Chapter-6 Security

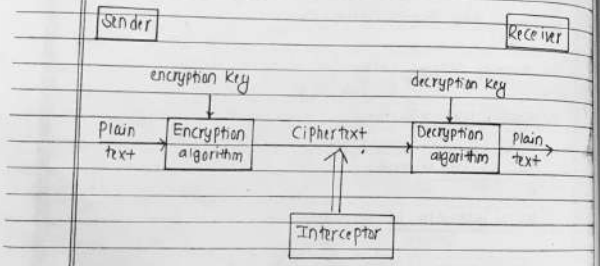
2011 spring

4a) What is cryptography? Explain the types of cryptographic system

- A cryptography is an implementation of cryptographic techniques and their accompanying infrastructure to provide information security services. A cryptosystem is also referred as a cipher system.

- Let us discuss a simple model of a cryptosystem that provides confidentiality to the information being transmitted.

10120181 Page 11



⇒ Components of a cryptosystem

- Plaintext :- It is the data to be protected during transmission

- Encryption Algorithm :- It is a mathematical process that produces a chipertext for any given plaintext and

240

10120181 Page 12

encryption key. It is a cryptographic algorithm that takes plaintext and an encryption key as input and produces a chipertext.

- Ciphertext :- It is the scrambled version of the plaintext produced by the encryption algorithm using a specific encryption key. The ciphertext is not guarded. It flows on public channel. It can be intercepted or compromised by anyone who has access to the communication channel.

- Decryption Algorithm :- It is a mathematical process, that produces a unique plaintext for any given ciphertext and decryption key. It is a cryptographic algorithm that takes a ciphertext and a decryption key as input, and outputs a plaintext. The decryption algorithm essentially reverses the encryption algorithm and is thus closely related to it.

- Encryption Key :- It is a value that is known to the sender. The sender inputs the encryption key into the encryption algorithm along with the plaintext in order to compute the ciphertext.

- Decryption Key :- It is a value that is known to the receiver. The decryption key is related to the encryption key, but is not identical to it. The receiver inputs the decryption key into the decryption algorithm along with the ciphertext in order to compute the plaintext.

For a given cryptosystem, a collection of all possible decryption keys is called a Key space. An interceptor (an attacker) is an unauthorized entity who attempts to determine the plaintext. He can see the ciphertext and may know the decryption algorithm. This person, however must not know the decryption key.

241

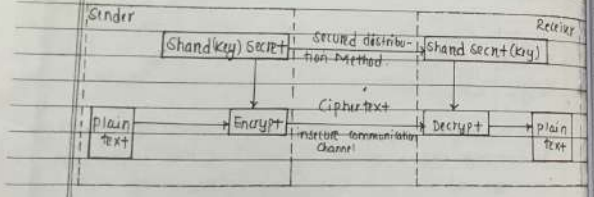
⇒ Types of cryptosystem

- Fundamentally, there are two types of cryptosystems based on the manner in which encryption-decryption is carried out in the system.

i) Symmetric Key Encryption

- The study of symmetric cryptosystems is referred to as symmetric cryptography. The encryption process where same keys are used for encrypting and decrypting the information is known as symmetric key Encryption.

- A few well-known examples of symmetric key encryption methods are :- AES (Advance Encryption Standard), DES (Data Encryption Standard), IDEA and BLOWFISH. The process of depicted in following illustration



⇒ The salient features of cryptosystem based on symmetric key encryption are :-

- Person using this must share a common key prior to exchange of information.

- Keys are recommended to be changed regularly to prevent any attack on the system

- A robust mechanism needs to exist to exchange the key between the communicating parties. As keys are regularly, this mechanism becomes expensive and cumbersome.

- In a group of n people, to enable two-party communication between any two persons, the number of keys required for group is $n \times (n-1)/2$.

- Length of key (number of bits) in this encryption is smaller and hence, process of encryption-decryption is faster than asymmetric key encryption.

- Processing power of computer system required to run symmetric algorithm is less.

* Challenges of symmetric key cryptosystem are key establishment and Trust issue.

ii) Asymmetric Key Encryption

- The encryption process where different keys are used for encrypting and decrypting the information is known as Asymmetric Key Encryption.

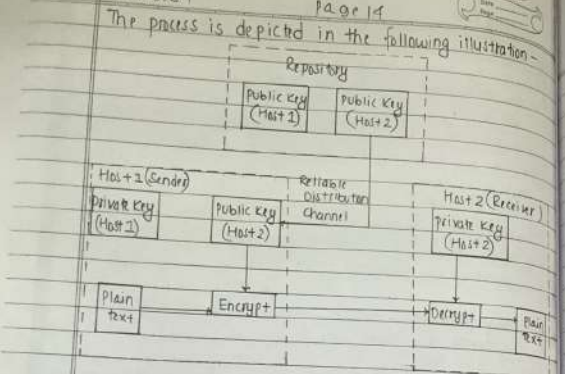
- Two keys private and public - public key are published. Private key is known to only to user to whom key belongs.

- If user1 wants to send encrypted data, user2 encrypts them using public key E1, decryption requires a private key D1.

- eg :- RSA, DSA, ECC, El Gamal etc.

10/12/2021 page 14

The process is depicted in the following illustration -



The silent features of this encryption

- Every user in this system needs to have a pair of dissimilar keys private key and public key. These keys are mathematically related - when one key is used for encryption, the other can decrypt the ciphertext back to the original plaintext.
- It requires to put the public key in public repository and the private key as a well-guarded secret. Hence, this schema of encryption is also called public key Encryption.
- Through public and private keys of the user are related, it is computationally not feasible to find one from another. This is a strength of the schema.
- When Host1 needs to send data to Host2, person obtains the public key of Host2 from repository, encrypts data and transmits

- Host2 uses his private key to extract the plaintext
- Length of keys (number of bits) in this ecryption is large and hence, the process of encryption and decription is slower than symmetric key encryption.
- processing power of computer system required to run asymmetric algorithm is higher.

2012-Fall

4a] It is necessary to manage security at OS level if security in database level is already done? Explain private key encryption.

- Yes, it is necessary to manage security at os level if security in database level is already done

⇒ Database security levels [For 3 marks]

- To ensure db security, security at different levels should be maintained
- A weakness at low level of security allows strict high level security measures.
- Step I:- Database security system
  Database system has to ensure that authorization restrictions are not violated.
- Step II:- operating security system
  - weakness in OS security is concerned with unauthorized access to db.

- Step III :- Network security
  - Security software level within network system is important

- Step IV :- Physical security
  - Sites with computer system must be physically secured against armed entry by intruders.

- Step V :- Human security
  - user must be authorized carefully.

2012-spring

4b) Explain the Major issue related to Database security.

The Major issue related to Database security are :-

i) Deployment failures :- The most common cause of database issue is a lack of due care at the moment they are deployed. Although any given database is tested for functionality and to make sure it is doing what the databases is designed to do, very few checks are made to check the database is not doing things it should not be doing.

ii) Data leaks :- Database may be considered a "back-end" part of the office and secure for internet-based threats, but this is not the case. Databases also contain a networking interfaces, and so hackers are able to capture this type of traffic to exploit it. To avoid such a pitfall, administrators should use SSL-or TLS-encrypted communication platform.

iii) A lack of segregation :- The separation of administrative and user powers, as well as the segregation of duties, can make it more difficult for fraud or theft undertaken by internal staff. In addition, limiting the power of user accounts may give a hacker a harder time in taking complete control of a database.

iv) Stolen database backups :- External attackers who infiltrate systems to steal data are one threat, but what about those inside the corporation? The report suggests that insiders are also likely to steal archives - including database backups - whether for money, profit or revenge. This is the common problem for the modern enterprise, and business should consider encrypting archives to mitigate the insiders risks.

v) Sql injections :- A popular method for hacker to take, sql injections remains a critical problem in the protection of enterprise databases. Applications are attacked by injections, and the database administrator is left to clean up the mess caused by unclean variables and malicious code which is inserted into strings passed to an instance of sql server for parsing and execution. The best ways to protect against these threats are to protect web-facing databases with firewalls and to sqit input variables for sql injection during development.

vi) Hopscotch :- Rather than taking advantage of buffer overflows and gaining complete access to a database in the first stage, cybercriminals often play a game of hopscotch, finding a weakness within the infrastructure that can be used as leverage for more serious attacks until they reach the

vii) Access to key fields

- Suppose we have a user role with access rights to table A and to table C but not to table B. The problem is that the foreign key in C includes columns from B. The following questions arise:

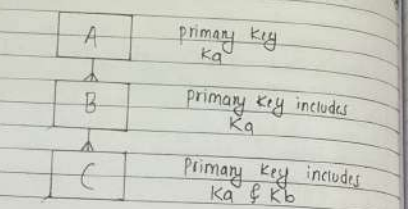


Fig 1.1

- Do you have the access to the foreign key in C?
- If you do, you know at least that a tuple exists in B and you know such information about B that is restricted from you.
- Can you update the foreign key columns?
- If so, it must cascade, generating an update to B for which no privileges have been given.
- These problems do not directly arise where the database is implemented by internal pointers - as a user we need to have knowledge of the relationships between the data we are accessing. These arise in relationships on data values.

often knowing the foreign key will not be sensitive in itself. If it is, then the declaration of a view may solve the problem.

2013 Fall

4a) What is integrity violation? Discuss the security levels that can be applied to dbms.

