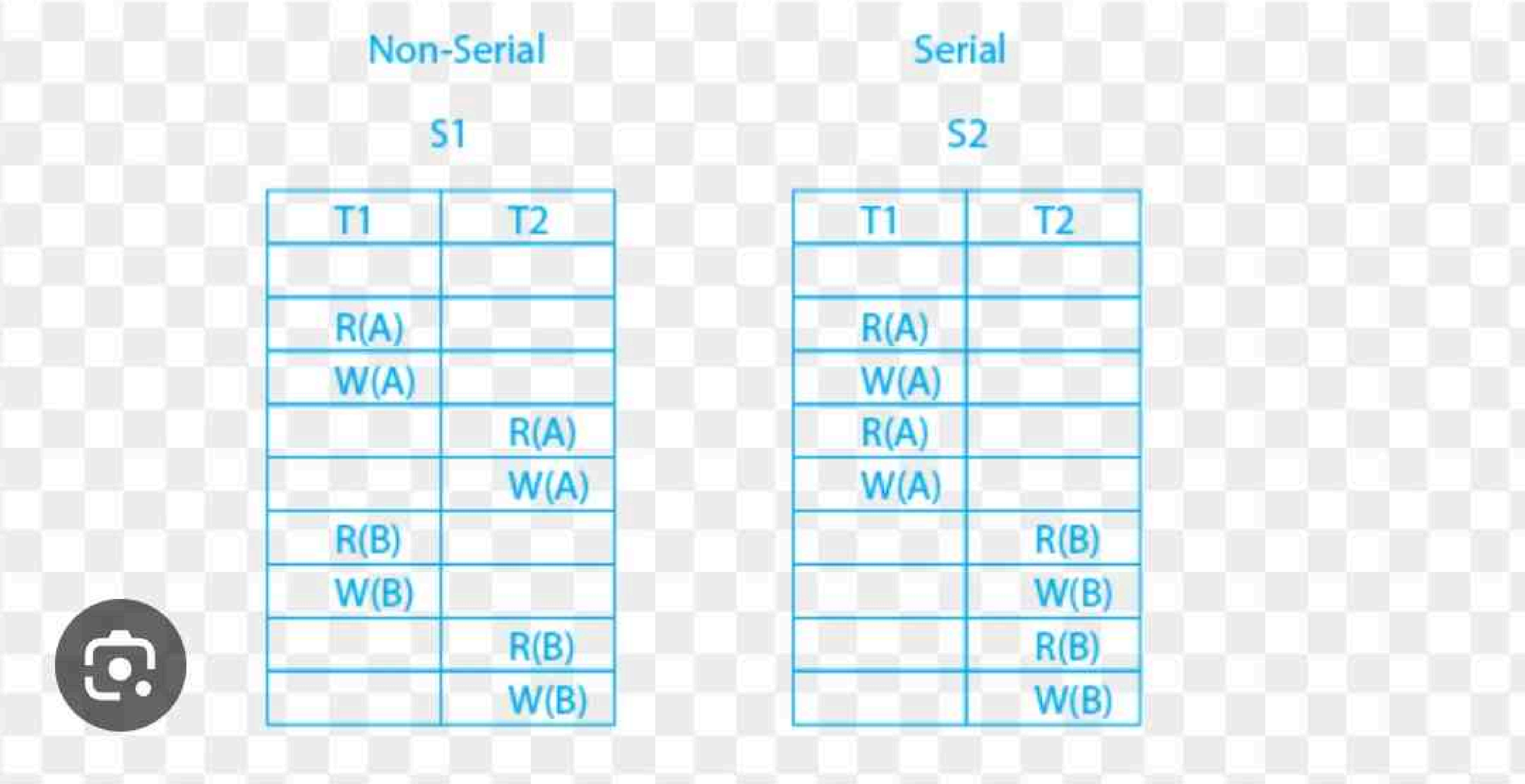
Discover

Search

Saved

21:56 83

PrepBytes

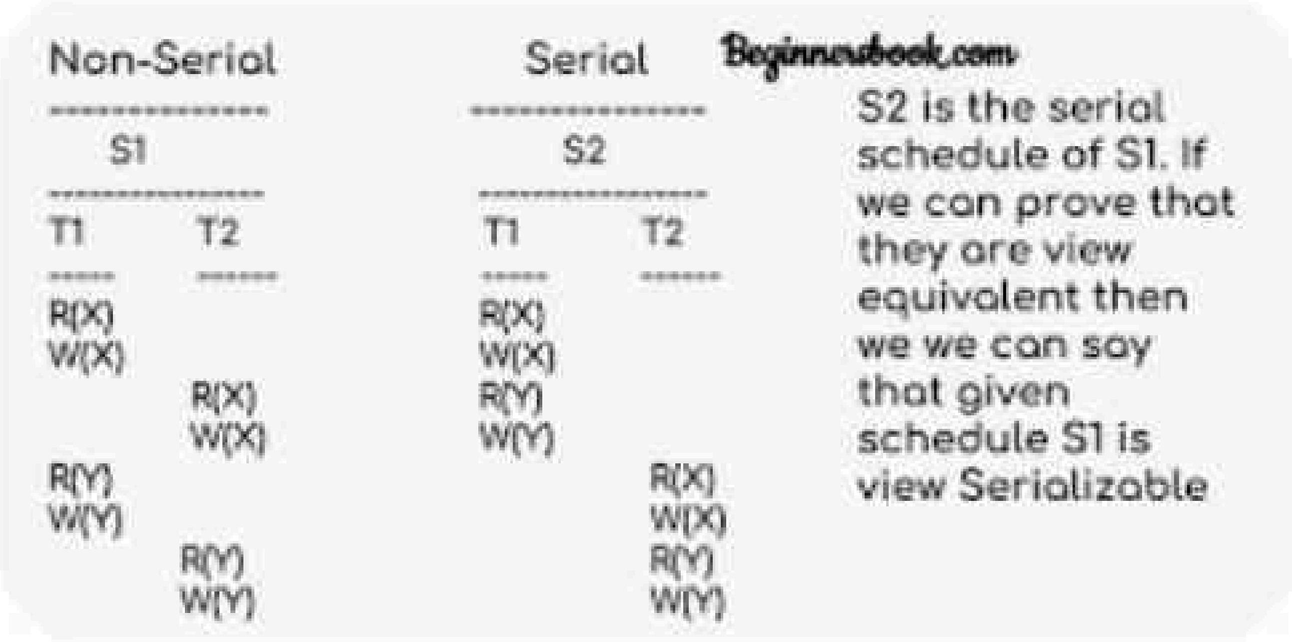


View Serializability in DBMS

Visit >

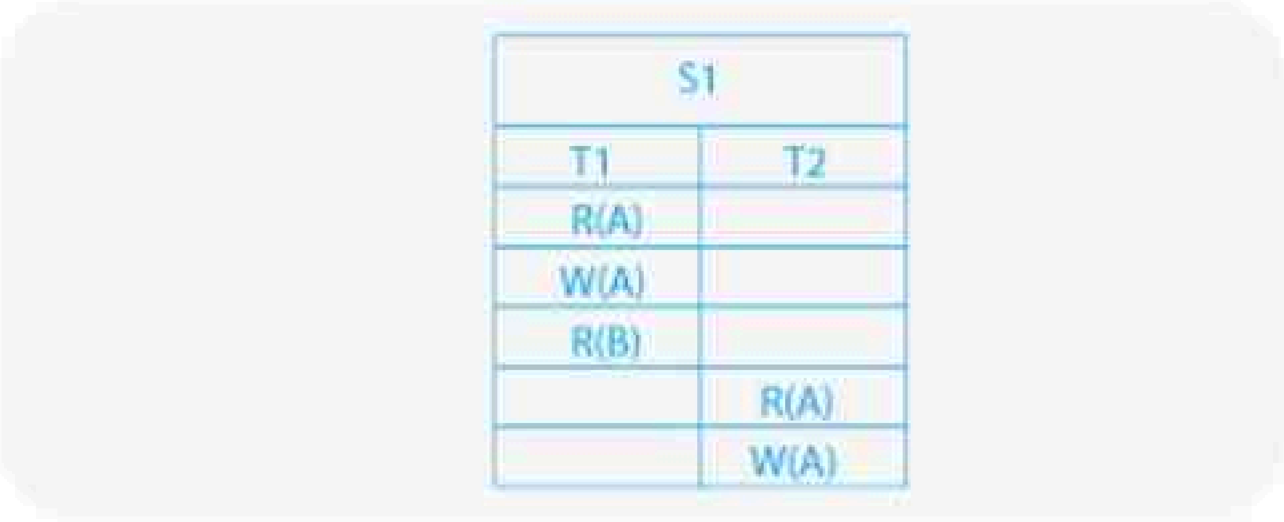
Share

Save



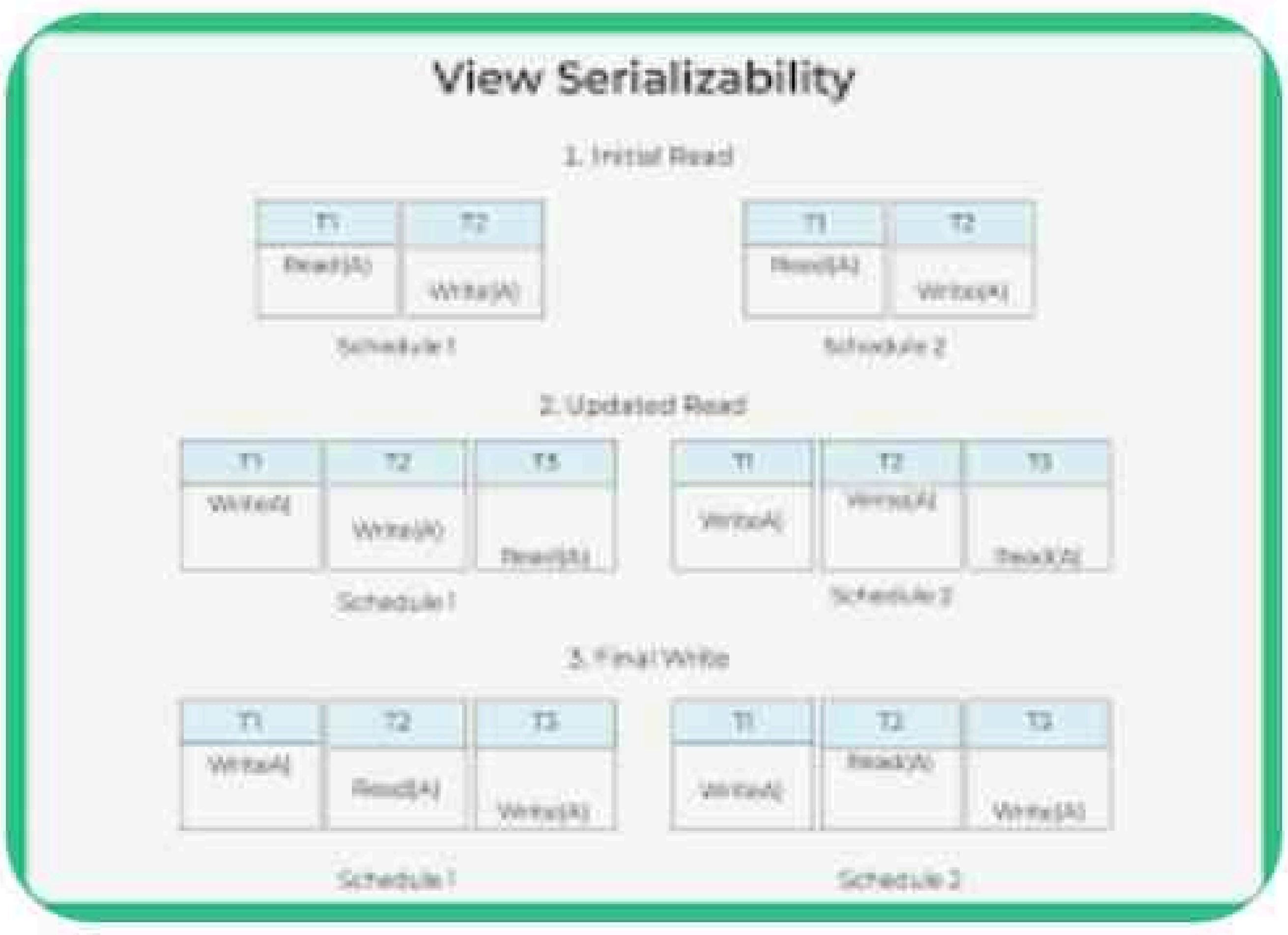
BeginnersBook

DBMS View Serializability



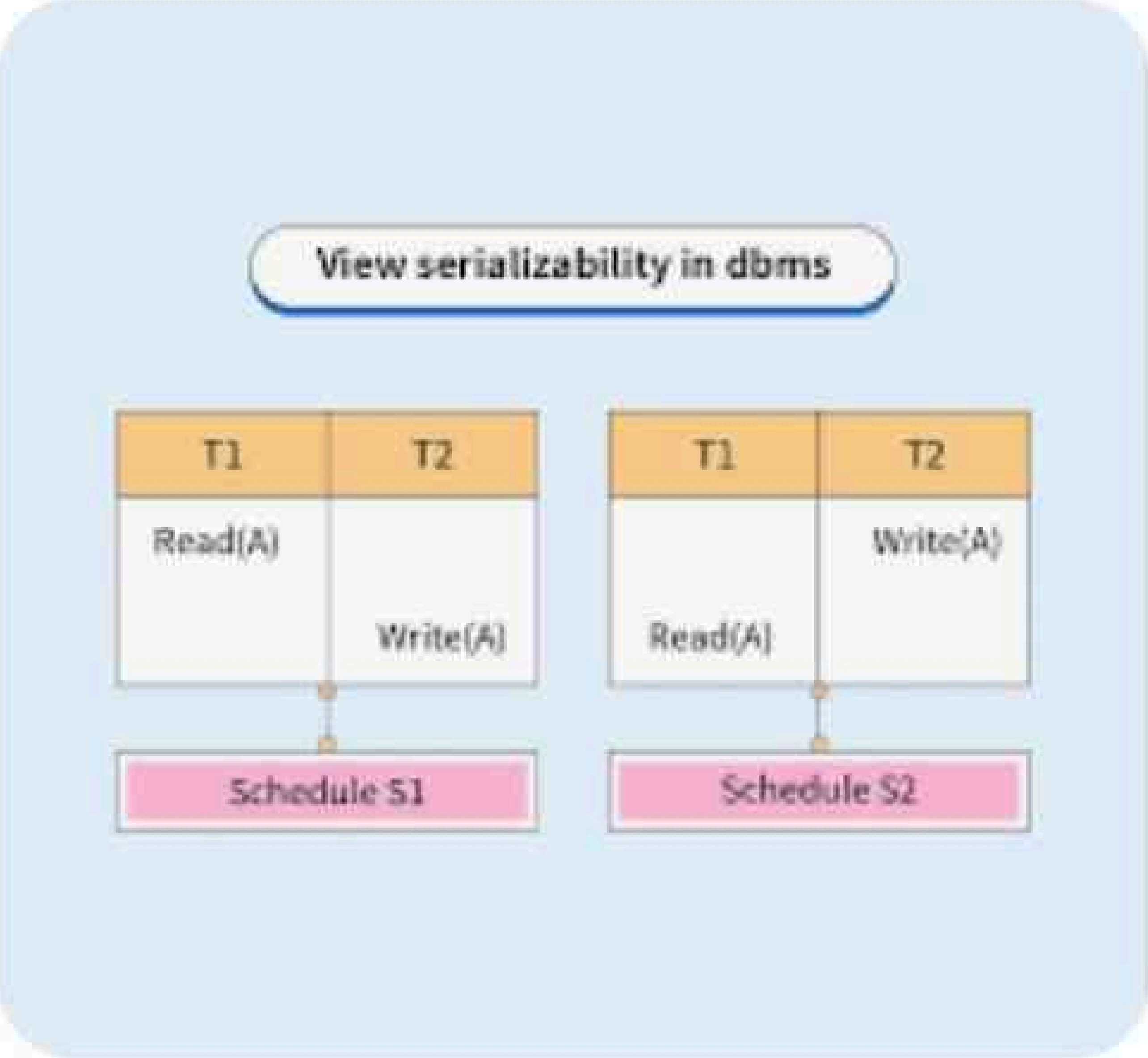
PrepBytes

Conflict Serializability in ...



PrepInsta

View Serializability in DB...