- Integrity violations occur when an insert, update or delete statement violates a primary key, foreign key, check or unique constraint or a unique index.
- There are mainly three operations that have the ability to change the state of relations they are insert, delete and update.
- Whenever we apply the above modification to the relation in the database, the constraints on the relational database should not get violated.

Insert operation: on inserting the tuples in the relation, it may cause violation of the constraints in the following ways

1) Domain constraint: Domain constraint get violated only when a given value to the attribute does not appear in the corresponding domain or in case it is not of the appropriate datatype.

eg:- Assume that the domain constraint says that all the values we insert in the relation should be greater than 10, and in case we insert a value less than 10 will cause the violation of the domain constraint, so gets rejected.

iii) Entity Integrity constraint:

- on inserting NULL values to any part of the PK of a new tuple in the relation can cause voilation of the Entity integrity constraint.

eg:-

Insert (NULL, 'Bikash' '1854') into EMP

Null value in the schema so, it voilates the Entity Integrity constraint that PK can't be null.

iii) Key constraints:

- on inserting a value in the new tuple of a relation which is already existing in another tuple of the same relation can cause voilation of key constraints.

iv) Referential Integrity

- on inserting a value in the F.K of relation 1, for which there is no corresponding value in the PK which is referred to in relation 2, in such case referential integrity is voilated.

eg:-

When we try to insert a value say 1200 in EID(FK) of table 1, for which there is no corresponding EID (primary key) of table 2, then it causes the voilation, so gets rejected.

Database Security levels [for 3 marks]

- To protect the database, we must take security measures at several levels.

To ensure db security, the security at different levels should be maintained. A weakness at low level of security allows strict high level security measures.

Various security levels are:-

i) Physical:

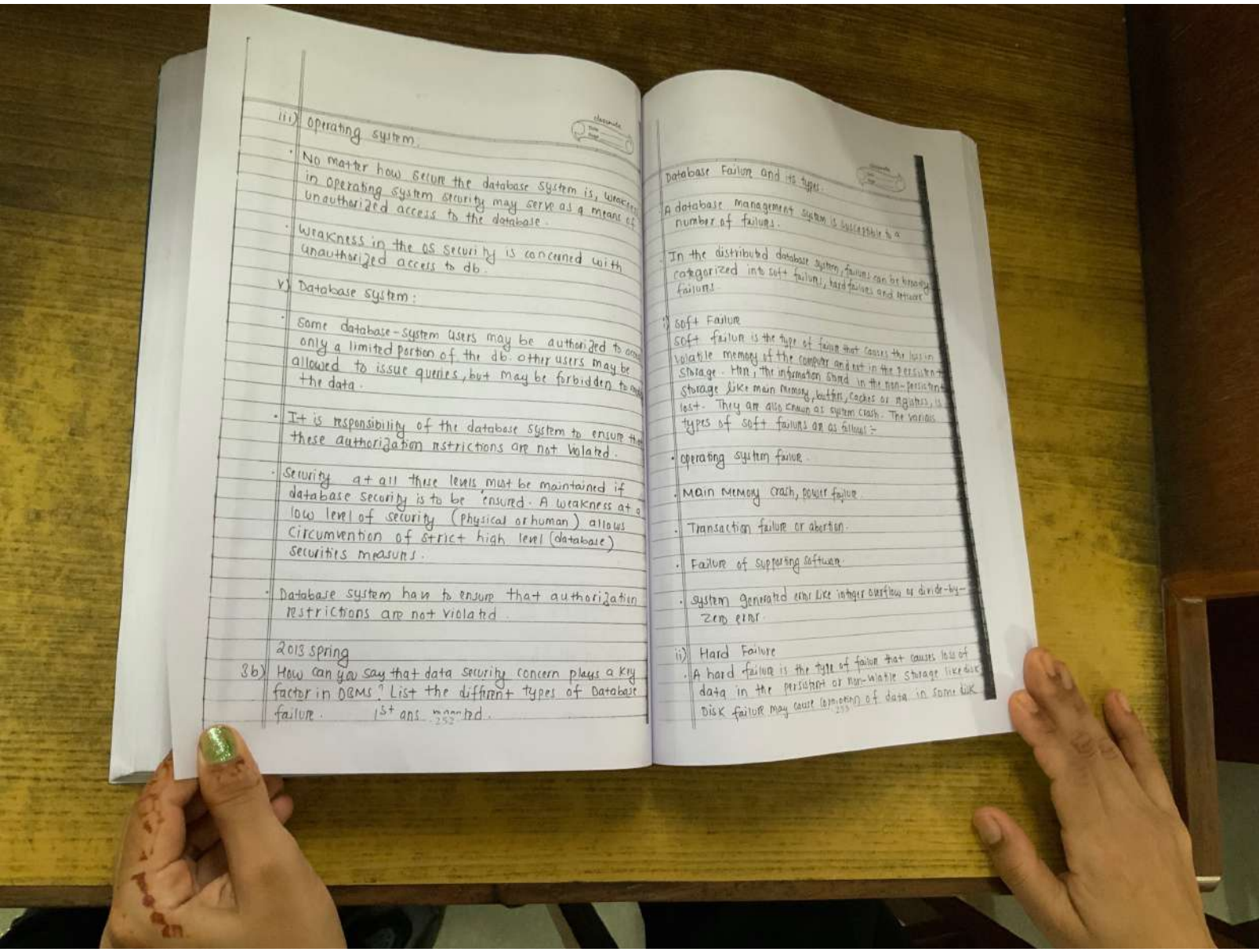
- The sites containing the computer systems must be secured against armed or surreptitious entry by intruders.

ii) Human:

- users must be authorized carefully to reduce the chance of any such user giving access to an intruder in exchange for a bribe or other favors.

iv) Network:

- Since almost all database systems allow remote access through terminals or networks,
- software level security within the network software is as important as physical security.
- Both on the internet and in networks private to an enterprise.
- This Network system security is very important



ii) Operating system:

- No matter how secure the database system is, weakness in operating system security may serve as a means of unauthorized access to the database.
- Weakness in the OS security is concerned with unauthorized access to db.

v) Database system:

- Some database-system users may be authorized to access only a limited portion of the db. other users may be allowed to issue queries, but may be forbidden to modify the data.
- It is responsibility of the database system to ensure that these authorization restrictions are not violated.
- Security at all these levels must be maintained if database security is to be ensured. A weakness at a low level of security (physical or human) allows circumvention of strict high level (database) securities measures.
- Database system have to ensure that authorization restrictions are not violated.

2013 spring

3b) How can you say that data security concern plays a key factor in DBMS? List the different types of Database failure. 1st ans marked.

Database Failure and its types:

A database management system is susceptible to a number of failures.

In the distributed database system, failures can be broadly categorized into soft failure, hard failures and byzantine failures.

i) Soft Failure

Soft failure is the type of failure that causes the loss in volatile memory of the computer and not in the persistent storage. Then, the information stored in the non-persistent storage like main memory, buffer, caches or registers, is lost. They are also known as system crash. The various types of soft failures are as follows:-

- operating system failure
- Main memory crash, power failure
- Transaction failure or abortion.
- Failure of supporting software
- System generated error like integer overflow or divide-by-zero error.

ii) Hard Failure

- A hard failure is the type of failure that causes loss of data in the persistent or non-volatile storage like disk. Disk failure may cause destruction of data in some disk.

blocks or failure of the total disk.
The causes of a hard failure are :-

- power failure, Faults in media.
- Read-write malfunction.
- corruption of information on the disk.
- Read/write head crash of disk.

Recover from disk failures can be short, if there is one formatted and ready-to-use disk on reserve. Otherwise, duration includes the time it takes to get a purchase order, buy the disk and prepare it.

iii) Network Failure

Network Failures are prevalent in distributed or network databases. These comprises of the errors induced in the db system due to the distributed nature of the data and transferring the data over the network.
The causes of network failure are as follow :-

- communication link failure.
- Network congestion.
- Information corruption during transfer.
- Site failures.
- Network partitioning.

2019 Fall

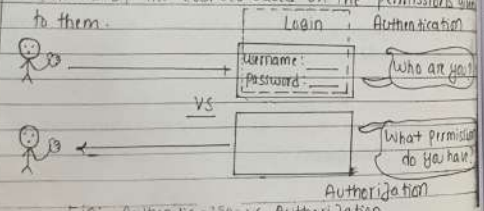
Differentiate between authentication and authorization. How encryption & decryption occurs in private key & public key cryptography?

Authentication :-

- It is the process of confirming the truth of an attribute of a single piece of data claimed true by an entity.
- checks a person's details to identify him. It verifies user's credentials.
- occurs before authorization and is the process of verifying the identity of a user.
- It determines whether user is what he/she claims to be. Authentication usually requires a username and a password.
- Types of Authentication: Windows authentication, Forms authentication and passport authentication.
- Methods are :- a) login form b) HTTP authentication c) HTTP digest d) X.509 certificates e) custom authentication methods.
- eg :-
A Student can authenticate himself before accessing the learning management system of a university.

Authorization

- It is a process of specifying access rights/privileges to resources related to the information security.
- Checks a user's privilages to resources related to access resources. validates user's permission.
- occours after authentication and it is the process of checking whether the user has access rights to the system.
- It determines what user can and cannot access and authentication factors required for authorization may vary, dependent on the security level
- Types of Authorization: ACL authorization (also known as file authorization), URL authorization
- methods are :- a) access controls for URLs b) secure objects and methods c) Access controls lists (ACLs)
- eg:- He/she can access lecture slides and other learning material of the courses based on the permissions given to them.



Eg:- Authentication vs Authorization

2019 spring

5a) What are the needs of cryptography? How security can be granted using Virtu? Explain.

Needs and Importance of cryptography and advantage

It secures information and communications using a set of rules that allows only those intended and no one else to receive the information to access and process it

- Cryptography enables secure communications because we live in the age of information and there is distrust in this world.
- Military applications: command and control
- Diplomatic applications: information gathering, spy
- Economic application: Information protection, Banking transaction, Authentication and signatures, modern E-commerce, pay TV, mobile phones
- Cryptography can provide the basic functions such as
  i) confidentiality:- ensures only authorized parties can view it.
  ii) Integrity:- Ensures information is correct and unaltered.
  iii) Authentication:- Ensures sender can be verified through cryptography.

iv) Non-repudiation: proves that a user performed an action

v) obfuscation: making something obscure or unclear.

- Security through obscurity

- An approach in security where virtually any system can be made secure as long as outsiders and unaware of its how it functions.

Security granted using view

- Through a view, users can query and modify only the data they can see. The rest of the database is neither visible nor accessible.

- permission to access the view must be explicitly granted or revoked, regardless of the permissions on the virtual underlying tables.

- This could lead rad degradation in query performance at execution. views can be an important security mechanism for a DBMS views are virtual tables derived from one more stored base tables, using the relational operators and/or statical summary.

- The view mechanism provides a powerful and flexible security mechanism by hiding parts of the database from certain users.

- example of view in security

Suppose, a bank clerk needs to know the names of the customers of each branch, but it is not authorized to see specific loan information.

Approach: Deny direct access to the loan relation, but grant access to the view cust-loan, which access only of the names of customers and the branches at which they have a loan.

The cust-loan view is defined in sql as follows:

```
create view cust-loan as
    select branchname, customer-name from
    borrower, loan
    where borrower.loan-number = loan.loan-number
```

2015 Fall 3b repeated.

2015 spring 5a repeated

2016 Fall also explain authentication & authorization

How is any user prevented from accessing a certain resource

4b) Discuss the access control mechanism.

- Database access control is a method of allowing access to particular's sensitive data only to those user of database who are allowed to access such data and to restrict access to unathorized users.

- It includes two main components - authentication and authorization.

Two levels of access controls are :-

- Physical levels
  - It limits access to offices, rooms, and physical IT assets
- Logical levels
  - It limits connection to computer networks, digital infrastructure, system files and data.

- Access control is necessary because it regulates which user, applications and devices can view, edit add and delete resources in an organization's environment.
- Controlling access is one of the key practices to protect sensitive data from theft, misuse, abuse and any other threats.
- Types of Access control

Obsolete access models includes DAC, MAC, RBAC is the most common method today, and the most recent model is Attribute based Access control (ABAC)

A) Discretionary Access control (DAC)

- DAC is an identity-based access control model that provides users a certain amount of control over their data. Data owners (or any users authorized to control data) can define access permissions for specific users or groups of users.
- Access permissions for each piece of data are stored in an Access-control list (ACL).

260

- The list can be generated automatically when a user grant access to somebody or can be created by an administrator. In this case, ACL acts as a security policy, and regular users can't edit or override it.

⇒ Gaining access to DAC model works like this:

- user 1 creates a file and becomes its owner or obtains access rights to an existing file.
- user 2 requests access to the file
- user 1 grants access if at their own discretion. However, user 1 can't grant access rights that exceed their own. For eg:- if user 1 can only read a document, they can't allow user 2 to edit it
- If there's no contradiction between the ACL created by an administrator and the decision made by user 1, access is granted.

- DAC (access control) is quite a popular model because it allows a lot of freedom for users and doesn't cause administrative overhead. However it has several considerable limitations also.

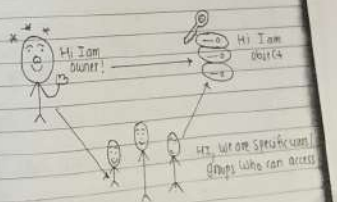


Fig:- DAC

261

- Pros of DAC

- User-friendly :- User can manage their data and quickly access data of other users.
- Flexible :- User can configure data access parameters without administrator.
- Easy to maintain :- Adding new objects and users doesn't take much time for the administrator.
- Granular :- Users can be avoided here they can configure access parameters for each piece of data.

- Cons of DAC

- Low level of data protection :- DAC can't ensure reliable security because users can share their data however they like.
- Obscure :- There's no centralized access management, so in order to find out access parameters, you have to check each ACL.

A.ii) Mandatory Access control (MAC) :-

- MAC is a model of access control where the OS provides users with access based on data confidentiality and user clearance levels.

- In this model, access is granted on a need to know basis: users have to prove a need for information before gaining access.

- MAC is considered the most secure of all access control models.

=> With MAC, the process of gaining access looks like this.

- The administrator configures access policies and defines security attributes: confidentiality levels, clearances for accessing different points and types of systems.

The administrator assigns each subject (user or process that accesses data) and object (files, database, ports etc.) a set of attributes.

When a subject attempts to access an object, the OS examines the subject's security attributes and decides whether access can be granted.

For eg :- Let's consider data that has the "top secret" confidentiality level and "engineering papers" security label. It's available to a set of users that have "top secret" clearance and authorization to access engineering documents. Such users can also access information that requires a lower level of clearance. But employees with lower levels of clearance will not have access to information that requires a higher level of clearance.

- MAC brings lots of benefits to a cybersecurity system.

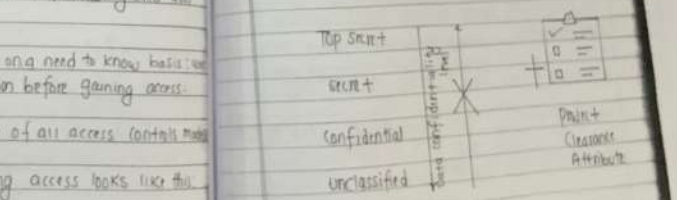


Fig :- MAC

- MAC is used by the US government to secure classified information and to support multilevel security policies and applications.
- This access control model is mostly used by government organizations, militaries and law enforcement institutions.

Pros of MAC

- High level of data protection - An administrator defines access to objects, and users can't edit that access.
- Granular - An administrator sets user access rights and object access parameters manually.
- Immune to Trojan Horse attacks - users can't declassify data or share access to classified data.

Cons of MAC

- Maintainability - Manual configuration of security levels and clearances requires constant attention from administrators.
- Scalability - MAC doesn't scale automatically.
- Not user-friendly - users have to request access to each new piece of data; they can't configure access parameters for their own data.

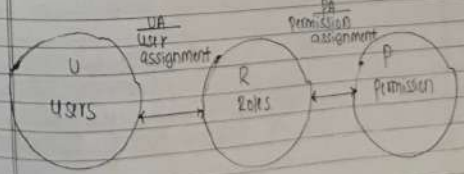
iii) Role-based access control (RBAC)

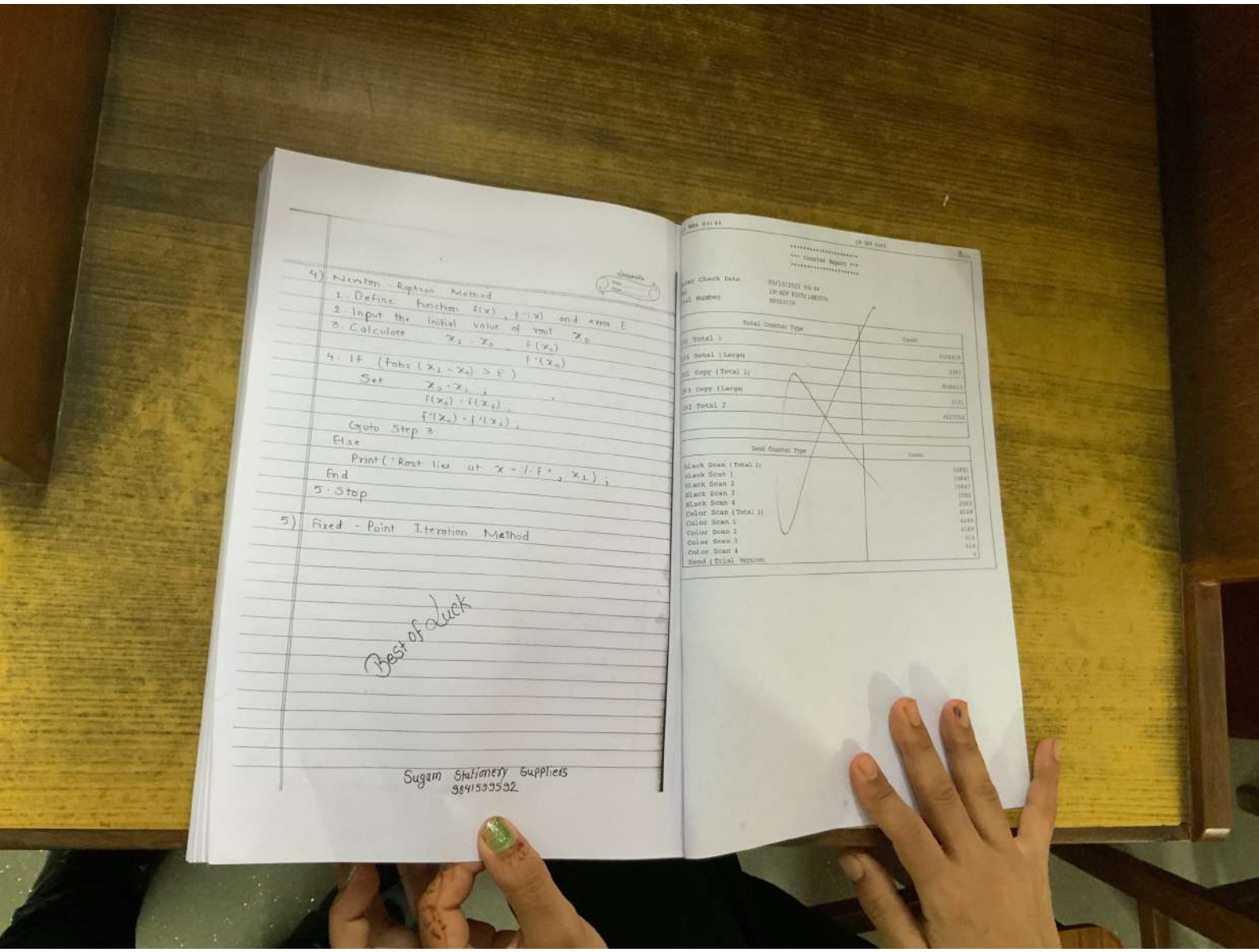
- RBAC is an access control method based on defining employee roles and corresponding privileges within the organization.
- The idea of this model is that every employee is assigned a role. Every role has a collection of permissions and restrictions. A user can access objects and execute operations only if their role in the system has the relevant permissions.
- The main components of Role-based access control (RBAC) are: user, Role, permission, session, object and operation.