Scaler

Discover

Search

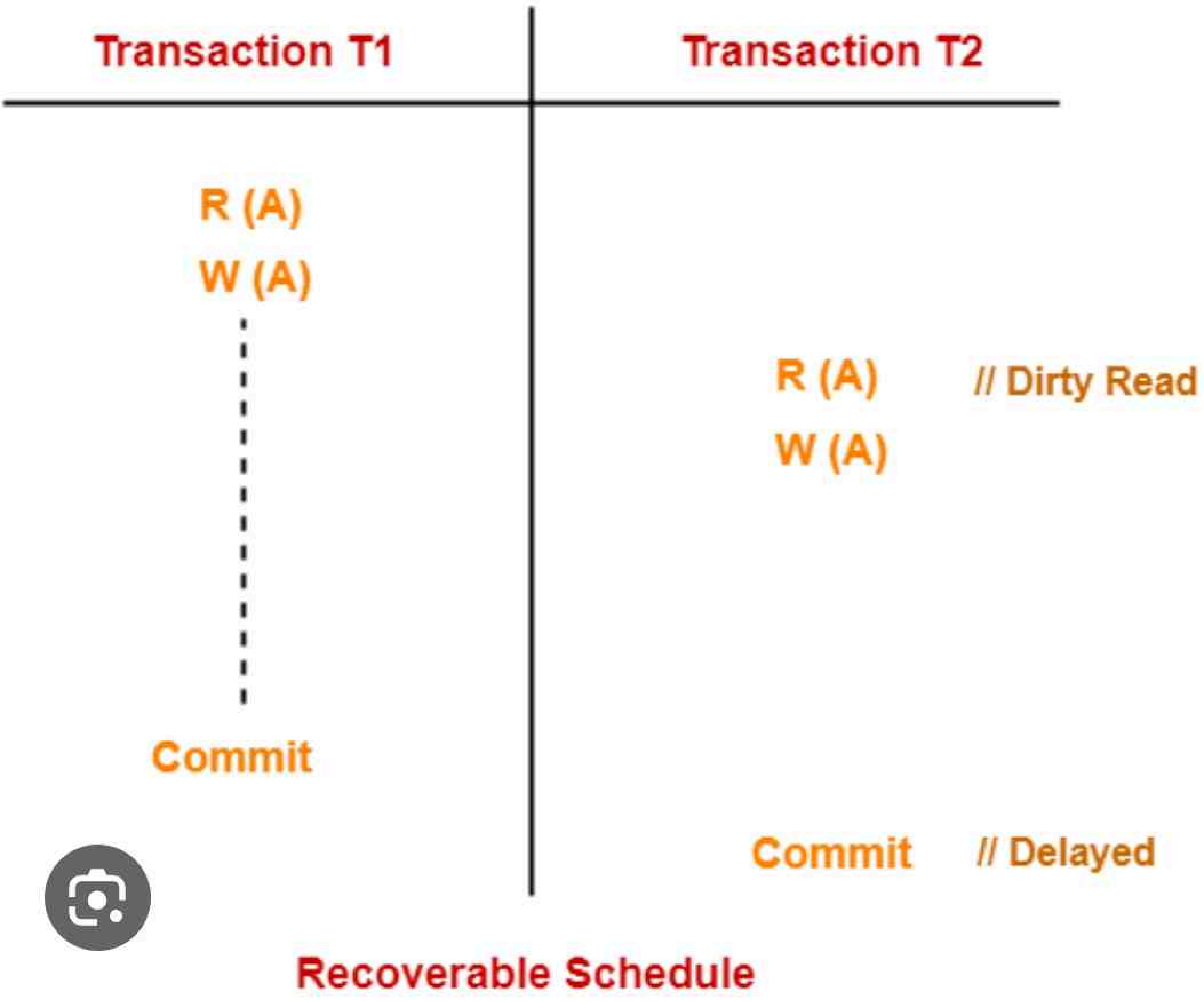
Saved

T1	T2
R (A)	
A = A + 10	
	R (A)
	A = A + 10
W (A)	
	W (A)
R (B)	
B = B + 20	
	R (B)
	B = B x 1.1
W (B)	
	W (B)

View Serializability in DBMS | Practice Problems | Gate...

21:57

Gate Vidyalay

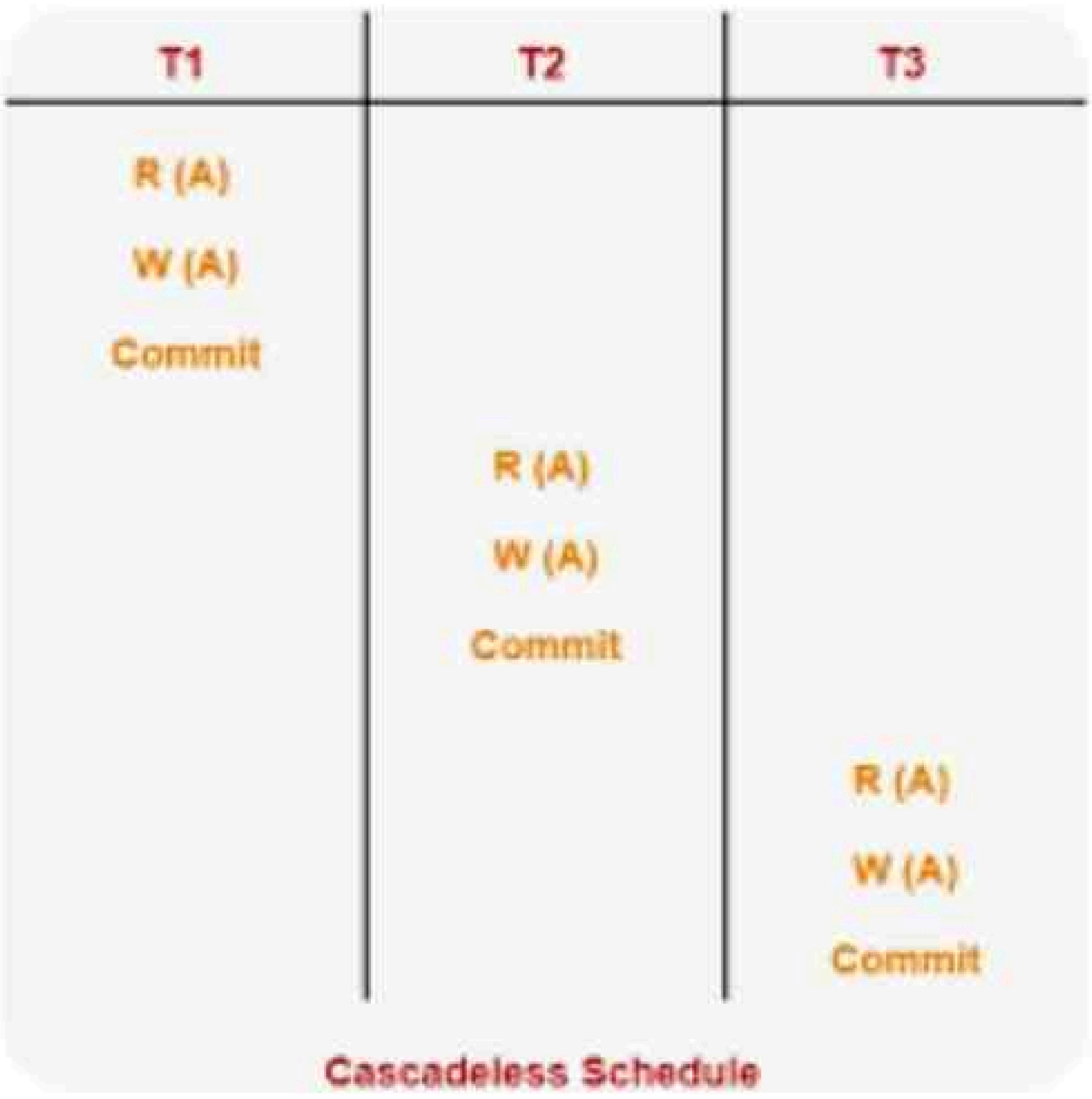


Recoverability in DBMS | Recoverable Schedule | Gate...

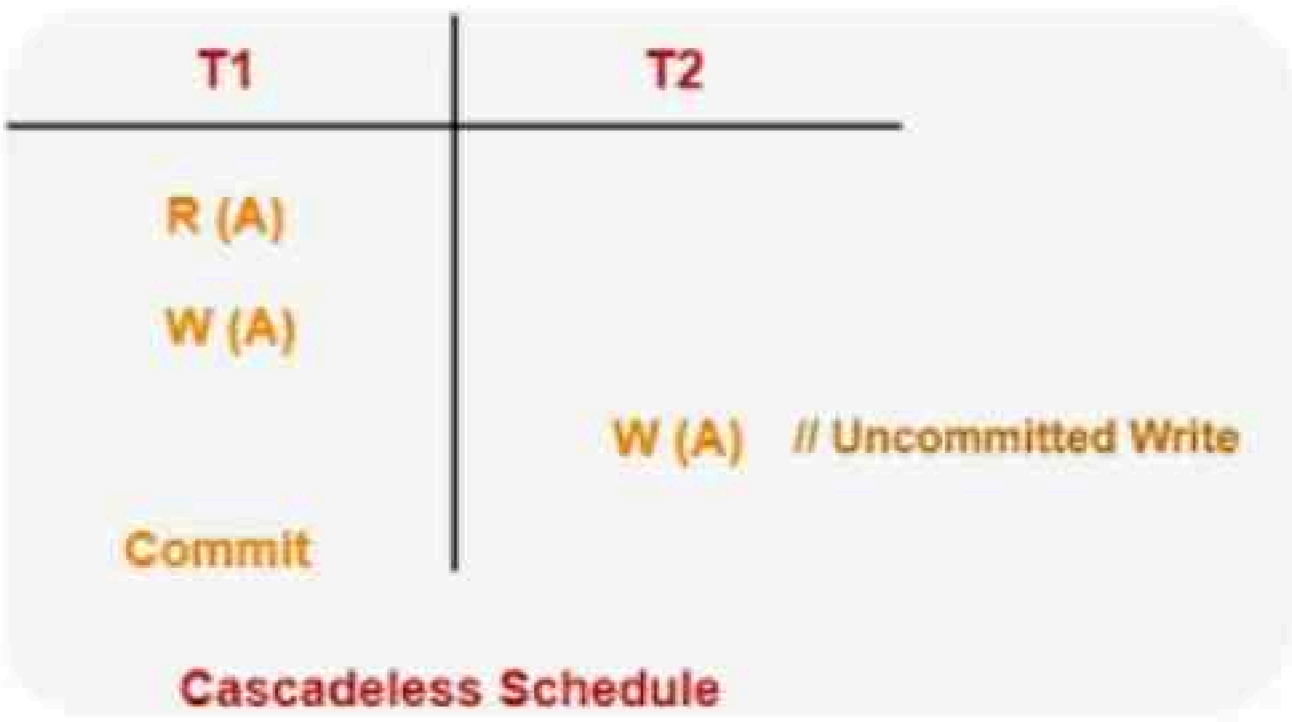
Visit >

Share

Save

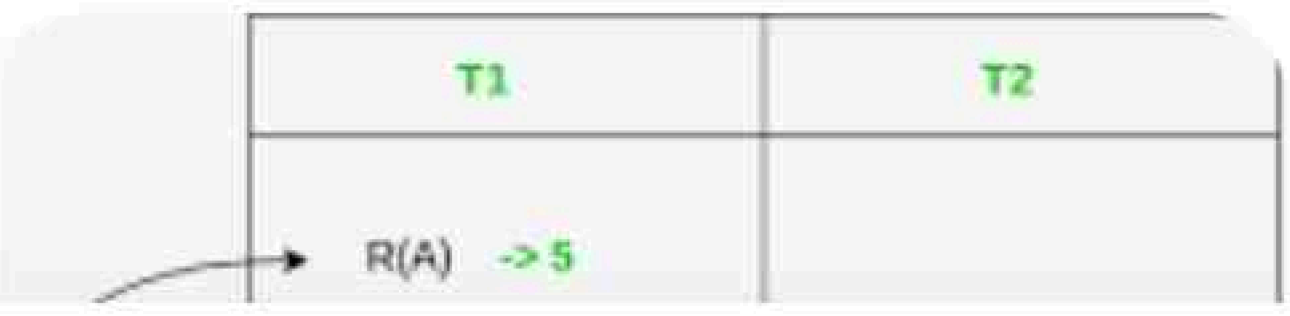


Gate Vidyalay



Gate Vidyalay

serializability and recover...



Discover

Search

Saved

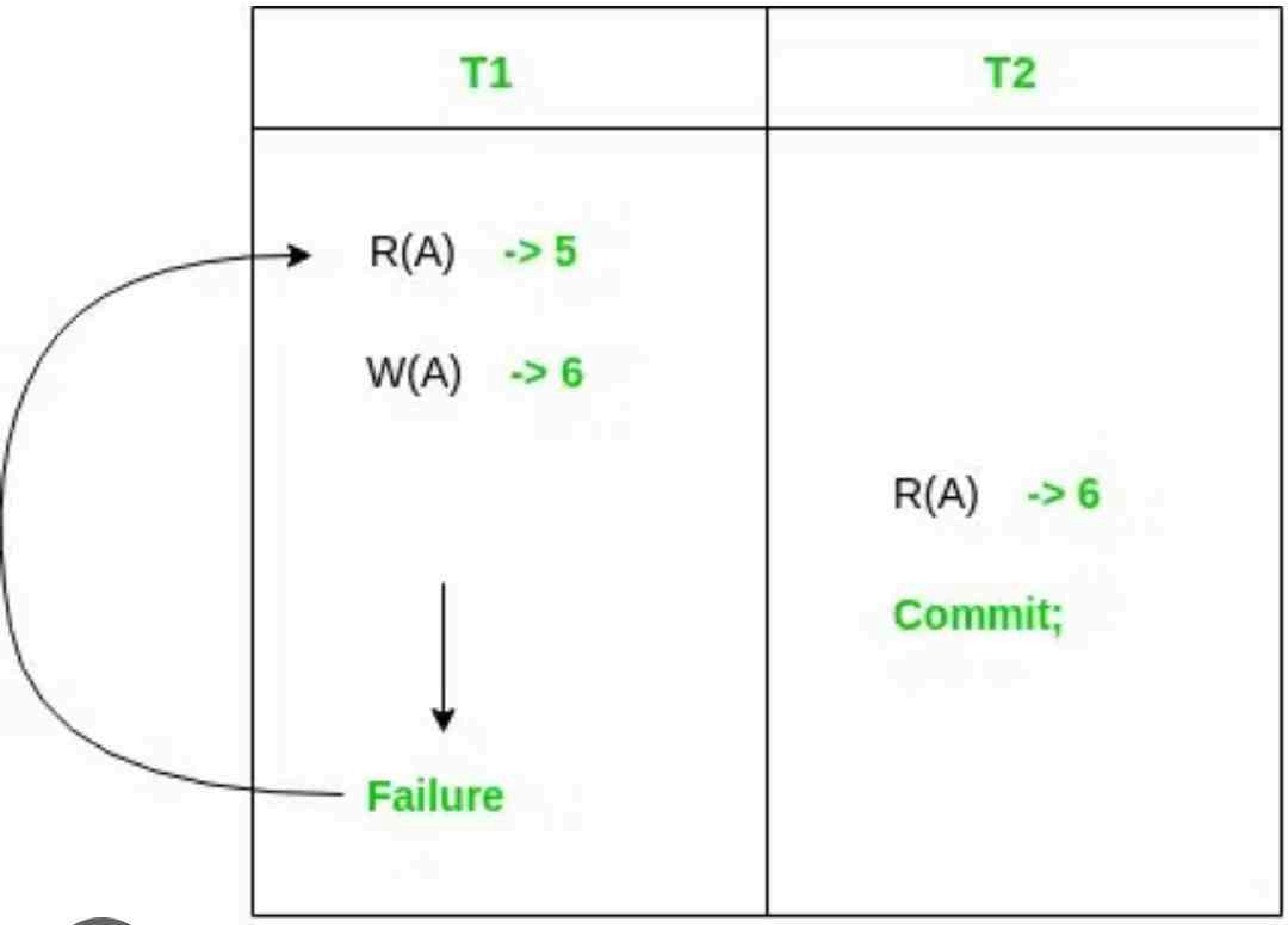


Figure - Dirty Read Problem

Types of Schedules based Recoverability in DBMS -...

Visit >

Share Save

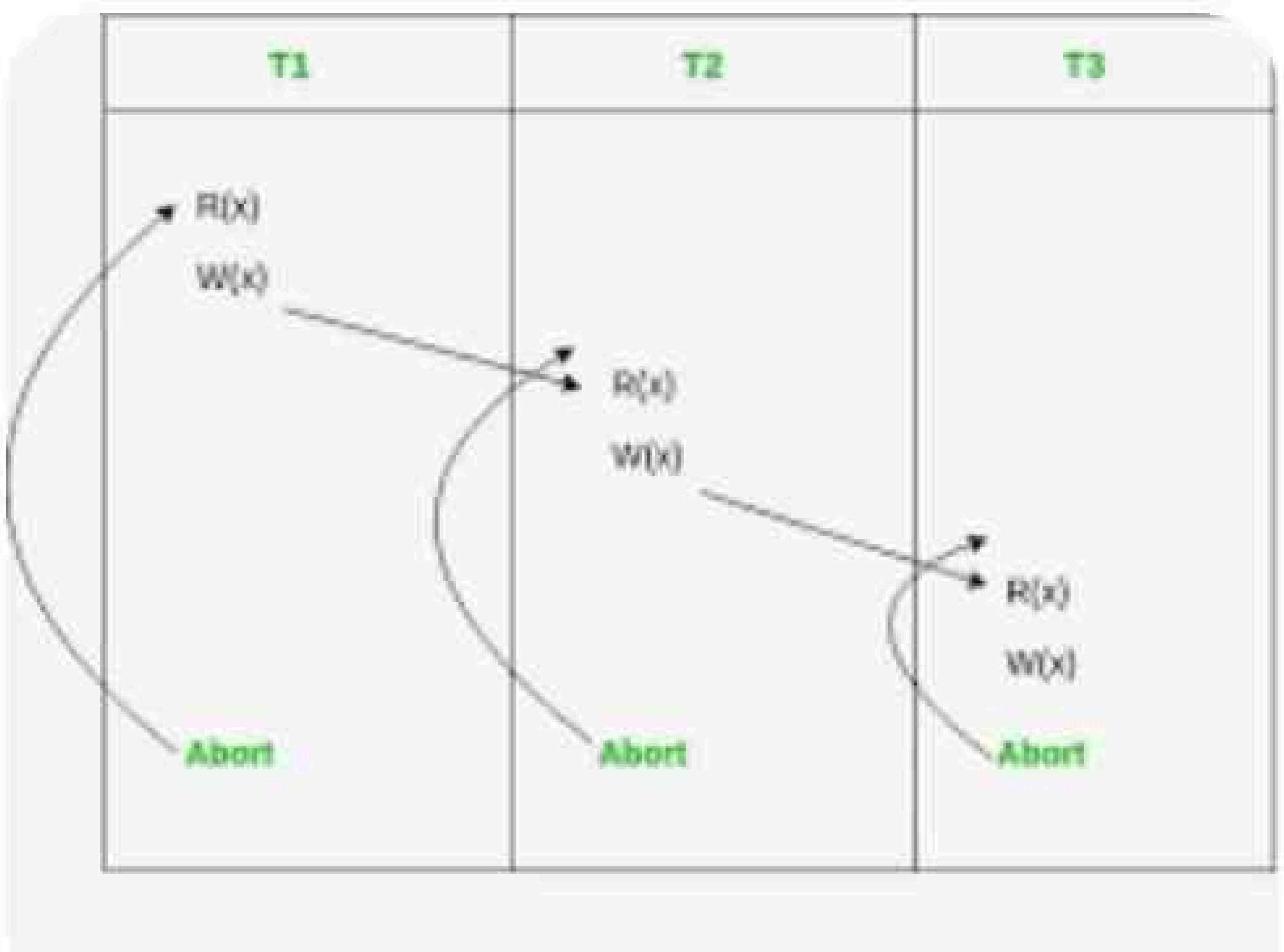


Figure - Cascading Abort

GeeksforGeeks

Types of Schedules base...