⇒ RBAC can be implemented on four levels, according to the NIST RBAC model. Each subsequent level includes the properties of the previous. They are-

i) Flat RBAC

- It is an implementation of the basic functionality of the RBAC model.
- Here a single user can be assigned to multiple roles, and one role can be assigned to multiple users.





4) Newton Raphson Method

1. Define function $f(x)$, $f'(x)$ and error E
2. Input the initial value of root x_0
3. Calculate $x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$
4. If $(\text{fabs}(x_1 - x_0) > E)$

   Set $x_0 = x_1$,

   $f(x_0) = f(x_1)$,

   $f'(x_0) = f'(x_1)$,

   Goto Step 3

   Else

   Print("Root lies at x = %f", x_1);

   End

5. Stop

5) Fixed - Point Iteration Method

Best of luck

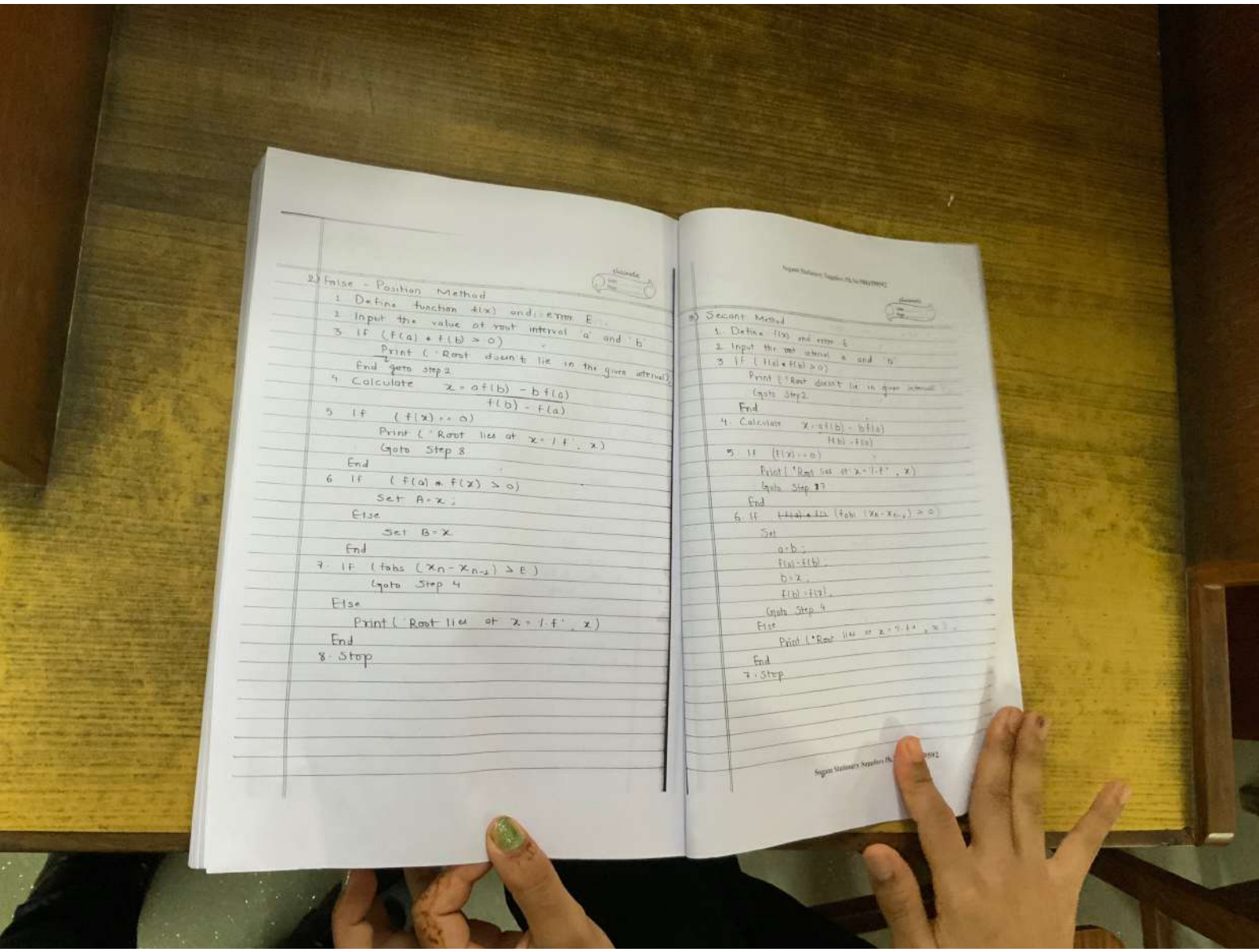
Sugam Stationery Supplies
9841533532.

Total Counter Type

	Count
Total 1	
Total 1 Large	
Copy (Total 1)	
Copy (Large	
Total 2	

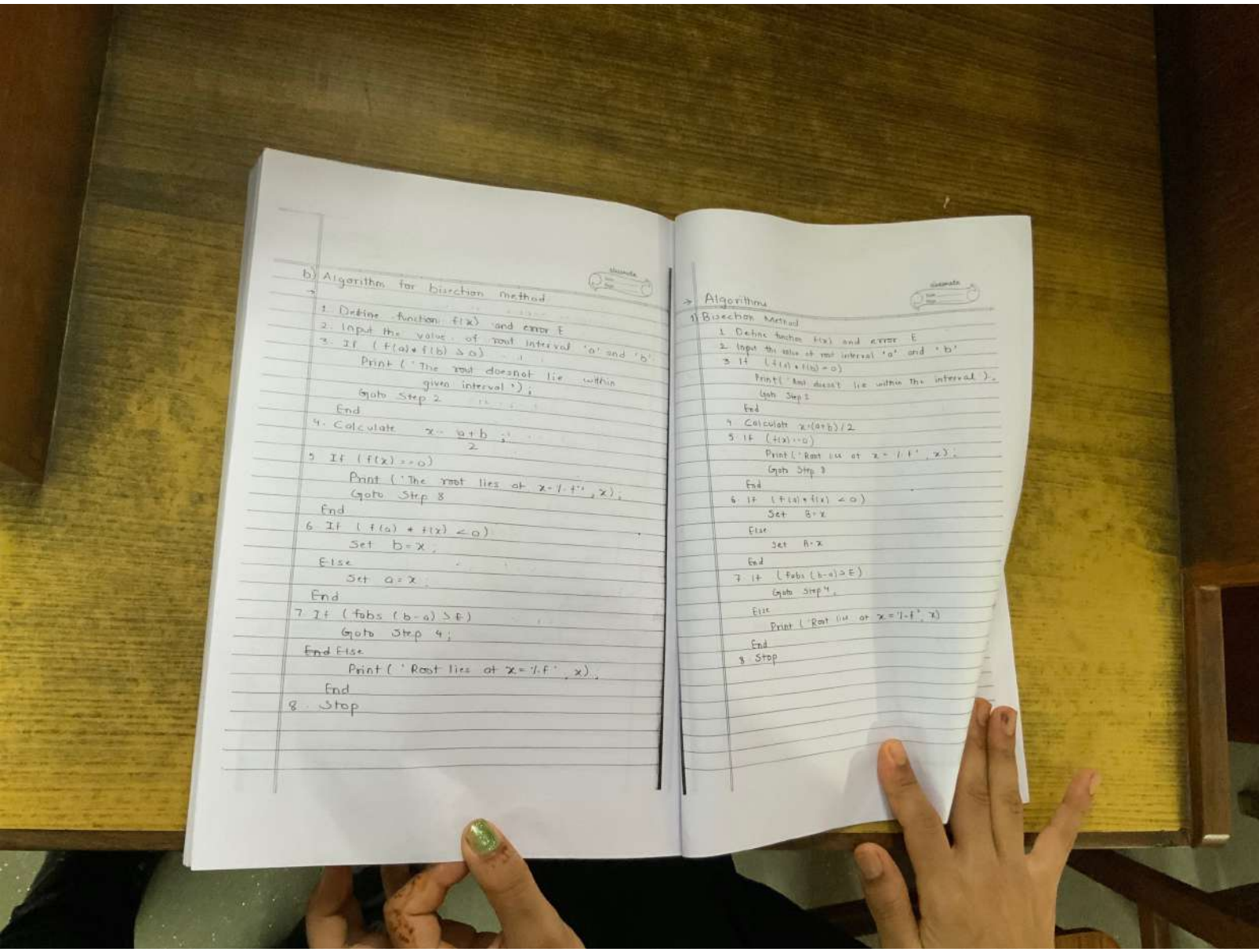
Send Counter Type

	Count
Black Scan (Total 1)	
Black Scan 1	
Black Scan 2	
Black Scan 3	
Black Scan 4	
Color Scan (Total 1)	
Color Scan 1	
Color Scan 2	
Color Scan 3	
Color Scan 4	
Send (Trial Version)	



2) False - Position Method
1. Define function f(x) and error E
2. Input the value of root interval 'a' and 'b'
3. If (f(a) * f(b) > 0)
   Print ('Root doesn't lie in the given interval')
   End goto step 2
4. Calculate $x = \frac{af(b) - bf(a)}{f(b) - f(a)}$
5. If (f(x) == 0)
   Print ('Root lies at x = ', x)
   Goto Step 8
   End
6. If (f(a) * f(x) > 0)
   Set A = x;
   Else
   Set B = x
   End
7. If (fabs ($x_n - x_{n-1}$) > E)
   Goto Step 4
   Else
   Print ('Root lies at x = ', x)
   End
8. Stop

3) Secant Method
1. Define f(x) and error E
2. Input the int interval a and b
3. If (f(a) * f(b) > 0)
   Print ('Root doesn't lie in given interval')
   Goto Step 2
   End
4. Calculate $x = \frac{af(b) - bf(a)}{f(b) - f(a)}$
5. If (f(x) == 0)
   Print ('Root lies at x = ', x)
   Goto Step 7
   End
6. If ~~f(a) * f(x)~~ (fabs ($x_n - x_{n-1}$) > 0)
   Set
   a = b;
   f(a) = f(b);
   b = x;
   f(b) = f(x);
   Goto Step 4
   Else
   Print ('Root lies at x = ', x)
   End
7. Stop



b) Algorithm for bisection method

1. Define function: f(x) and error E
2. Input the value of root interval 'a' and 'b'
3. If (f(a)*f(b) > 0)
   Print ('The root doesnot lie within given interval');
   Goto Step 2
   End
4. Calculate x = (a+b)/2;
5. If (f(x) == 0)
   Print ('The root lies at x = %f', x);
   Goto Step 8
   End
6. If (f(a) * f(x) < 0)
   Set b = x;
   Else
   Set a = x;
   End
7. If (fabs (b-a) > E)
   Goto Step 4;
   End Else
   Print ('Root lies at x = %f', x);
   End
8. Stop

→ Algorithm

1) Bisection Method
1. Define function f(x) and error E
2. Input the value of root interval 'a' and 'b'
3. If (f(a)*f(b) > 0)
   Print('Root doesn't lie within the interval');
   Goto Step 2
   End
4. Calculate x=(a+b)/2
5. If (f(x)==0)
   Print('Root lies at x = %f', x);
   Goto Step 8
   End
6. If (f(a)*f(x) < 0)
   Set B=x
   Else
   Set A=x
   End
7. If (fabs(b-a)>E)
   Goto Step 4;
   Else
   Print('Root lies at x=%f', x)
   End
8. Stop

$u_1 = -\frac{1}{\sqrt{3}}$, $u_2 = \frac{1}{\sqrt{3}}$

$\therefore I = 1 \times f\left(-\frac{1}{\sqrt{3}}\right) + 1 \times f\left(\frac{1}{\sqrt{3}}\right)$

Changing limit $[0.5, 1.5]$ to $[-1, 1]$ by applying transformation

$$x = \frac{b+a}{2}u + \frac{b-a}{2}$$

$$= \frac{1.5+0.5}{2}u + \frac{1.5-0.5}{2}$$

$$\therefore x = \frac{2u+1}{2}$$

For $n=2$,

$$I = 1 \times e^{2\left(\frac{-1/\sqrt{3}}{2}\right)+1} + 1 \times e^{2\left(\frac{1/\sqrt{3}}{2}\right)+1}$$

$$= 3.8624$$

For $n=3$,

$I = w_1 f(u_1) + w_2 f(u_2) + w_3 f(u_3)$

where $w_1 = \frac{5}{9}$, $w_2 = \frac{8}{9}$, $w_3 = \frac{5}{9}$

$x_1 = -\sqrt{\frac{3}{5}}$, $x_2 = 0$, $x_3 = \sqrt{\frac{3}{5}}$

$$\therefore I = \frac{5}{9} e^{2\left(\frac{-\sqrt{3/5}+1}{2}\right)} + \frac{8}{9} e^{2\left(\frac{0+1}{2}\right)} + \frac{5}{9} e^{2\left(\frac{\sqrt{3/5}+1}{2}\right)}$$

$$= 3.8750$$

Write short notes

Ill Conditioned System

If small change in matrix of coefficient variable produces large change in result of any simultaneous linear system of eqn then the system is said to be ill conditioned system. It may occur when the determinant of coefficient matrix is very small or nearly equal to zero

Example :

$$\begin{bmatrix} 400 & 201 \\ -800 & 401 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 200 \\ -200 \end{bmatrix}$$

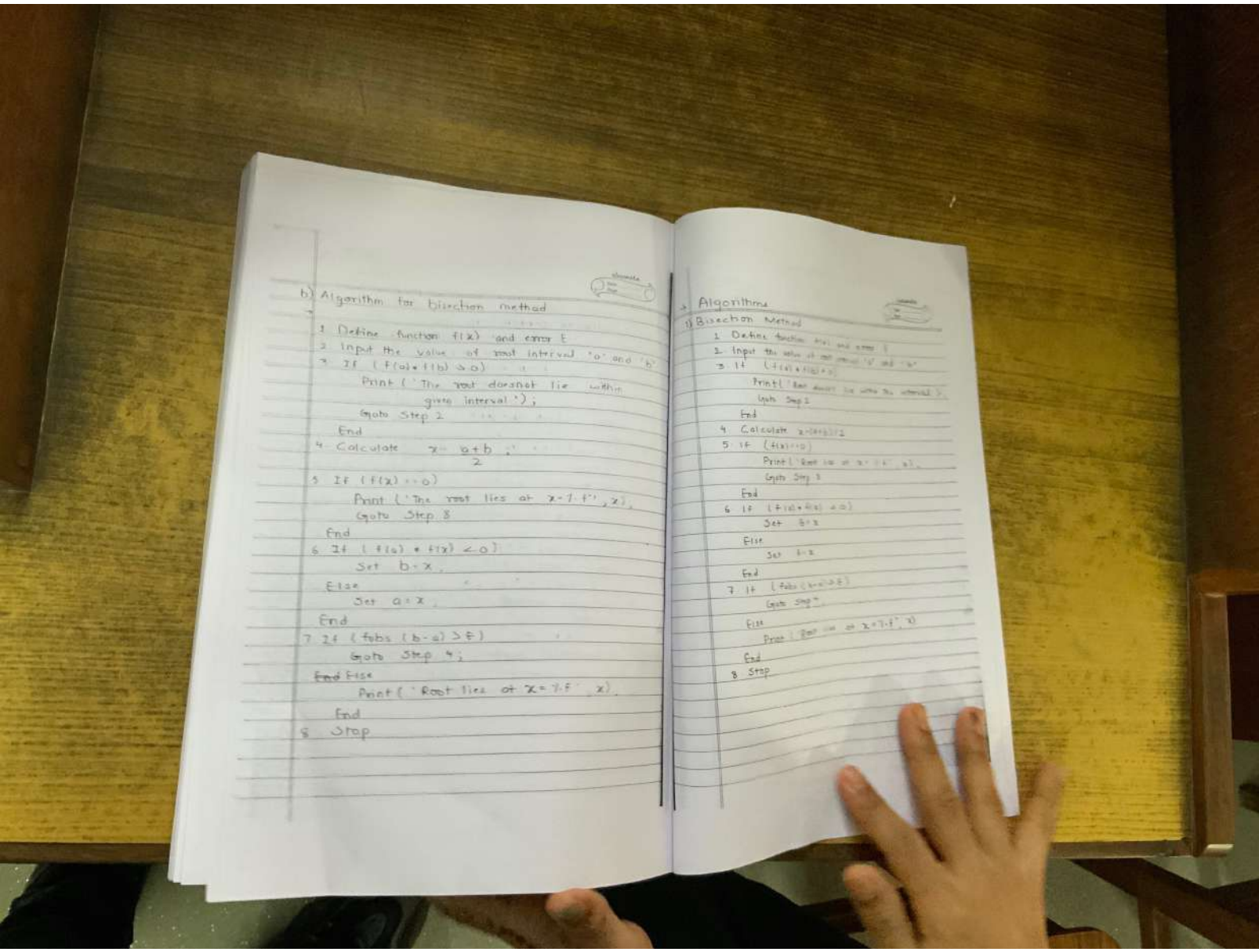
The system produces solution as $x_1 = -100$ and $x_2 = -200$

While changing by small value in coefficient matrix value as given

$$\begin{bmatrix} 401 & -201 \\ -800 & 401 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 200 \\ -200 \end{bmatrix}$$

The system gives solution as $x_1 = 40000$ and $x_2 = 79800$

The ill-conditioned system can be resolved either by scaling an equation ie ($\times 10$ 20) or changing the method of solution for ill conditioned system.