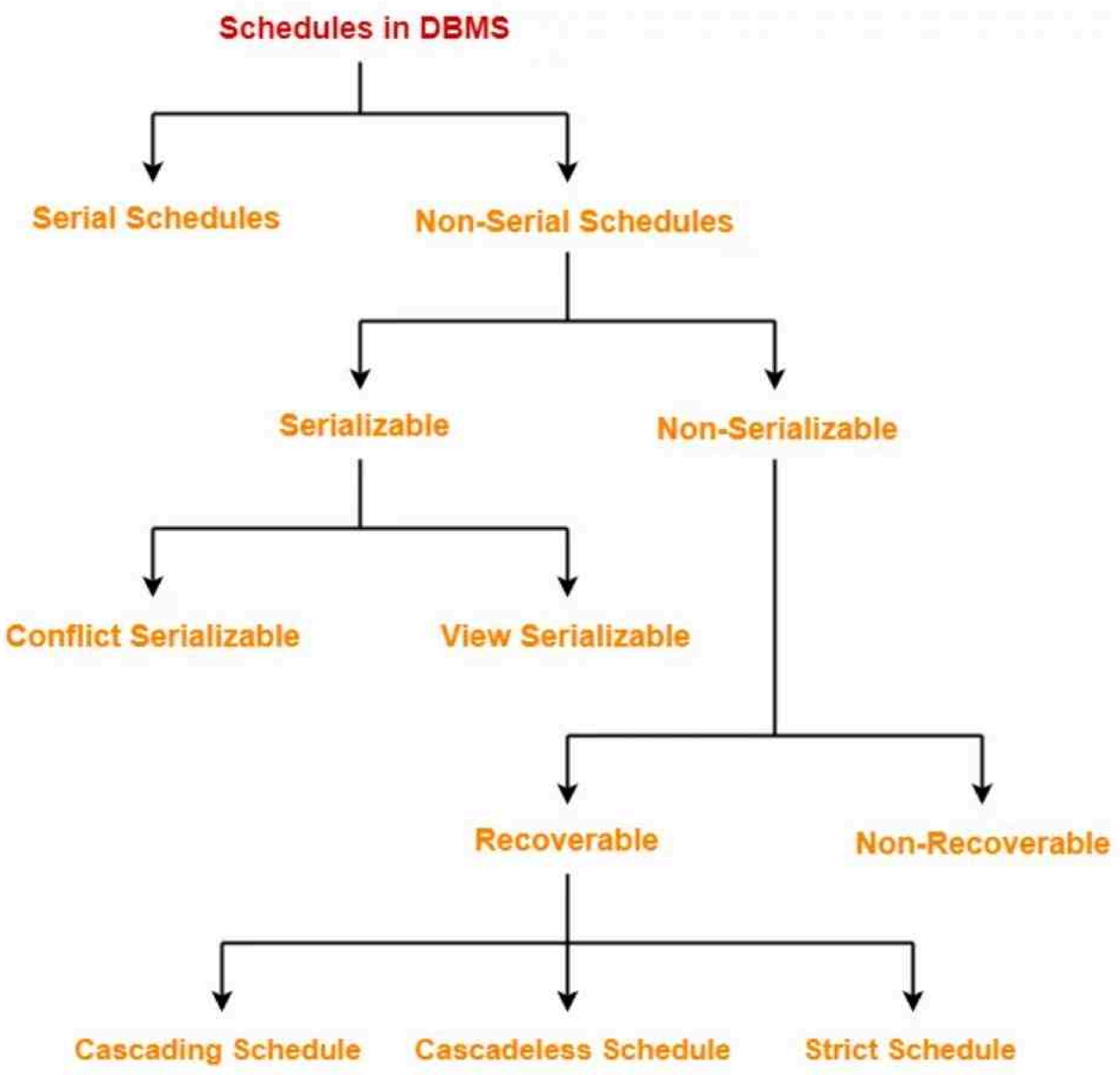
Figure - Strict Schedule

GeeksforGeeks

Types of Schedules base

[Set Up And Restore Previous Versions](#)

- A schedule is the order in which the operations of multiple transactions appear for execution.
- Non-serial schedules may be serializable or non-serializable.



In this article, we will discuss about Non-Serializable Schedules.

Also read- Serializability in DBMS

Irrecoverable Schedules-

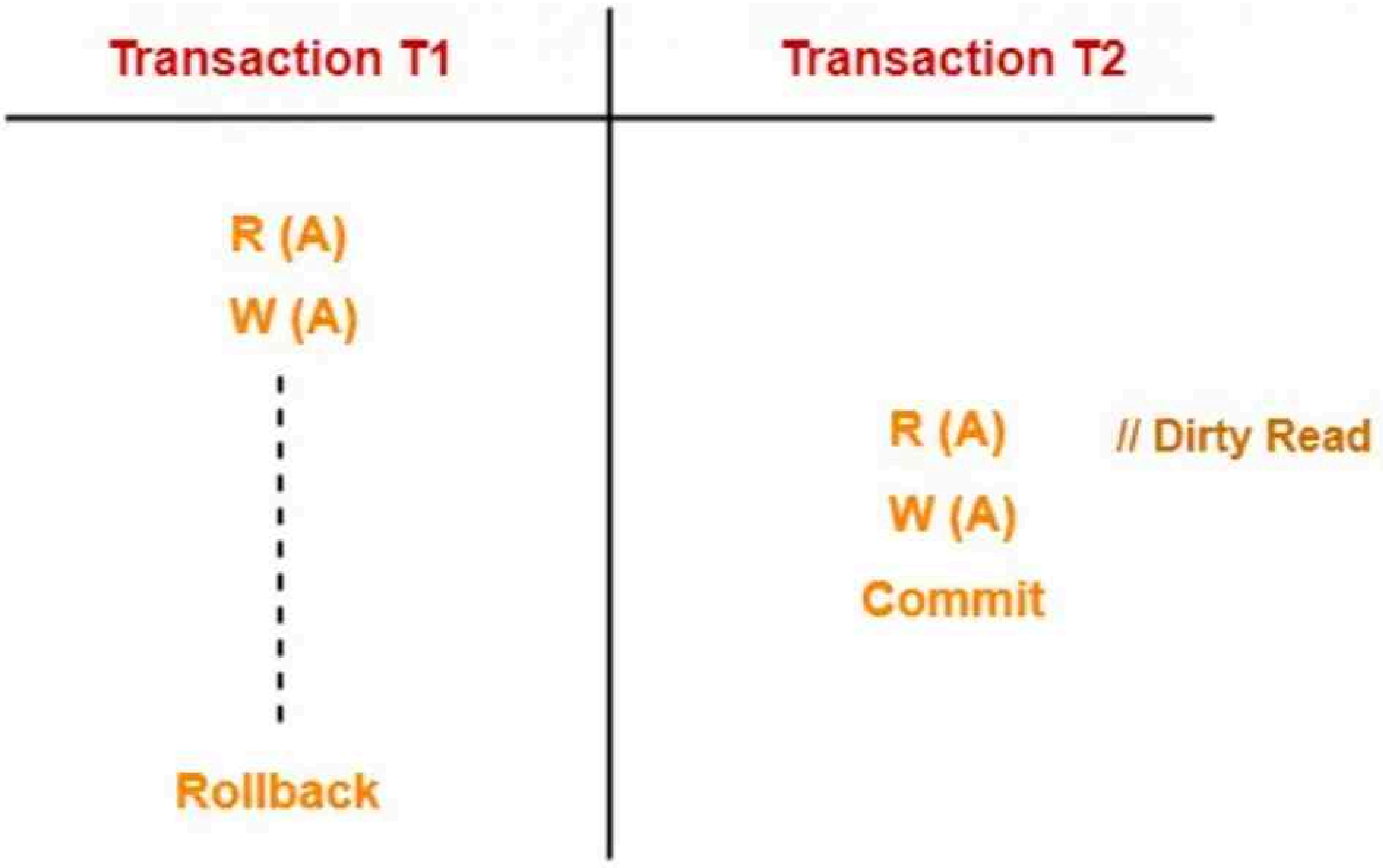
If in a schedule,

- A transaction performs a dirty read operation from an uncommitted transaction
- And commits before the transaction from which it has read the value

then such a schedule is known as an **Irrecoverable Schedule**.

Example-

Consider the following schedule-

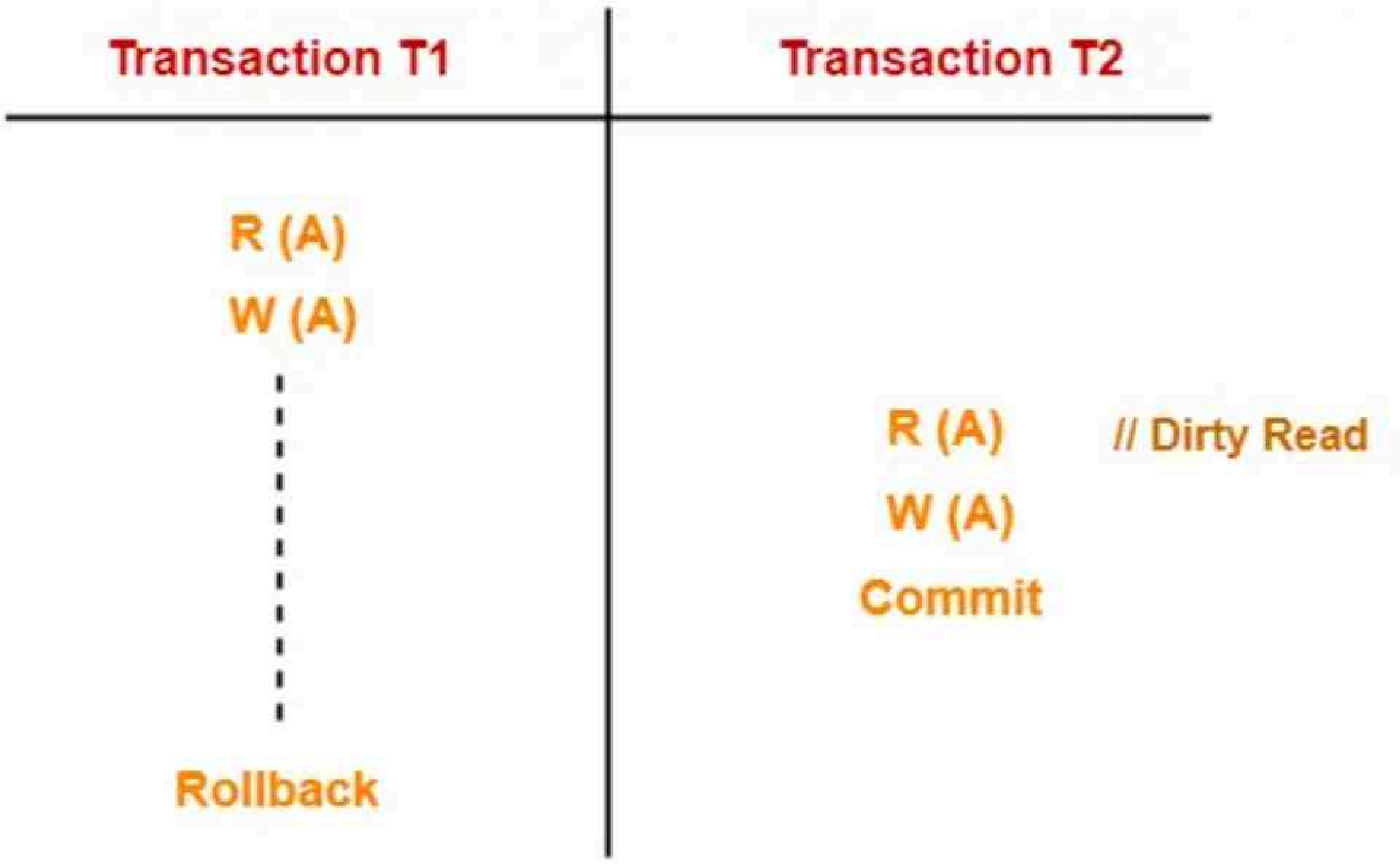


Irrecoverable Schedule

then such a schedule is known as an **Irrecoverable Schedule**.

Example-

Consider the following schedule-



Irrecoverable Schedule

Here,

- T2 performs a dirty read operation.
- T2 commits before T1.
- T1 fails later and roll backs.
- The value that T2 read now stands to be incorrect.
- T2 can not recover since it has already committed.

Recoverable Schedules-

If in a schedule,

- A transaction performs a dirty read operation from an uncommitted transaction
- And its commit operation is delayed till the uncommitted transaction either commits or roll backs

then such a schedule is known as a **Recoverable Schedule**.

Here,

- The commit operation of the transaction that performs the dirty read is delayed.
- This ensures that it still has a chance to recover if the uncommitted transaction fails later.

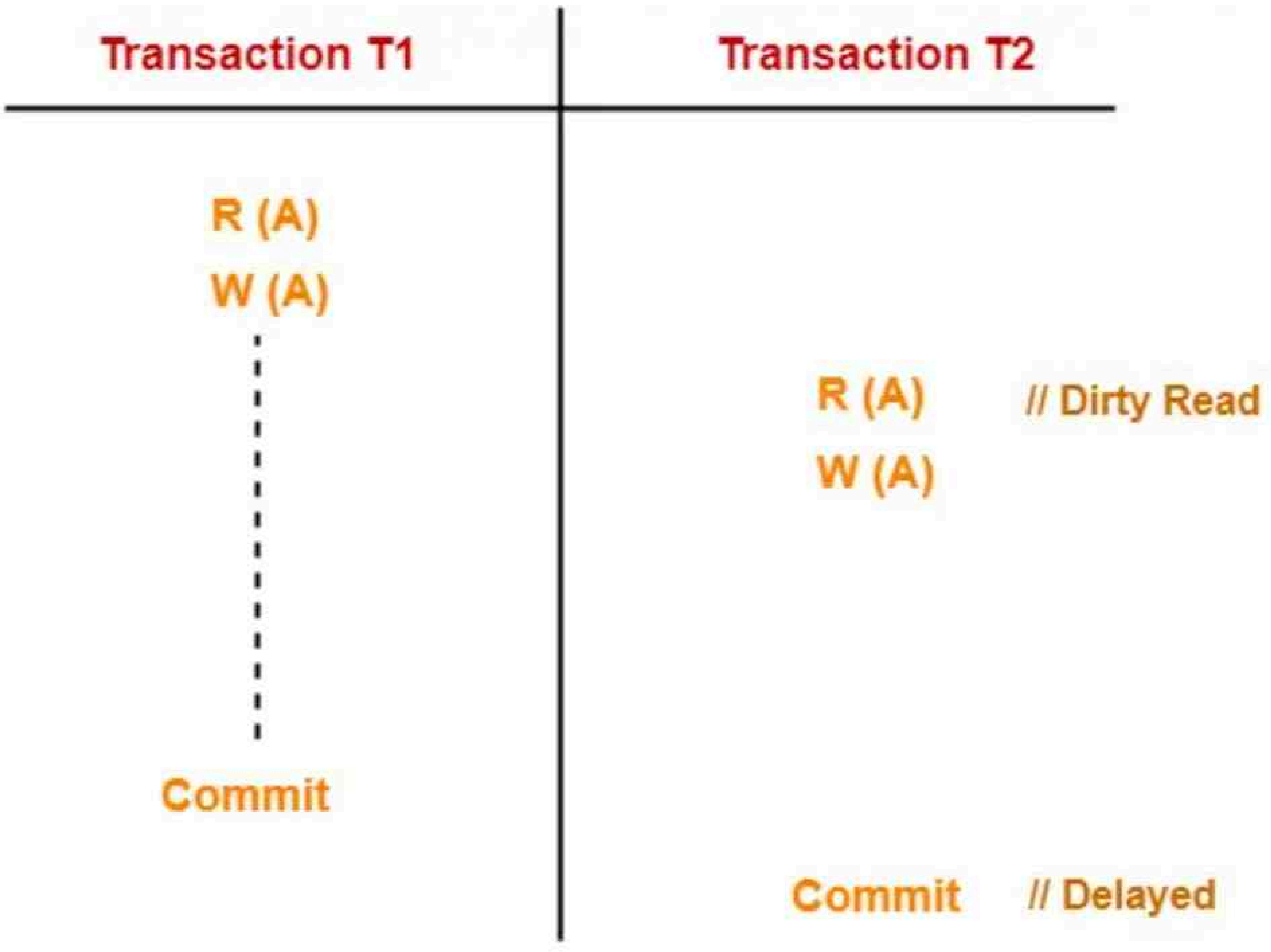
Example-

Consider the following schedule-

Transaction T1	Transaction T2
R (A)	
W (A)	

Example-

Consider the following schedule-



Recoverable Schedule

Here,

- T2 performs a dirty read operation.
- The commit operation of T2 is delayed till T1 commits or roll backs.
- T1 commits later.
- T2 is now allowed to commit.
- In case, T1 would have failed, T2 has a chance to recover by rolling back.

Checking Whether a Schedule is Recoverable or Irrecoverable-

Method-01:

Check whether the given schedule is conflict serializable or not.

- If the given schedule is conflict serializable, then it is surely recoverable. Stop and report your answer.
- If the given schedule is not conflict serializable, then it may or may not be recoverable. Go and check using other methods.

Thumb Rules

- All conflict serializable schedules are recoverable.
- All recoverable schedules may or may not be conflict serializable.

Method-02:

Check if there exists any dirty read operation.

(Reading from an uncommitted transaction is called as a dirty read)

Thumb Rules

- All conflict serializable schedules are recoverable.
- All recoverable schedules may or may not be conflict serializable.

Method-02:

Check if there exists any dirty read operation.