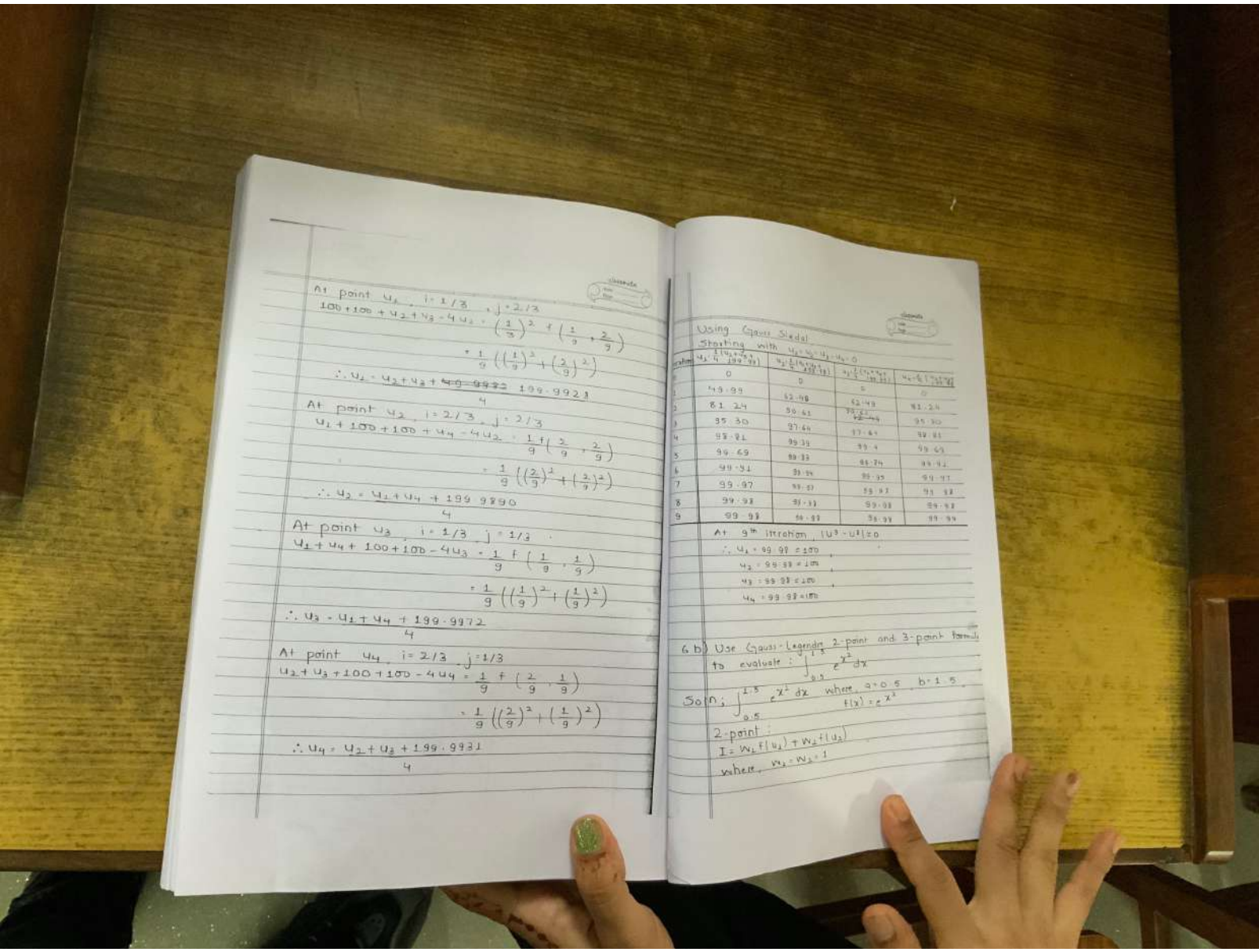


b) Algorithm for bisection method

1 Define function f(x) and error E
2 Input the value of root interval 'a' and 'b'
3 If (f(a)*f(b) > 0)
    Print ('The root doesnot lie within given interval');
    Goto Step 2
  End
4 Calculate x = (a+b)/2;
5 If (f(x) == 0)
    Print ('The root lies at x=%f', x);
    Goto Step 8
  End
6 If (f(a) * f(x) < 0)
    Set b = x;
  Else
    Set a = x;
  End
7 If (fabs (b-a) > E)
    Goto Step 4;
  End Else
    Print ('Root lies at x=%f', x);
  End
8 Stop

Algorithm

1) Bisection Method
1 Define function and error
2 Input the value of root interval 'a' and 'b'
3 If (f(a)*f(b) > 0)
    Print('Root doesn't lie within the interval');
    Goto Step 2
  End
4 Calculate x=(a+b)/2
5 If (f(x)==0)
    Print('Root lie at x' (f x), x);
    Goto Step 8
  End
6 If (f(a)*f(x) < 0)
    Set b=x
  Else
    Set a=x
  End
7 If (fabs(b-a)>E)
    Goto Step 4
  Else
    Print('Root lie at x=%f', x)
  End
8 Stop



At point u_1, $i = 1/3$, $j = 2/3$

$$100 + 100 + u_2 + u_3 - 4u_1 = \frac{1}{9}f\left(\frac{1}{3}, \frac{2}{3}\right)$$

$$= \frac{1}{9}\left(\left(\frac{1}{3}\right)^2 + \left(\frac{2}{3}\right)^2\right)$$

$$\therefore u_1 = \frac{u_2 + u_3 + 199.9923}{4}$$

At point u_2, $i = 2/3$, $j = 2/3$

$$u_1 + 100 + 100 + u_4 - 4u_2 = \frac{1}{9}f\left(\frac{2}{3}, \frac{2}{3}\right)$$

$$= \frac{1}{9}\left(\left(\frac{2}{3}\right)^2 + \left(\frac{2}{3}\right)^2\right)$$

$$\therefore u_2 = \frac{u_1 + u_4 + 199.9890}{4}$$

At point u_3, $i = 1/3$, $j = 1/3$

$$u_1 + u_4 + 100 + 100 - 4u_3 = \frac{1}{9}f\left(\frac{1}{3}, \frac{1}{3}\right)$$

$$= \frac{1}{9}\left(\left(\frac{1}{3}\right)^2 + \left(\frac{1}{3}\right)^2\right)$$

$$\therefore u_3 = \frac{u_1 + u_4 + 199.9972}{4}$$

At point u_4, $i = 2/3$, $j = 1/3$

$$u_2 + u_3 + 100 + 100 - 4u_4 = \frac{1}{9}f\left(\frac{2}{3}, \frac{1}{3}\right)$$

$$= \frac{1}{9}\left(\left(\frac{2}{3}\right)^2 + \left(\frac{1}{3}\right)^2\right)$$

$$\therefore u_4 = \frac{u_2 + u_3 + 199.9931}{4}$$

Using Gauss Siedal

Starting with $u_1 = u_2 = u_3 = u_4 = 0$

iteration	$u_1 = \frac{1}{4}(u_2 + u_3 + 199.99)$	$u_2 = \frac{1}{4}(u_1 + u_4 + 199.98)$	$u_3 = \frac{1}{4}(u_1 + u_4 + 199.99)$	$u_4 = \frac{1}{4}(u_2 + u_3 + 199.99)$
0	0	0	0	0
1	49.99	62.48	62.49	81.24
2	81.24	90.61	90.62	95.30
3	95.30	97.64	97.64	98.81
4	98.81	99.39	99.4	99.69
5	99.69	99.85	99.84	99.92
6	99.92	99.94	99.95	99.97
7	99.97	99.97	99.97	99.98
8	99.98	99.99	99.99	99.99
9	99.99	99.99	99.99	99.99

At 9th iteration, $|u^9 - u^8| = 0$

$\therefore u_1 = 99.99 \approx 100$

$u_2 = 99.99 \approx 100$

$u_3 = 99.99 \approx 100$

$u_4 = 99.99 \approx 100$

6 b) Use Gauss-Legendre 2-point and 3-point formula to evaluate: $\int_{0.5}^{1.5} e^{x^2}\,dx$

Soln: $\int_{0.5}^{1.5} e^{x^2}\,dx$ where $a = 0.5$, $b = 1.5$, $f(x) = e^{x^2}$

2-point:

$I = W_1 f(u_1) + W_2 f(u_2)$

where, $W_1 = W_2 = 1$

5 h) Solve : $y'' = y + e^x$, $y(0) = 0$ for $y(0.2)$ and $y(0.4)$ by RK 4th order method

Soln:

$\frac{dy}{dx} = y + e^x$, $y(0) = 0$

$k_1 = hf(x_0, y_0)$
$= 0.2 f(0, 0)$
$= 0.2 (0 + e^0)$
$= 0.2$

$k_2 = hf(x_0 + h/2, y_0 + k_1/2)$
$= 0.2 f(0.1, 0.1)$
$= 0.2 (0.1 + e^{0.1})$
$= 0.24$

$k_3 = hf(x_0 + h/2, y_0 + k/2)$
$= 0.2 f(0.1, 0.12)$
$= 0.2 (0.12 + e^{0.12})$
$= 0.24$

$k_4 = hf(x_0 + h, y_0 + k_3)$
$= 0.2 f(0 + 0.2, 0 + 0.24)$
$= 0.2 (0.24 + e^{0.2})$
$= 0.292$

$y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 3k_3 + k_4)$

$y(x_0 + h) = 1 + \frac{1}{6}(0.2 + 2 \times 0.24 + 3 \times 0.24 + 0.292)$

$\therefore y(0.2) = 1.282$

Iteration 2.