(Reading from an uncommitted transaction is called as a dirty read)

- If there does not exist any dirty read operation, then the schedule is surely recoverable. Stop and report your answer.
- If there exists any dirty read operation, then the schedule may or may not be recoverable.

If there exists a dirty read operation, then follow the following cases-

Case-01:

Case-01:

If the commit operation of the transaction performing the dirty read occurs before the commit or abort operation of the transaction which updated the value, then the schedule is irrecoverable.

Case-02:

If the commit operation of the transaction performing the dirty read is delayed till the commit or abort operation of the transaction which updated the value, then the schedule is recoverable.

Thumb Rule

No dirty read means a recoverable schedule.

Next Article- Cascading Schedule | Cascading Rollback | Cascadeless Schedule

Cascading Schedule-

- If in a schedule, failure of one transaction causes several other dependent transactions to rollback or abort, then such a schedule is called as a **Cascading Schedule** or **Cascading Rollback** or **Cascading Abort**.
- It simply leads to the wastage of CPU time.

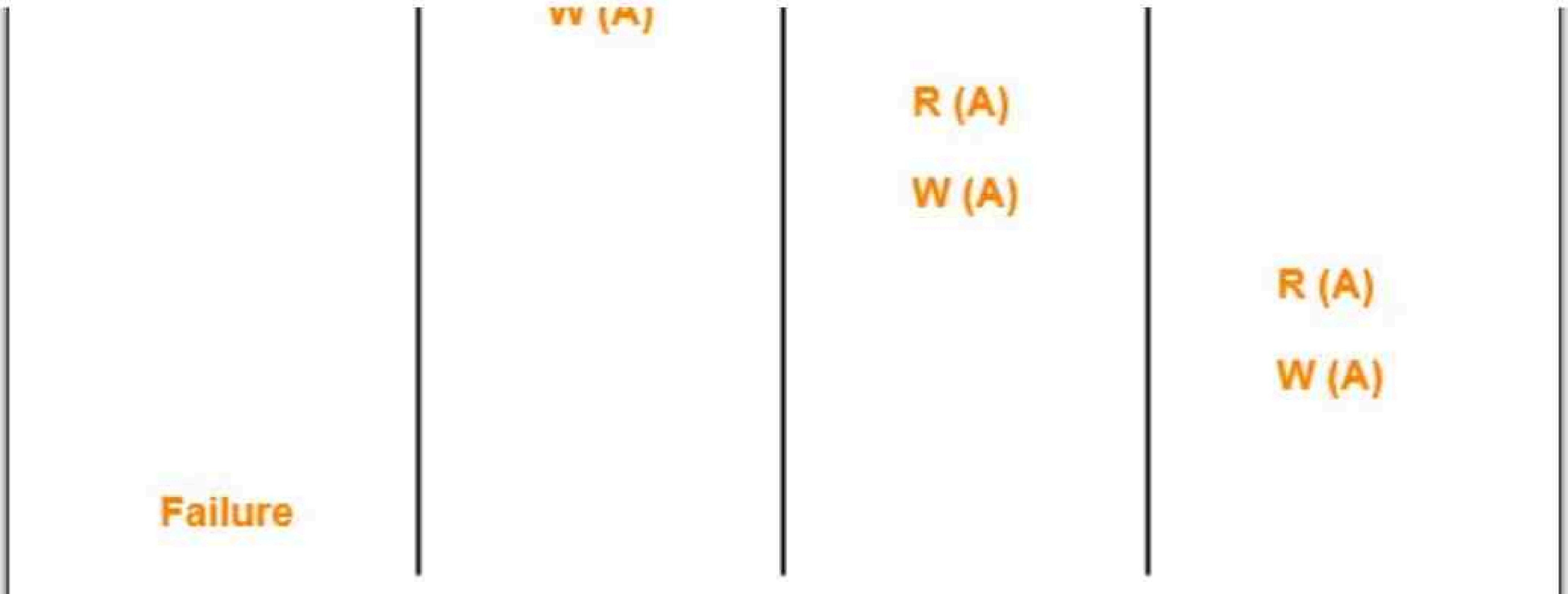
Example-

T1	T2	T3	T4
R (A)			
W (A)			
	R (A)		
	W (A)		
		R (A)	
		W (A)	
			R (A)
			W (A)
Failure			

Cascading Recoverable Schedule

Here,

- Transaction T2 depends on transaction T1.
- Transaction T3 depends on transaction T2.
- Transaction T4 depends on transaction T3.



Cascading Recoverable Schedule

Here,

- Transaction T2 depends on transaction T1.
- Transaction T3 depends on transaction T2.
- Transaction T4 depends on transaction T3.

In this schedule,

- The failure of transaction T1 causes the transaction T2 to rollback.
- The rollback of transaction T2 causes the transaction T3 to rollback.
- The rollback of transaction T3 causes the transaction T4 to rollback.

Such a rollback is called as a **Cascading Rollback**.

NOTE-

If the transactions T2, T3 and T4 would have committed before the failure of transaction T1, then the schedule would have been irrecoverable.

Cascadeless Schedule-

If in a schedule, a transaction is not allowed to read a data item until the last transaction that has written it is committed or aborted, then such a schedule is called as a **Cascadeless Schedule**.

In other words,

- Cascadeless schedule allows only committed read operations.
- Therefore, it avoids cascading roll back and thus saves CPU time.

Example-

T1	T2	T3
R (A)		
W (A)		
Commit		
	R (A)	
	W (A)	
	Commit	
		R (A)
		W (A)
		Commit

Cascadeless Schedule

	W (A)	
	Commit	
		R (A)
		W (A)
		Commit

Cascadeless Schedule

NOTE-

- Cascadeless schedule allows only committed read operations.
- However, it allows uncommitted write operations.

Example-

T1	T2
R (A)	
W (A)	
	W (A) // Uncommitted Write
Commit	

Cascadeless Schedule

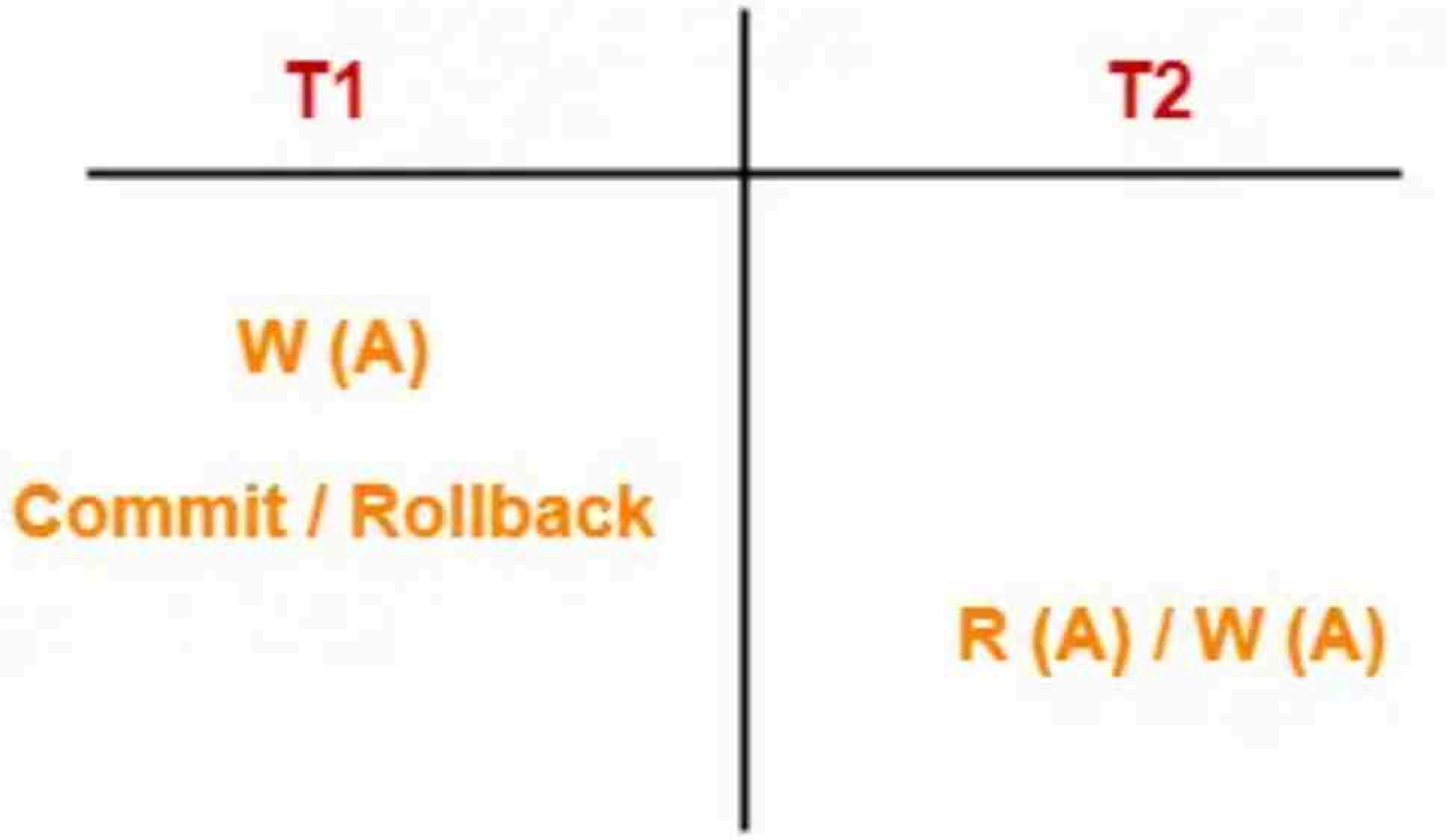
Strict Schedule-

If in a schedule, a transaction is neither allowed to read nor write a data item until the last transaction that has written it is committed or aborted, then such a schedule is called as a **Strict Schedule**.

In other words,

- Strict schedule allows only committed read and write operations.
- Clearly, strict schedule implements more restrictions than cascadeless schedule.

Example-

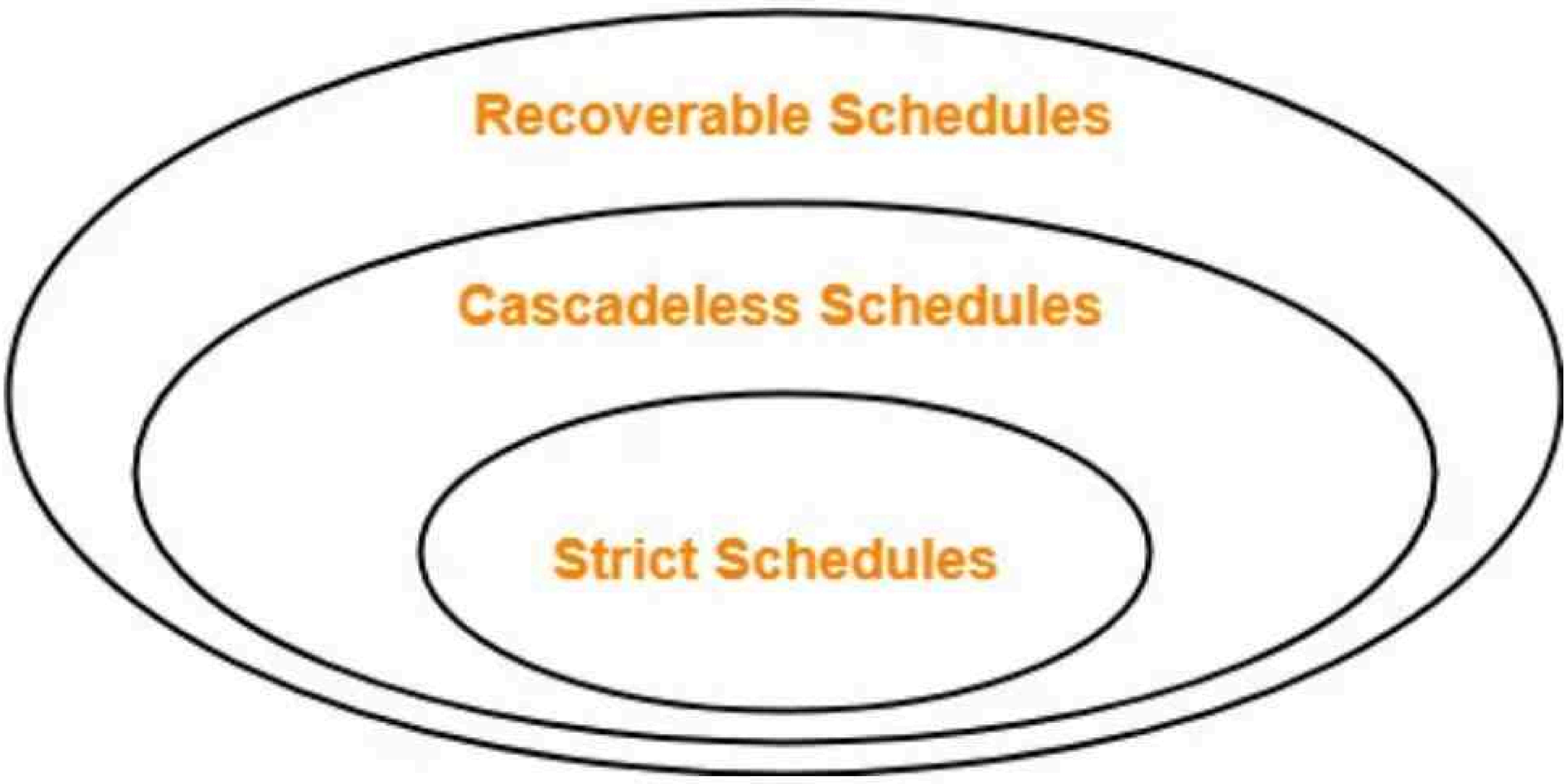


Strict Schedule

Remember-

Remember-

- Strict schedules are more strict than cascadeless schedules.
- All strict schedules are cascadeless schedules.
- All cascadeless schedules are not strict schedules.



Next Article- **Equivalence of Schedules**

Get more notes and other study material of **Database Management System (DBMS)**.

Watch video lectures by visiting our YouTube channel **LearnVidFun**.

Summary

1. Result Equivalent Schedules-

- If any two schedules generate the same result after their execution, then they are called as result equivalent schedules.
- This equivalence relation is considered of least significance.
- This is because some schedules might produce same results for some set of values and different results for some other set of values.

2. Conflict Equivalent Schedules-

If any two schedules satisfy the following two conditions, then they are called as conflict equivalent schedules-

1. The set of transactions present in both the schedules is same.
2. The order of pairs of conflicting operations of both the schedules is same.

3. View Equivalent Schedules-

We have already discussed about **View Equivalent Schedules**.

T_3	T_4
lock-X(B)	
read(B)	
$B := B - 50$	
write(B)	
	lock-S(A)
	read(A)
	lock-S(B)
lock-X(A)	

Two-Phase Locking Protocol

T_1	T_2
lock-X on A	
write (A)	
	lock-X on B
	write (B)
	wait for **lock-X** on A
wait for **lock-X** on B	

Chapter 10 : Concurrency...

T1	T2
Exclusive_Lock(X)	
X:= X+10	
WRITE (X)	
	Shared_Lock(Y)
	READ (Y)
	Shared_Lock (X)

Deadlock Prevention - Timestamps

- When T_i wants an item locked by T_j, one of the following two rules can be used to avoid deadlock:
- wait-die rule: if $TS(T_i) < TS(T_j)$
  T_i waits
  else T_i aborts and restarts with same TS
- wound-die rule: if $TS(T_i) < TS(T_j)$
  T_j aborts and restarts with same TS
  else T_i waits
- Prevent deadlocks but abort transactions which are not in deadlock
- A transaction my be aborted many times because an older transaction holds an item for a long period of time (need delay in restart)
- Restarted transactions retain their original "seniority" to avoid starvation

Deadlock prevention (Cont.)

- Both in *wait-die* and in *wound-wait* schemes, a rolled back transactions is restarted with its original timestamp. Older transactions thus have precedence over newer ones, and starvation is hence avoided.
- Timeout-Based Schemes :
  - a transaction waits for a lock only for a specified amount of time. After that, the wait times out and the transaction is rolled back.
  - thus deadlocks are not possible
  - simple to implement; but starvation is possible. Also difficult to determine good value of the timeout interval.

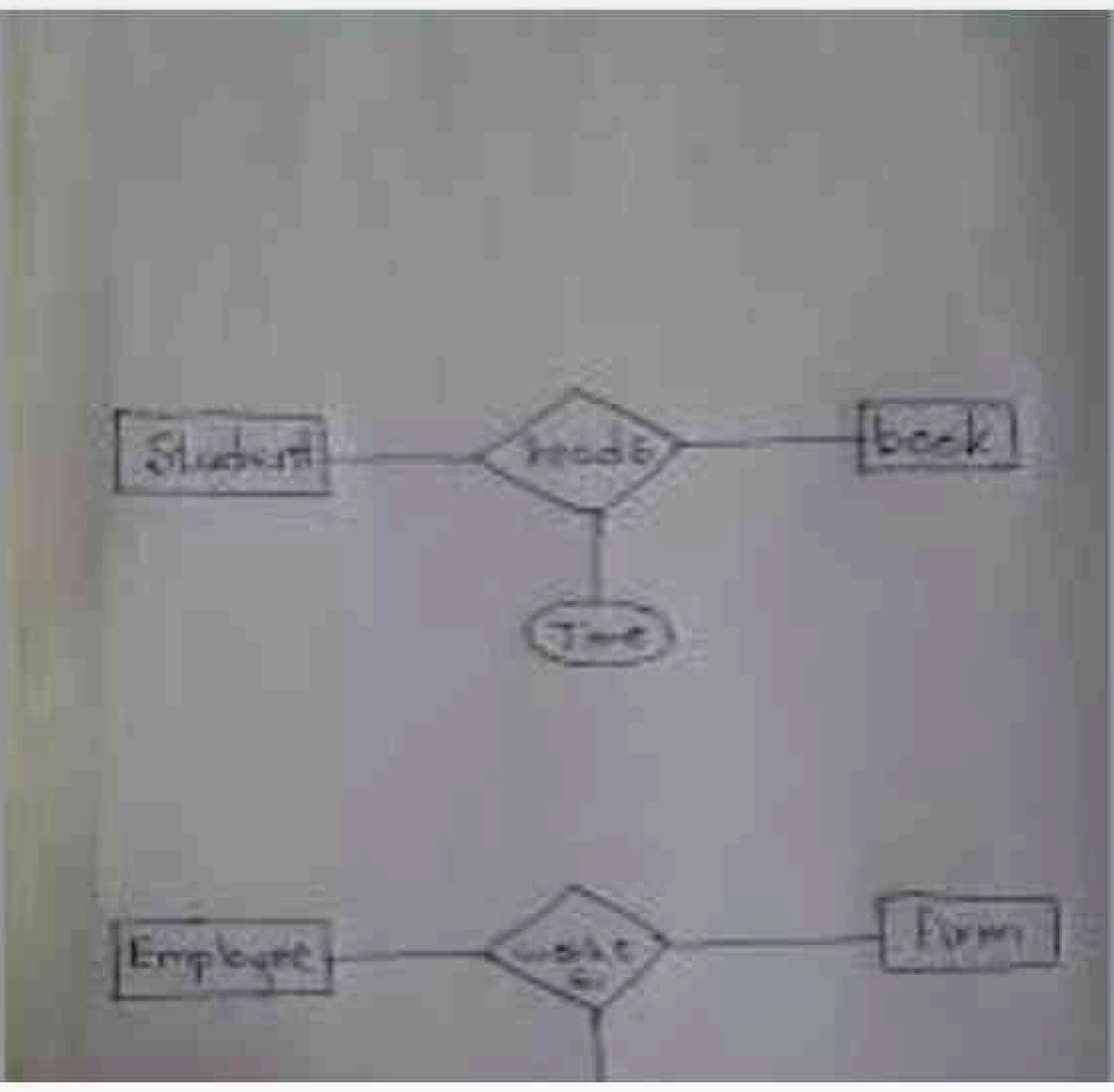
Deadlock Prevention (contd.)