$x_0 = 0.2$, $y_0 = 1.24$

$k_1 = hf(x_0, y_0)$
$= 0.2 f(0.2, 1.24)$
$= 0.2 (1.24 + e^{0.2})$
$= 0.492$

$k_2 = hf(x_0 + h/2, y_0 + k_1/2)$
$= 0.2 f(0.3, 1.34)$
$= 0.2 (1.34 + e^{0.3})$
$= 0.537$

$k_3 = hf(x_0 + h/2, y_0 + k_2/2)$
$= 0.2 f(0.3, 1.36)$
$= 0.2 (1.36 + e^{0.3})$
$= 0.541$

$k_4 = hf(x_0 + h, y_0 + k_3)$
$= 0.2 f(0.4, 1.48)$
$= 0.2 (1.48 + e^{0.4})$
$= 0.594$

$y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 3k_3 + k_4)$

$= 1.24 + \frac{1}{6}(0.492 + 2 \times 0.537 + 3 \times 0.541 + 0.594)$

$y(0.4) = 1.8705$

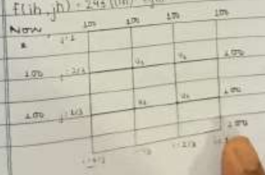
6 a) Solve the Poisson's equation $u_{xx} + u_{yy} = 243(x^2 + y^2)$ over a square domain $0 \le x \le 1$, $0 \le y \le 1$ with step size $h = 1/3$ with $u = 100$ on the boundary

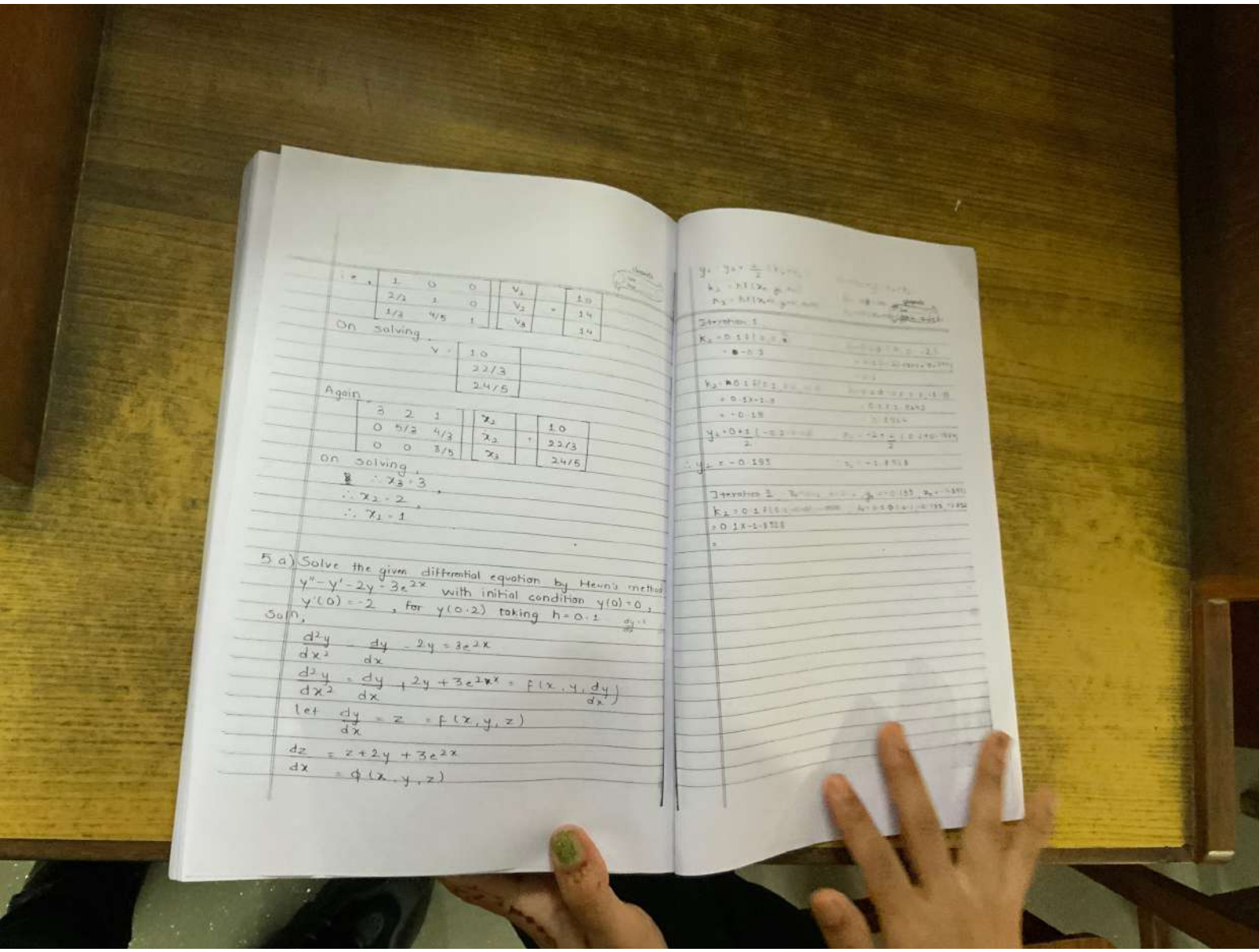
Soln:

$u_{xx} + u_{yy} = 243(x^2 + y^2) = f(x, y)$

$f(ih, jh) = 243((ih)^2 + (jh)^2)$

Now,





$$\begin{bmatrix} 1 & 0 & 0 \\ 2/3 & 1 & 0 \\ 1/3 & 4/5 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} 1.0 \\ 1.4 \\ 1.4 \end{bmatrix}$$

On solving,

$$v = \begin{bmatrix} 1.0 \\ 22/3 \\ 24/5 \end{bmatrix}$$

Again,

$$\begin{bmatrix} 3 & 2 & 1 \\ 0 & 5/2 & 4/3 \\ 0 & 0 & 3/5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1.0 \\ 22/3 \\ 24/5 \end{bmatrix}$$

On solving,

$\therefore x_3 = 3$

$\therefore x_2 = 2$

$\therefore x_1 = 1$

5 a) Solve the given differential equation by Heun's method $y'' - y' - 2y = 3e^{2x}$ with initial condition $y(0) = 0$, $y'(0) = -2$, for $y(0.2)$ taking $h = 0.1$

Soln,

$$\frac{d^2y}{dx^2} - \frac{dy}{dx} - 2y = 3e^{2x}$$

$$\frac{d^2y}{dx^2} = \frac{dy}{dx} + 2y + 3e^{2x} = f\left(x, y, \frac{dy}{dx}\right)$$

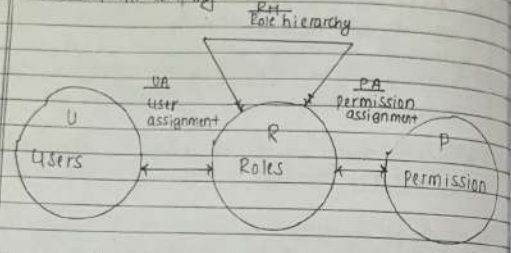
let $\frac{dy}{dx} = z = f(x, y, z)$

$$\frac{dz}{dx} = z + 2y + 3e^{2x}$$

$$= \phi(x, y, z)$$

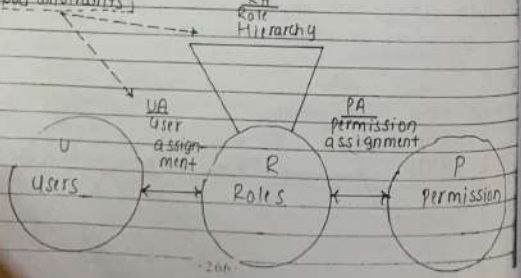
ii) Hierarchical RBAC

- As the name suggests implements a hierarchy within the role structure. The hierarchy establishes the relationships between roles.
- The complexity of the hierarchy is defined by the needs of the company.



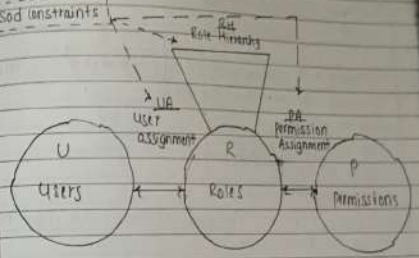
iii) Constrained RBAC :-

- It adds a separation of duties (SOD) to a security system. SOD is a well-known security practice when a single duty is spread among several employees. It's quite important for medium-sized businesses and large enterprises.



iv) Symmetric RBAC

- It supports permission-role review as well as user-role review.
- It allows identification of permissions assigned to existing roles (and vice versa).



Pros of RBAC

- Improving operational efficiency
- Enhancing compliance.
- Reducing costs.
- Decreasing risk of breaches and data leakage.
- Giving administrators increased visibility.

Cons of RBAC

- security Risk tolerance
- Role Explosion
- Expensive and difficult implementation
- Scalability and Dynamism.