- The Wait-Die Scheme:
  - A non-preemptive approach
  - When process Pi requests a resource currently held by Pj, Pi is allowed to wait only if it has a smaller timestamp than does Pj (Pi is older than Pj)
  - Otherwise Pi is rolled back
- The Wound-Wait scheme:
  - A preemptive approach
  - When process Pi requests a resource currently held by Pj, Pi is allowed to wait only if it has a larger timestamp than Pj (pi is younger)
  - Otherwise Pj is rolled back (pj is wounded)

Timestamp-Based Protocol
- The timestamp ordering protocol ensures that *any conflicting*

22:49

descriptive attribute in db...



বাংলায় In English

Attributes of the relationship is called descriptive attribute. For example, employee works for department. Here 'works for' is the relation between employee and department entities. The relation 'works for' can have attribute DATE_OF_JOIN which is a descriptive attribute. Aug 20, 2016

https://www.quora.com › What-is-d...

[What is descriptive attribute in ER model? - Quora]

About featured snippets Feedback

Discover Search Saved

5.

E_id → E_Name

P_id → Hour

P_id → P_Name

P_id → P_Location

R(ABCDEF)

$A \to B$

$C \to D$

$C \to E$

$C \to F$

$A \to B$

$C \to D$

$C \to EF$

× $A^+ = AB$

× $AB^+ = AB$

✓ $AC^+ = AC$

$= ACB$

$= ACBDEF$

× $BC^+ = BC$

CK = {AC}

PA = {A, C}

NPA = {B, D, E, F}

2NF: $A \xrightarrow[PD]{} B$, $C \xrightarrow[PD]{} D$, $C \xrightarrow[PD]{} EF$

$R_1(AB)$, $R_2(ACD)$, $R_3(CEF)$

E_id	E_Name

E_id	P_id	Hour

P_id	P_Name	P_Location

3NF:

$R_1(AB)$ $R_2(ACD)$ $R_3(CEF)$

$A \to B$

$A \to$

$C \to D$

$C \to EF$

$C \to E$

$C \to F$

$R_1(AB)$, $R_2(CD)$, $R_3(CE)$, $R_4(CF)$, $R_5(AC)$

E_id	E_Name

P_id	Hour

P_id	P_Name

P_id	P_Location

E_id	P_id

BCNF: Already BCNF.

1:06

Google

generalisation and speciali...

All Images Videos News Shopping Books

Examples | Pdf | Geeksforgeeks | Difference bet

বাঙলায় অনুসন্ধান করুন

ডিবিএমএসে সাধারণীকরণ এবং

বাংলায় | In English

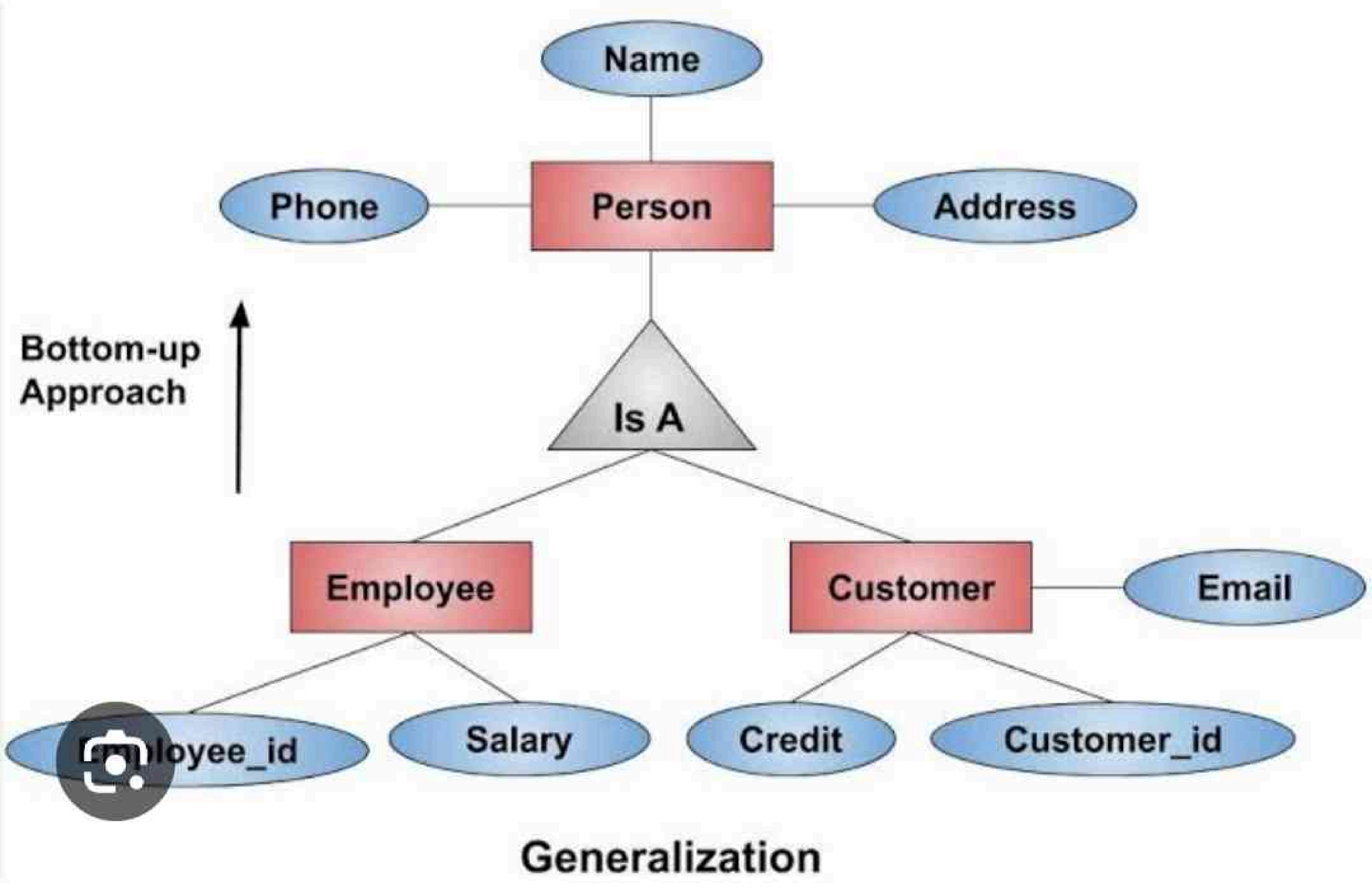
While Generalization uses a bottom-up approach and reduces the size of a schema, Specialization uses a top-down approach and increases the schema size. Generalization can be applied to multiple entities, while Specialization can only be applied to a single entity. Jul 31, 2023

https://testbook.com › key-differences

Difference Between Generalization and Specialization in DBMS | Testbook.com

About featured snippets | Feedback

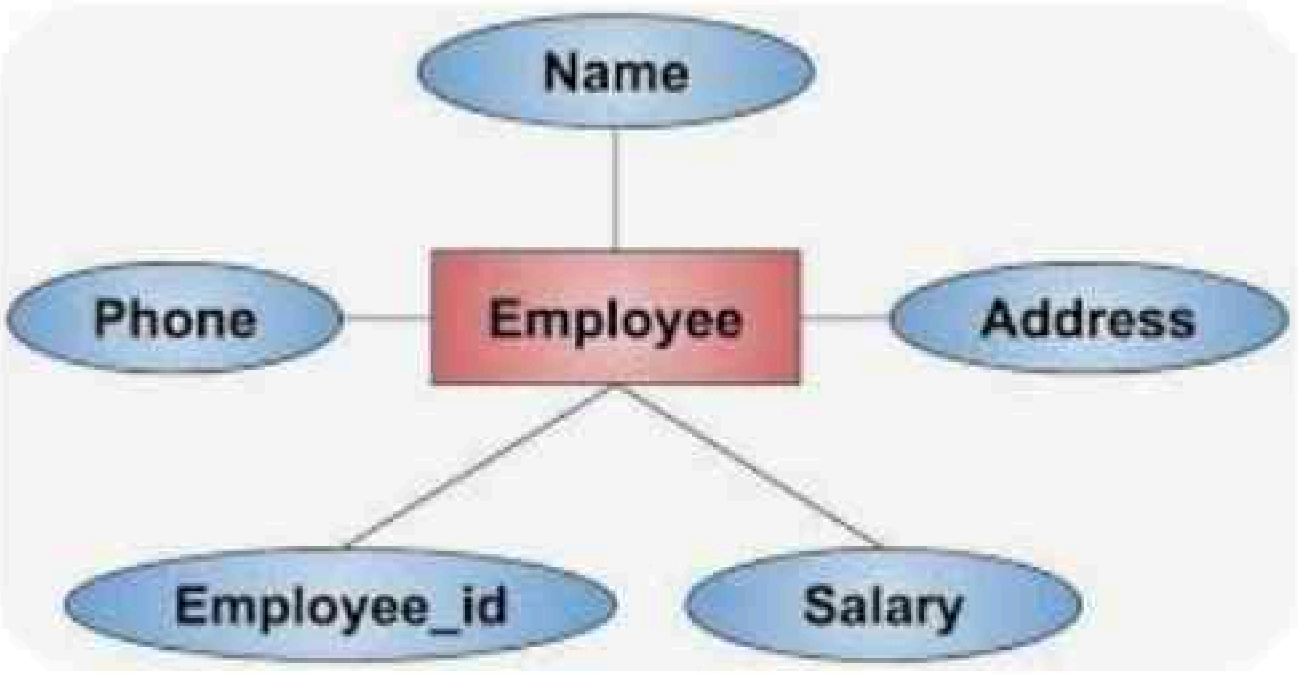
Discover Search Saved



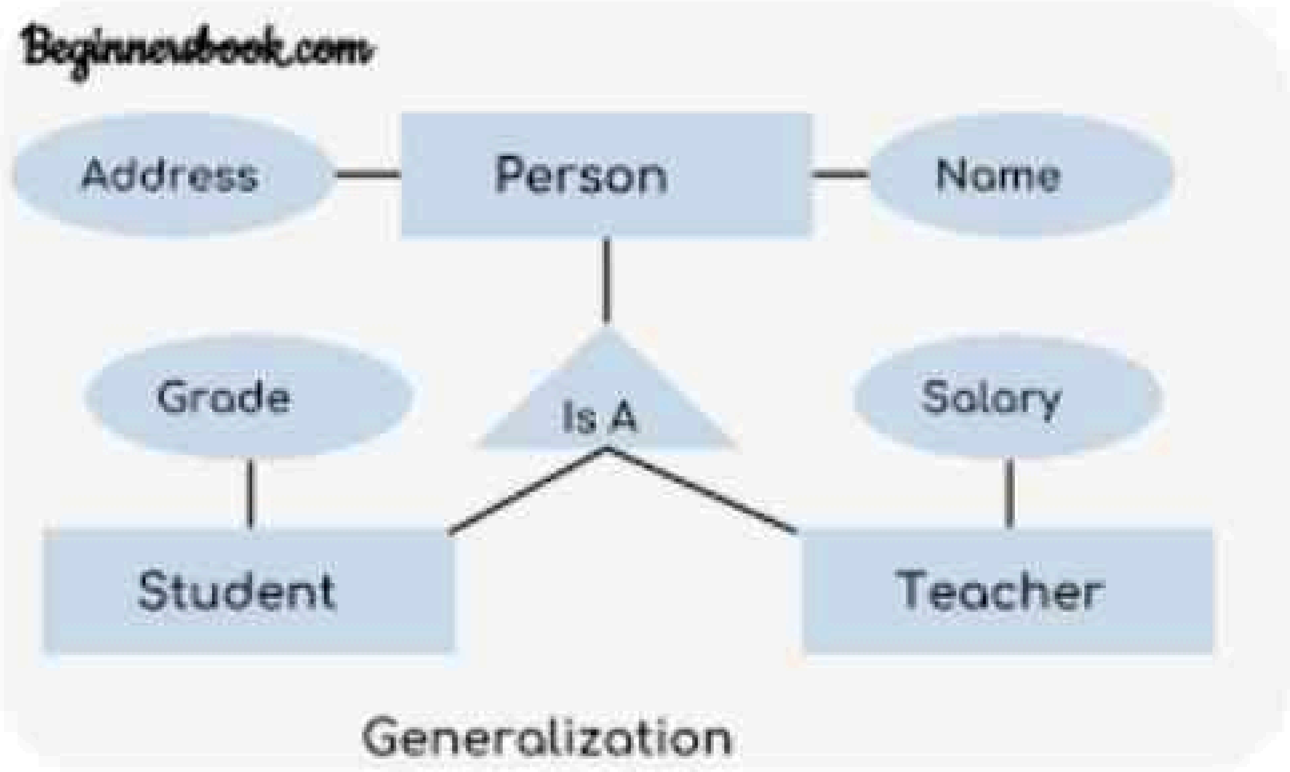
What is Generalization in DBMS?

Visit >

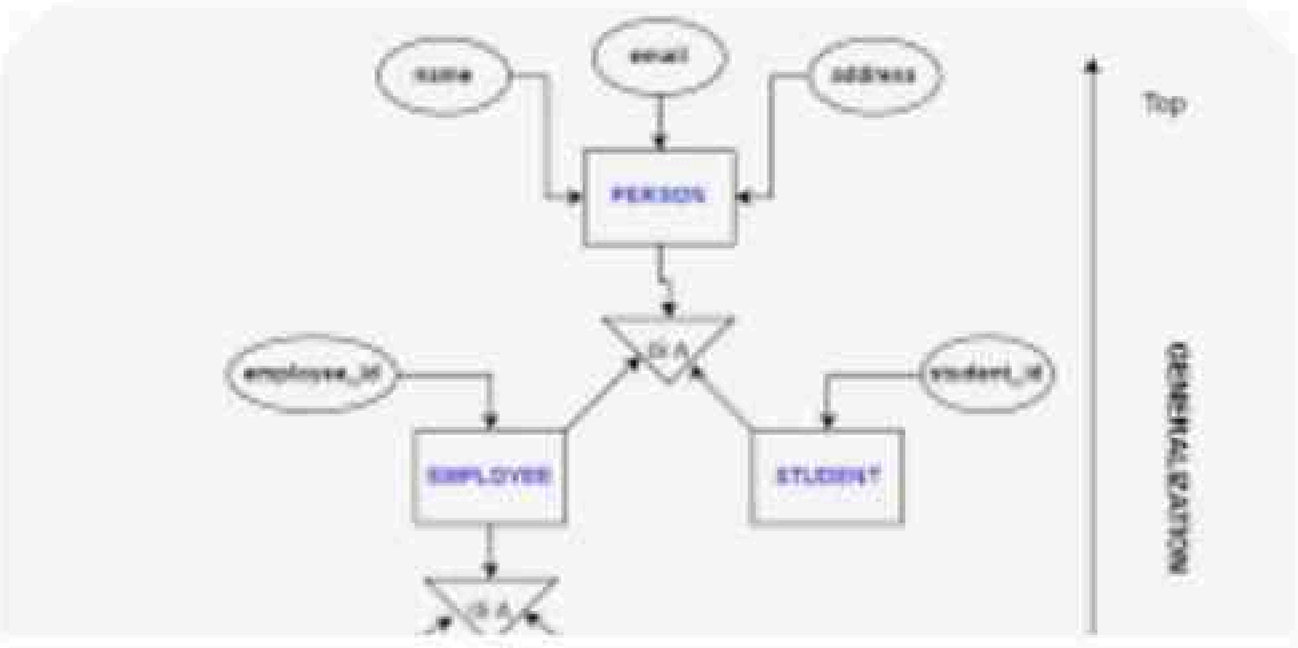
Share Save

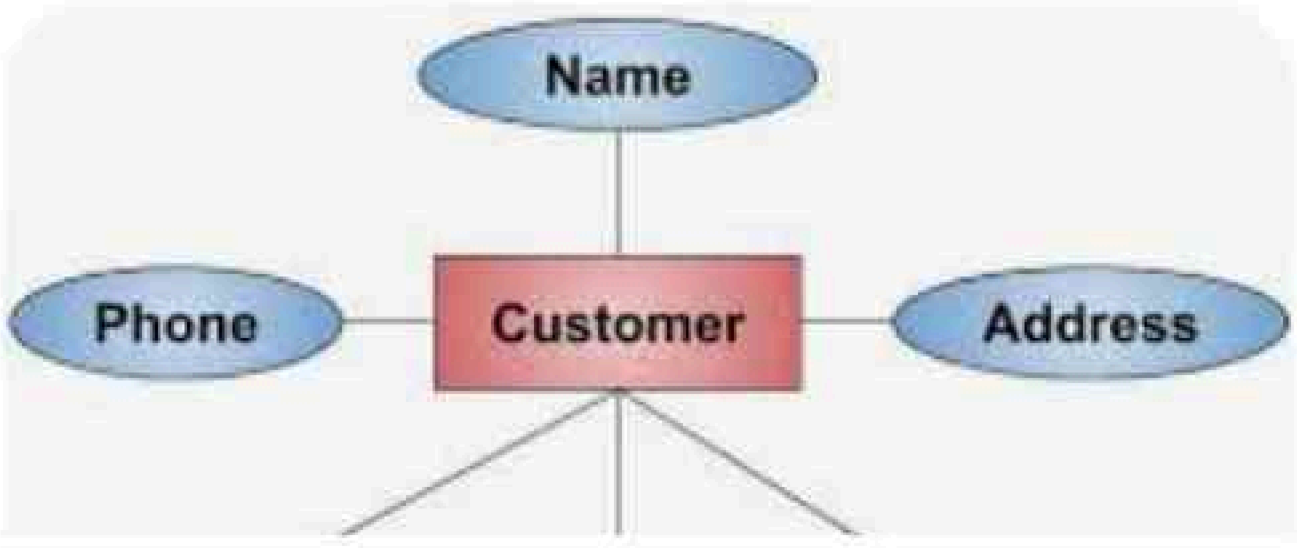


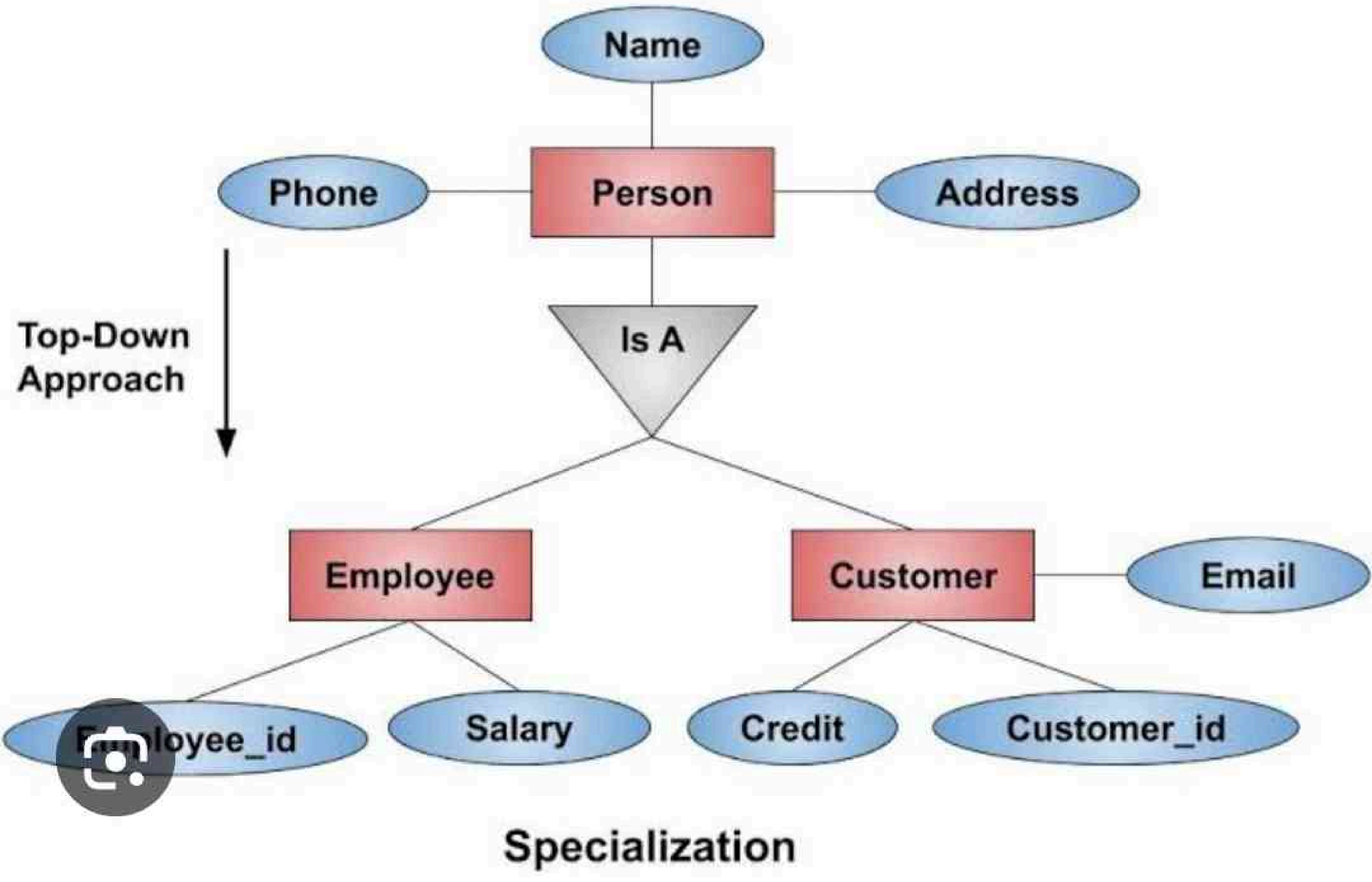
AfterAcademy
What is Generalization in ...



BeginnersBook
DBMS Generalization





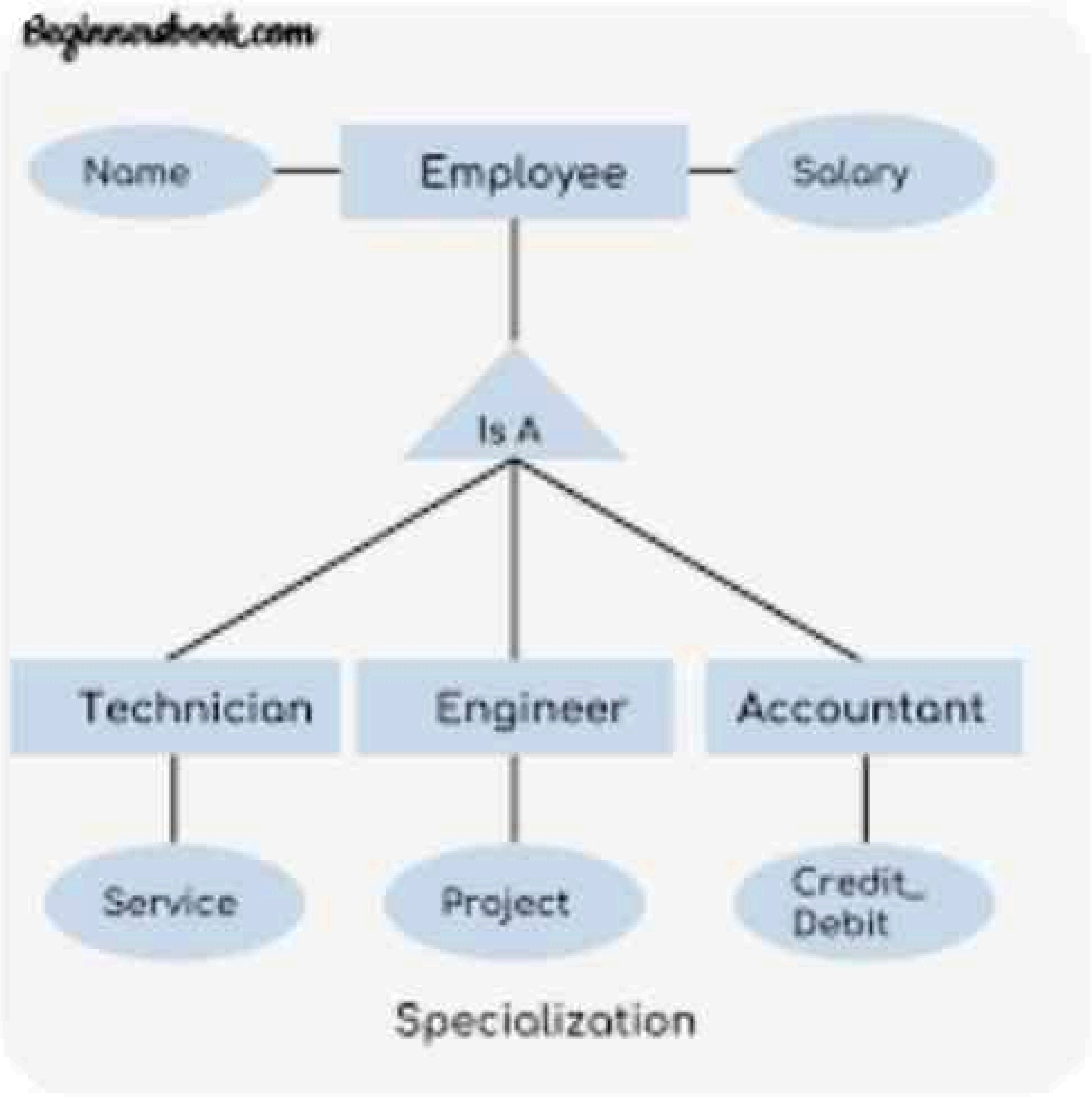


What is Specialization in DBMS?

Visit

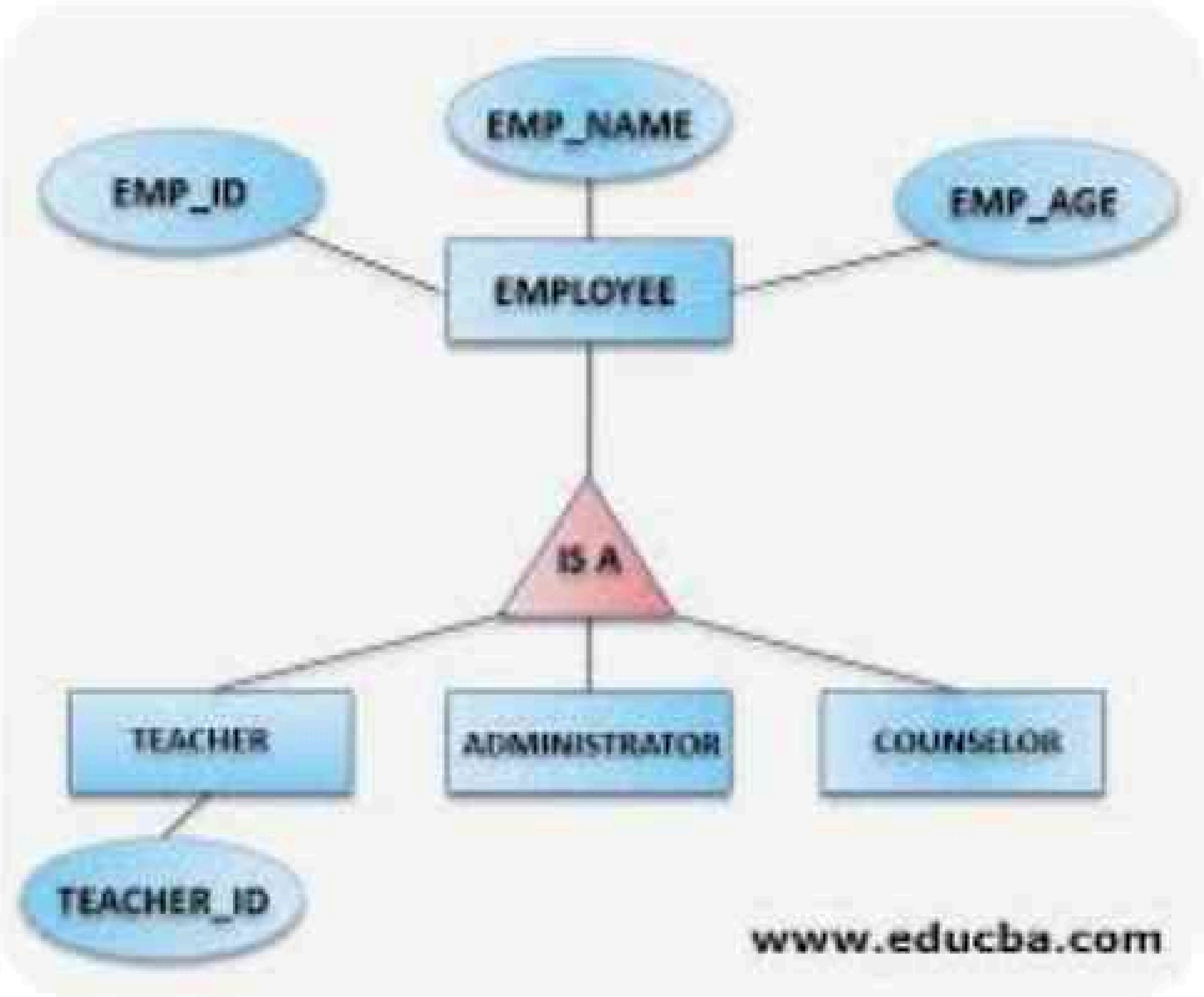
Share

Save



BeginnersBook

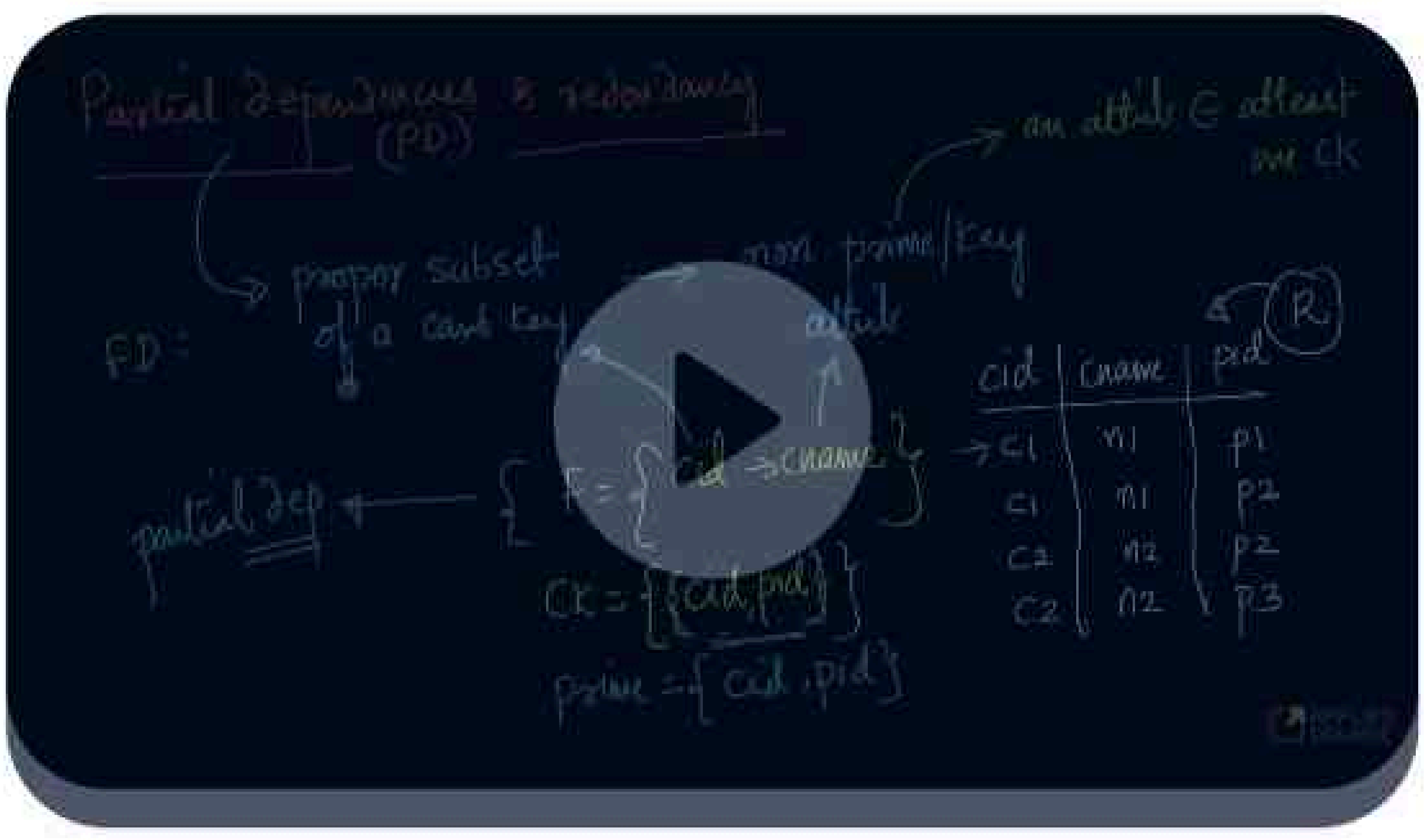
DBMS Specialization



EDUCBA

Specialization in DBMS | ...

having the same d
necessary to rem
causes anomalies
very hard for a da
maintain it.



Why Do We Need Normalization?

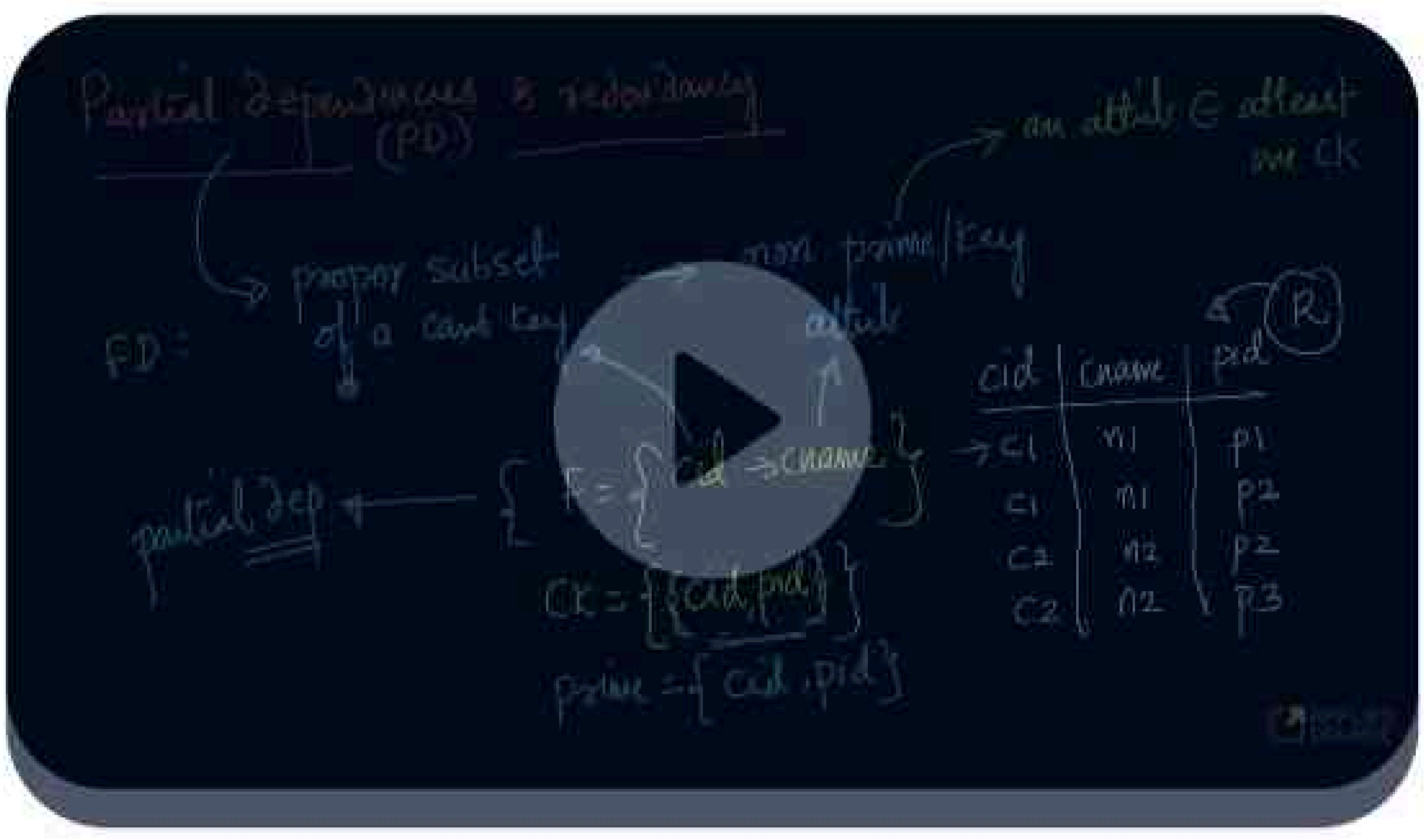
As we have discussed above, normalization is used to reduce data redundancy. It provides a method to remove the following anomalies from the database and bring it to a more consistent state:

A database anomaly is a flaw in the database that occurs because of poor planning and redundancy.

1. **Insertion anomalies**: This occurs when we are not able to insert data into a database because some attributes may be missing at the time of insertion.

2. **Updation anomalies:** This occurs when the same data items are repeated with the same values and are not linked to each other.

method to remov
the database and
state:



A database anom
occurs because of
redundancy.

1. **Insertion anomalies**: This occurs when we are not able to insert data into a database because some attributes may be missing at the time of insertion.

2. **Updation anomalies:** This occurs when the same data items are repeated with the same values and are not linked to each other.

3. **Deletion anomalies:** This occurs when deleting one part of the data deletes the other necessary information from the database.

In the next section, we'll see the types of normal forms in the database.

1:24

Google

closure in dbms

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন

dbms মধ্যে বন্ধ

বাংলায় In English

The set of all those attributes which can be functionally determined from an attribute set is called as a closure of that attribute set.

https://littleflowercollege.edu.in › ... PDF

Subject : DBMS Topic:Closure properties

About featured snippets Feedback

People also ask

What are closures in database?

What is closure of an attribute?

Discover Search Saved

Google

what is conflict in dbms

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন dbms মধ্যে দ্বন্দ্ব কি

বাংলায় In English

Conflict in DBMS can be defined as **two or more different transactions accessing the same variable and atleast one of them is a write operation**. For example: T1: Read(X) T2: Read (X) In this case there's no conflict because both transactions are performing just read operations.

Dec 11, 2012

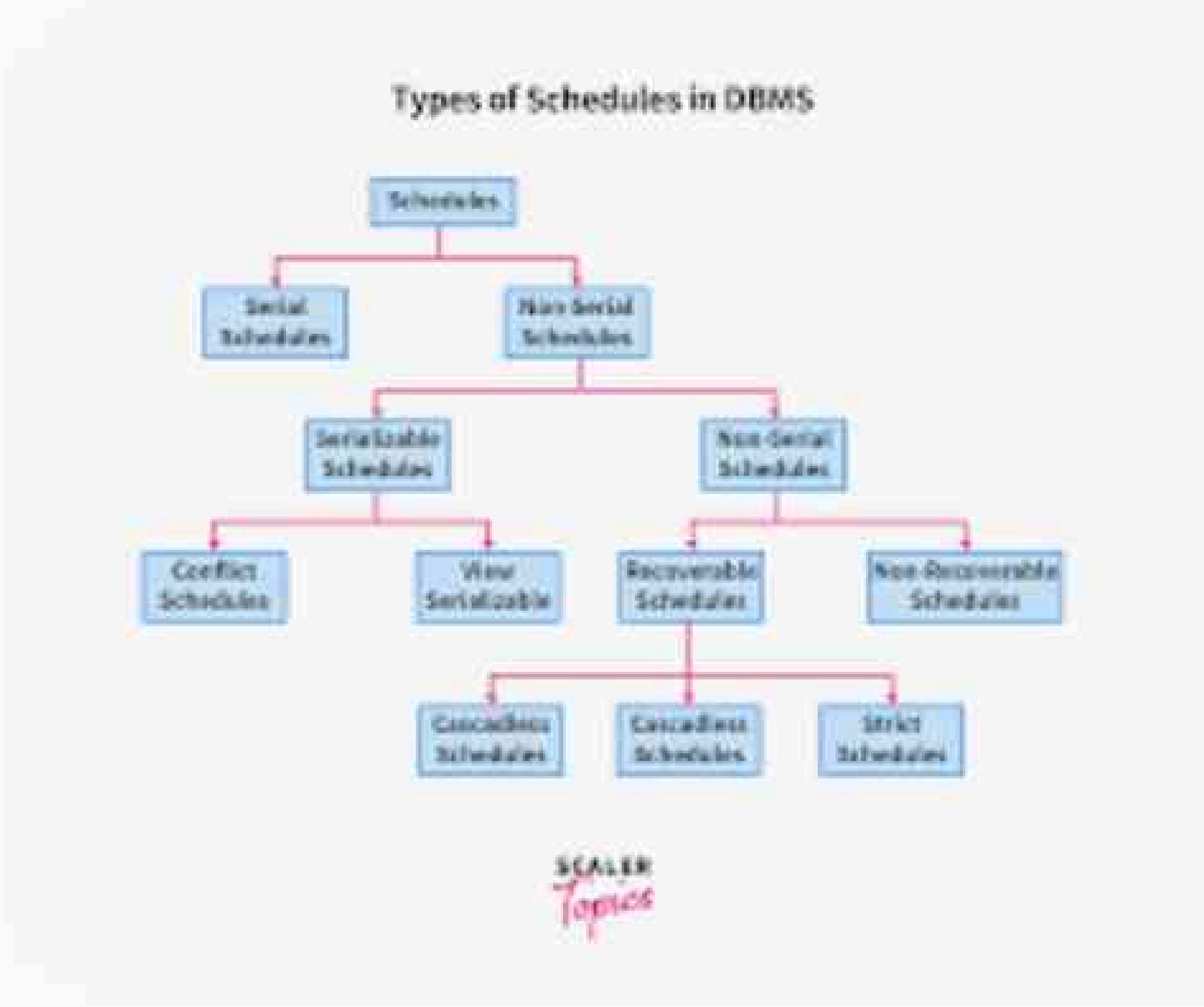
https://stackoverflow.com › questions

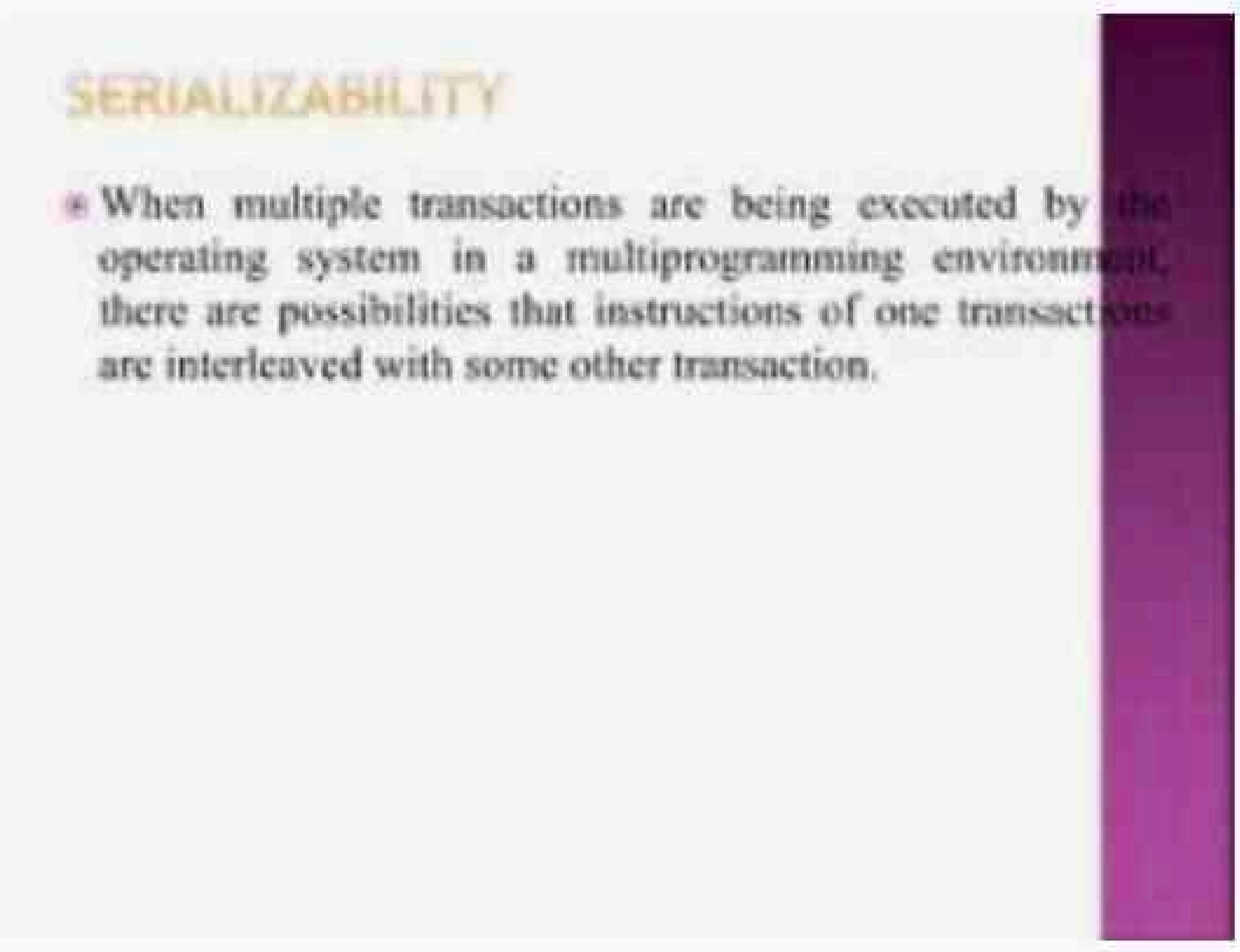
What is the difference between "conflict serializable" and "conflict equivalent"?

About featured snippets Feedback

3:04 96

serializabilty in dbms





বাংলায়   In English

Serializability in DBMS **guarantees that the execution of multiple transactions in parallel does not produce any unexpected or incorrect results**. This is accomplished by enforcing a set of rules that ensure that each transaction is executed as if it were the only transaction running in the system.

Feb 27, 2023

https://www.prepbytes.com › blog

Serializability in DBMS | Schedule in DBMS - PrepBytes

About featured snippets   Feedback

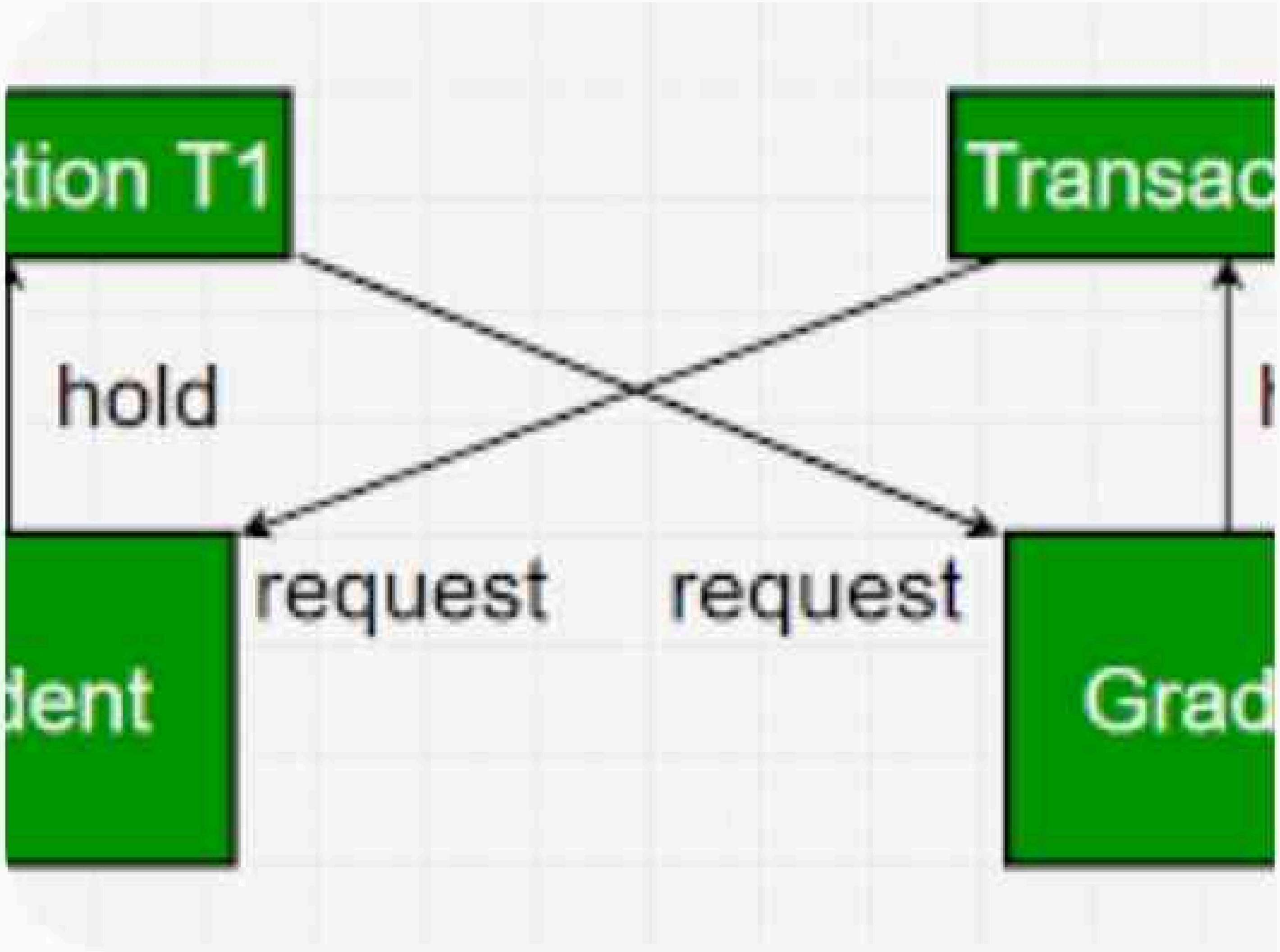
Discover   Search   Saved

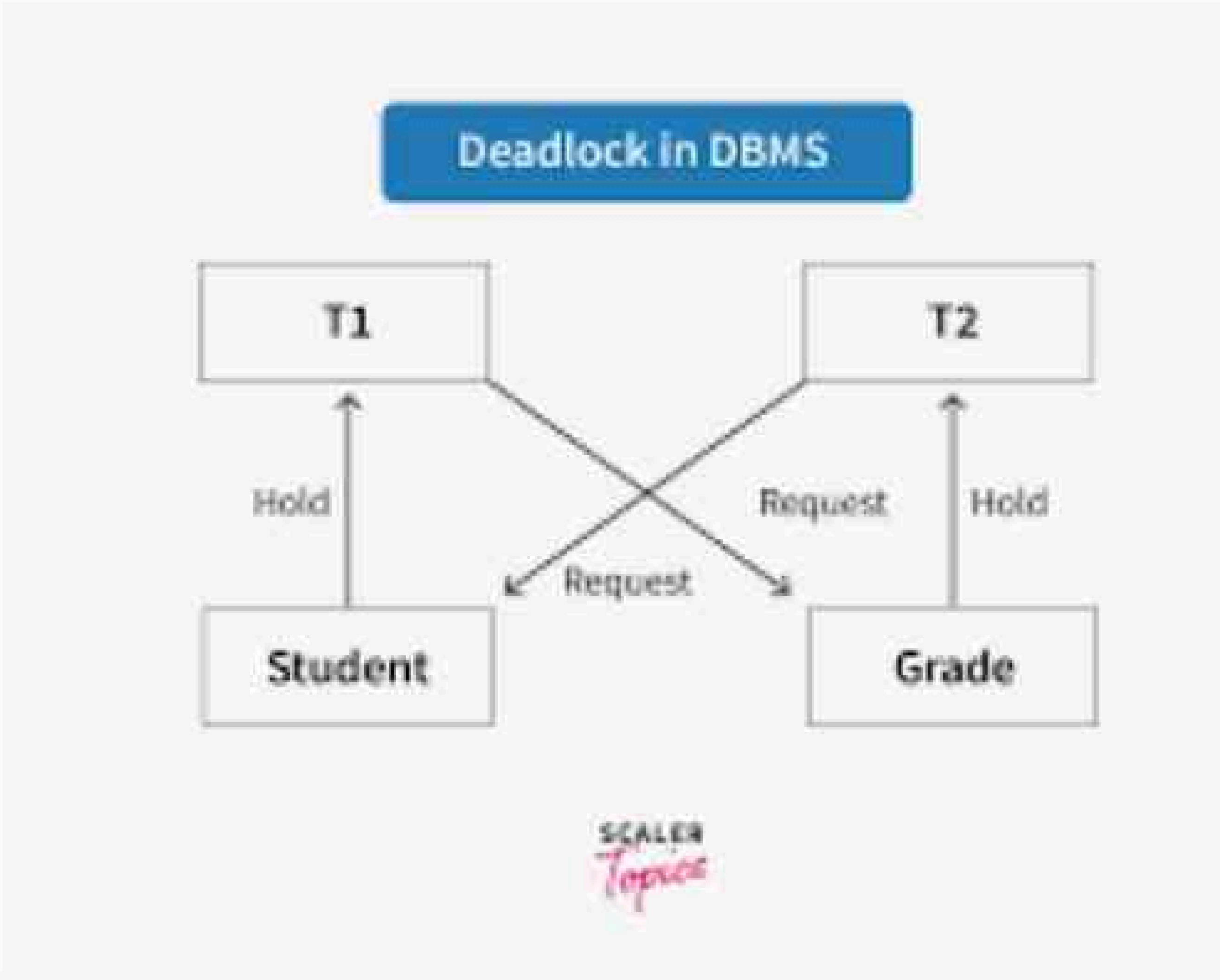
3:46 94

deadlock in dbms

বাঙলায় অনুসন্ধান করুন

ডিবিএমএসে অচলাবস্থা





বাংলায় In English

In a database management system (DBMS), a deadlock occurs when two or more transactions are waiting for each other to release resources, such as locks on database objects, that they need to complete their operations.

May 11, 2023

https://www.geeksforgeeks.org › de...

Deadlock in DBMS - GeeksforGeeks

About featured snippets Feedback

People also ask

Discover Search Saved

3:48

Google

starvation in dbms

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন dbms মধ্যে অনাহার

বাংলায় In English

Starvation happens when a transaction is repeatedly denied access to a data item because of other transactions holding locks on it. Both deadlock and starvation can affect the performance, reliability, and correctness of a DBMS.

Apr 3, 2023

https://www.linkedin.com › advice

Locking Techniques to Prevent Deadlock and Starvation in DBMS

About featured snippets Feedback

People also ask

What is deadlock and starvation in Java?

Discover Search Saved

3:50

Google

deadlock and starvation in...

All Images Videos News Shopping Books

বাঙলায় অনুসন্ধান করুন

ডিবিএমএসে অচলাবস্থা এবং অ.

বাংলায় In English

Deadlock happens when two or more transactions are waiting for each other to release a lock on a data item, and none of them can proceed. Starvation happens when a transaction is repeatedly denied access to a data item because of other transactions holding locks on it. Apr 3, 2023

https://www.linkedin.com › advice

Locking Techniques to Prevent Deadlock and Starvation in DBMS

About featured snippets Feedback

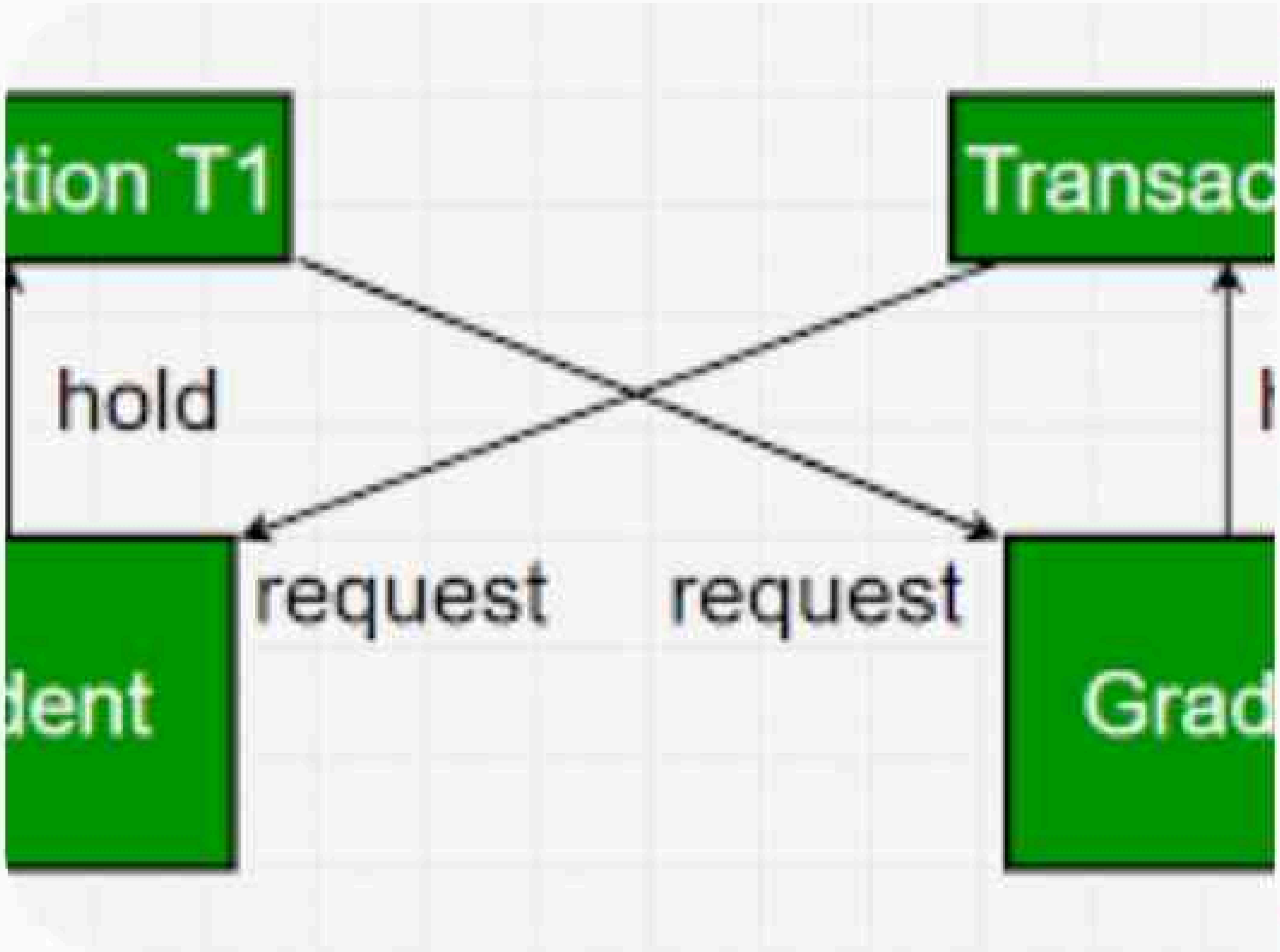
GeeksforGeeks

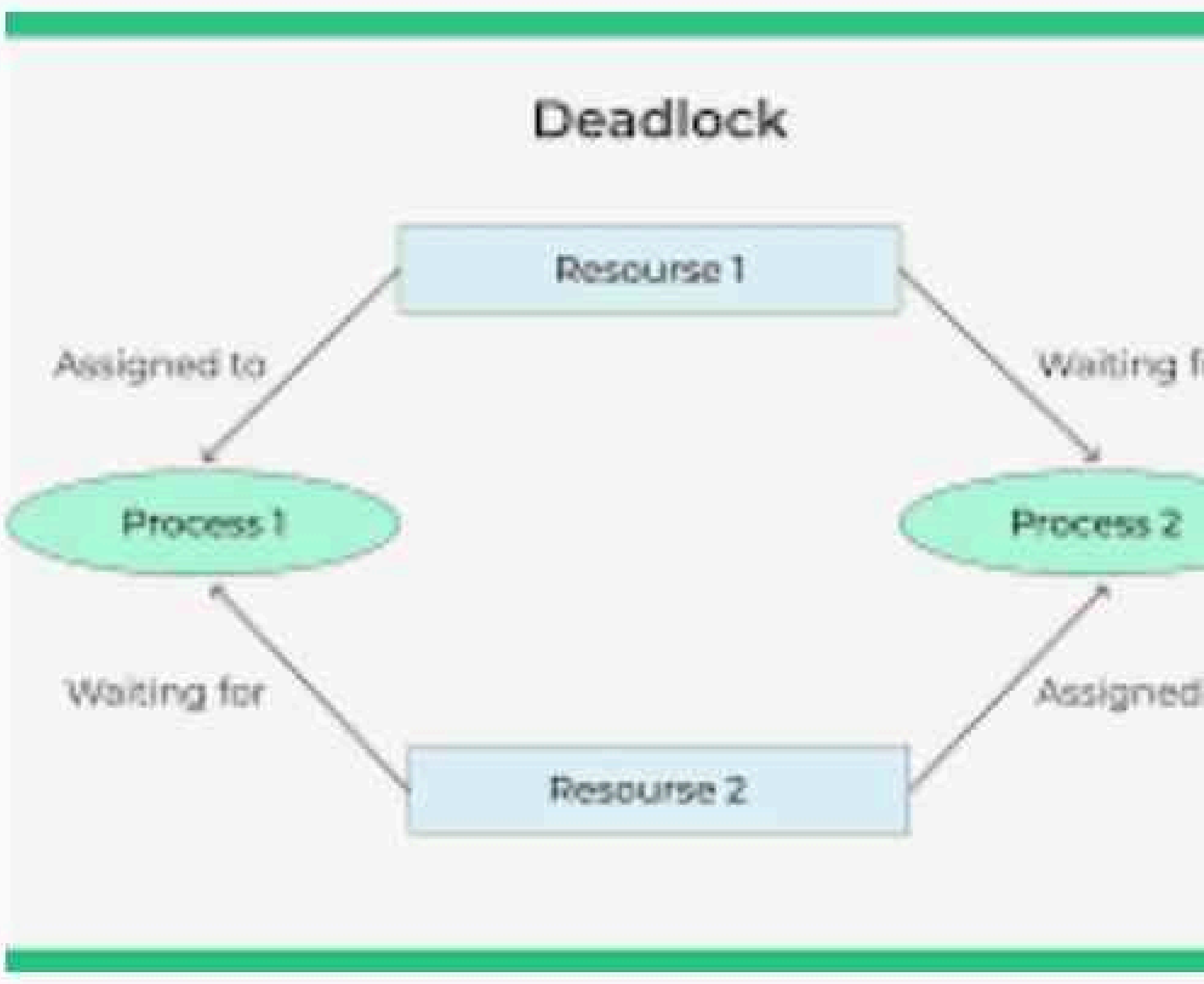
https://www.geeksforgeeks.org › sta...

Discover Search Saved

4:09

deadlock prevention in db...





বাংলায়   In English

There are two main methods for deadlock avoidance in DBMS are Resource Allocation Graph (RAG) Algorithm and Wait-For Graph Algorithms. Both of these approaches aim to ensure that there are no circular wait conditions in the system, which is the root cause of deadlocks. Feb 13, 2023

https://www.prepbytes.com › blog

Deadlock in DBMS - PrepBytes

About featured snippets   Feedback

GeeksforGeeks
https://www.geeksforgeeks.org › de...

Discover   Search   Saved

4:14

Google

deadlock detecton and rec...

All Videos Images Shopping News Books

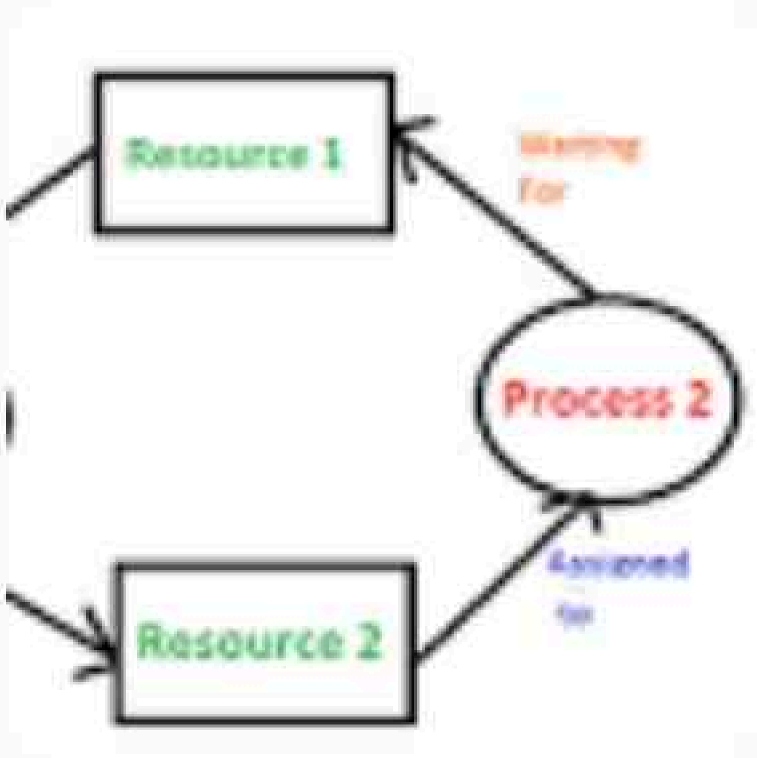
Showing results for deadlock ***detection*** and recovery in dbms

Search instead for deadlock detecton and recovery in dbms

বাঙলায় অনুসন্ধান করুন

ডিবিএমএসে অচলাবস্থা সনাক্ত...

বাংলায় In English



Deadlock detection algorithms, such as the Wait-For Graph, are used to identify deadlocks, and recovery algorithms, such as the Rollback and Abort algorithm, are used to resolve them. The recovery algorithm releases the resources held by one or more processes, allowing the system to continue to make progress. Mar 18, 2023

https://www.geeksforgeeks.org › de...

Discover Search Saved

progress.

There are two main approaches to deadlock detection and recovery:

1. **Prevention**: The operating system takes steps to prevent deadlocks from occurring by ensuring that the system is always in a safe state, where deadlocks cannot occur. This is achieved through resource allocation algorithms such as the Banker's Algorithm.
2. **Detection and Recovery:** If deadlocks do occur, the operating system must detect and resolve them. Deadlock detection algorithms, such as the Wait-For Graph, are used to identify deadlocks, and recovery algorithms, such as the Rollback and Abort algorithm, are used to resolve them. The recovery algorithm releases the resources held by one or more processes, allowing the system to continue to make progress.

Difference Between Prevention and Detection/Recovery: Prevention aims to avoid deadlocks altogether by carefully managing

Deadlock in DBMS

May 11, 2023 — The deadlock avoidance method is suitable for smaller databases whereas the deadlock prevention method is suitable for larger databases. One ...

People also ask

What are the disadvantages of deadlock prevention?

Disadvantages. Deadlock prevention can be difficult to implement as it requires careful consideration of resource allocation and scheduling policies to avoid circular dependencies between processes. Apr 6, 2023

https://www.tutorialspoint.com › dif...

Difference between Deadlock Prevention and Deadlock Avoidance

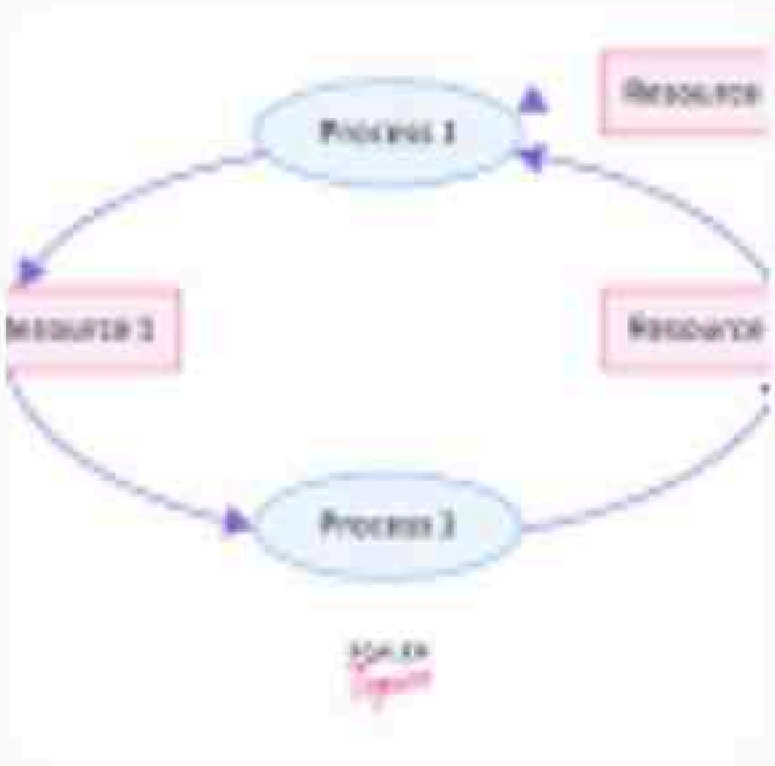
MORE RESULTS

What are the disadvantages of deadlock in DBMS?

What are the 4 ways to prevent deadlock?

What are the 4 ways to prevent deadlock?

Deadlock can be prevented by eliminating any of the four necessary conditions, which are **mutual exclusion, hold and wait, no preemption, and circular wait**.



https://www.scaler.com › topics › d...

Deadlock Prevention in OS (Operating System) - Scaler Topics

MORE RESULTS

How can we prevent deadlocks in DBMS?

There are two main methods for deadlock avoidance in DBMS are **Resource Allocation Graph (RAG) Algorithm and Wait-For Graph Algorithms**. Both of these approaches aim to ensure that there are no circular wait conditions in the system, which is the root cause of deadlocks. Feb 13, 2023