A iv) Attribut based access control (ABAC)

- Attribute based access control is a model that evolved from RBAC. This model is based on establishing a set of attributes for any element of your system.
- The main components of the ABAC model accourding to NIST :-
- Attribute :- A characteristic of any element in the network. An attribute can define :
  - User characteristics
  - object characteristics
  - Type of action
  - Environment characteristics.
- Subject - a subject is assigned attributes in order to describe its clearance level.
- object - object are assigned attributes in order to describe and identify them.
- operation - any action taken by any subject in the network.
- Policy - a set of rules allowing or restricting any action in your information retrieval system ; rules are "if/Then" statements based on attributes of any element (user, resource, environment)

Hint,

- Unlike the RBAC, In the ABAC we can even use attributes that aren't yet registered in the system but will appear during the work process.

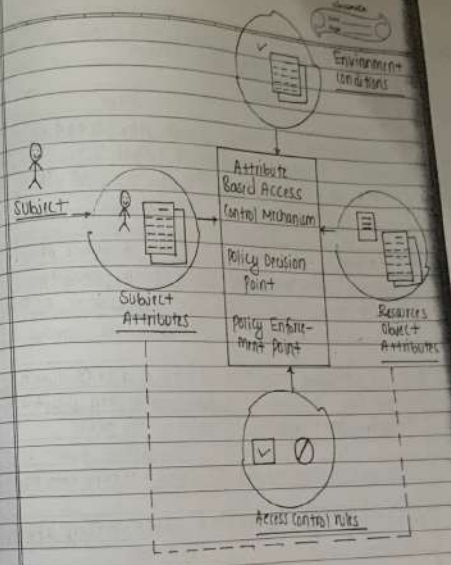


Fig :- ABAC

* Pros of ABAC are :-
  - Dynamic, can be mapped to MAC and RBAC Model
  - Contextual, easy administration
  - Fine grained, scalable, can easily adapt to Risk (RADAC)
* Cons of ABAC are :-
  - Complex to analyze, possibility of attribute explosion.
  - Attribute needs provisioning and maintenance.

2016 spring

2a) How does view differ with relation.

Relation	View
• A relation is used to organize data in the form rows and columns and displayed them in a structured format	• views are treated as a virtual/logical table used to view or manipulate parts of the table
• It makes the stored information more understandable to the human.	• It is a database object that contains rows and columns the same as real relation.
• It is a physical entity that means data is actually stored in the table.	• A view is a virtual entity which means data is not actually stored in the table.
• It is used to store the data.	It is used to extract data from table
• It occupies space on the system.	• It doesn't occupy space on the system.
• It generates a fast result.	• view generates a slow result because it renders the information from the table every time we query it.
• Relation allows us to perform DML operations.	• The view will enable us to perform DML operations.

It is an independent data object.	• It depends on the table we can't create a view without using table.
• It is not an easy task to replace table directly because of it's physical storage.	• It is an easy task to replace the view and recreate it wherever needs.
• SYNTAX :-	SYNTAX :-
CREATE TABLE table_name(column_definations1, datatype column_defination2, ... column_defination3 : column_defination n datatype table_constraints);	CREATE VIEW View_name AS SELECT columns FROM tables [WHERE conditions];
• eg :-	• eg :-
Create table ColorsCinema(Emp_ID int Not null, emp_Name varchar(25) not null, Address[Char varchar(25));	Create View [Brazil Customers] AS Select CostomerName, contactNa From Customers Where Country = 'Brazil';
2017 Fall 4b) repeated. 5a Repeated.	2017 spring 4b) Repeated.

2018 Fall

4.d Why security is needed in database?

i). Safeguards all valuable information: Sensitive information is never supposed to leak. Whether we are talking of bank customer's details or a hospital's patient's information. These are crucial information that are not ment for any prying eye. Data security keeps all this information exactly where it's meant to be.

ii). Important for your reputation: Any organization that can keep secrets also helps to build confidence among all stakeholders including customers, who know that their data is both safe and secure.

iii). Marketing and competitive edge: keeping sensitive information from illegal access and disclosure keeps to your future development or expansion plans is key in maintaining your competitive advantage.

iv). Saves in development and support costs: The earlier you plug security features into your application, the less costs you may incur from any future support and development costs in terms of code modification.

2018 spring repeated

2019 all repeated.

2019 spring

4.b) Explain needs/advantages of authorization, authentication and Access control

Authorization

- Authorization lists simplify managing authorities. User authority is defined for the authorization list, not for the individual objects on the list. If a new object is secured by the authorization list, the users on the list gain authority to the object.
- one operation can be used to give a user authority to all the objects on the list.
- Authorization lists provide a good way to secure files. If we use private authorities, each user will have a private authorities for each file member.
- From a security management view, an authorization list is the preferred method to manage objects that have the same security requirements.
- If we secure the file with an authorization list, we can change the authorities, even when the file is open.

Authentication

- This is used to identify the user of a home, Amora Security Clearance computer system.
- The main purpose of these systems is to validate the user's right to access the system and information, and

protect against identity theft and fraud.

- It identifies an individual process or entity that is attempting to log in to a secure domain.
- It's implementation is pretty simple, as there is no encryption involved and it takes relatively less time to respond as it has only one call.
- It helps to keep that unauthorized people can access APIs, it has to be authenticate the users using a checkpoint.

Access control

- It regulates which users, application and devices can view, edit and delete resources in an organization's environment.
- controlling access is one of the key practices to protect sensitive data from theft, misuse, abuse and only any other threats.
- It provides the level of security that minimizes the risk to the organization by keeping the data secure.
- Prevent against data breaches and to get rid of traditional keys.
- It protect against the unwanted visitors and also it keep the track of who comes and who goes.

2020 Fall

(a) Describe the GRANT functions and explain how it relates to security. What types of privileges may be granted? How rights could be revoked.

Grant functions

- The basic idea of grant statement is that different types of privileges can be granted to the user and if necessary, later revoked.
- This allows the users to manipulate data in the database or perform some actions such as select, create, delete etc. on some privileges that are granted to the users to select data from table, to create a relation or to delete records form a table etc.
- privileges can be system or object:
- System privileges examples are: Drop, Create, Alter etc.
- Object privileges examples are: Insert, execute, Select, update
- There are many types of privileges that can be granted some are to retrive data and add records.

Grant statement

grant <privilege list>
on <relation or view>
to <user/role list>

eg:- To read tuple in a relation, 'select' authorization is required.

eg:-
```
grant select
  on employee
  to Ram, Hari;
```

To update tuple in a relation, 'update' authorization is required.

eg:-
```
grant update
  on employee
  to sita, bita
```

```
grant update(salary)
  on employee
  to sita, Rita.
```

- To insert a tuple in a relation 'insert' authorization is used.
- 'insert' privileges may also specify a list of attributes.
- The system either gives default values or null for remaining attributes.

eg:-
```
grant insert
  on employee
  to Ram, sita
```

```
grant insert(name)
  on employee
  to Ram, sita
```

- To delete tuples from a relation, 'delete' authorization is used.

```
grant delete
  on employee
  to Ram, Hari
```

- Privileges granted to public are implicitly granted to all current and future users.

Grants related to Security

- It makes the information more secure by only giving the users that need to access it the permission to.
- GRANT statement is beneficial for security purposes as it grants certain access and privileges to specific users according to their requirements and does not give access to every object to all users which helps in securing the data and database objects as giving access to all users can sometimes be a security risk.
- So Grant statement by selective access to specific users for specific operations can be beneficial in securing the data in the database. eg from being corrupted in anyway.
- It defines rules for different users for performing different actions to make sure sure that users only access the database in ways they are granted or permitted to.
- GRANT allows each user only the privileges necessary to perform a task.
- So, By this [illegible] statement is related to the [illegible]

Revoke Statement

- The access rights given by the GRANT statement can be revoked by using the REVOKE command.
- This Statement is used to remove the privileges from the users or database objects.
- This statement involves the privileges to be revoked eg:- SELECT, the object on which the privilege is to be revoked such as table name and the username from whom the granted privileges is to be revoked.

SYNTAX:- REVOKE EXECUTE ON [PROCEDURE Function] object FROM user;

- example: function

```
Revoke Execute on Function CalcIncome
FROM 'smithj' @ 'localhost';
```

- example: procedure

```
Revoke execute on
procedure
MyFirstproc
FROM
'*' @ 'localhost';
```


chapter 7:- Query Processing old is gold solutions.

Q) What is query optimization? Explain the steps involved in query processing.

(a) 181546 17

Query

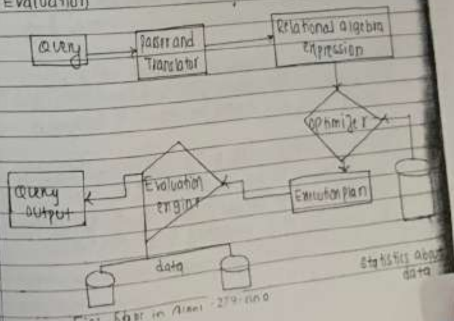
A query is a request for information from a database.

Query Processing with it's steps

- Query processing is transforming a query written in High level language typically in SQL into correct and efficient execution strategy expressed in a low level language and to execute the strategy to retrieve required data.

- Basic steps involved in query processing are:-

i) Parsing and transaction
ii) optimization
iii) Evaluation



~~DBMS~~ 19/2/2018 20

i) Parsing and Translating

- Translate the query into internal form. This is then translated into relational algebra.
- Parser checks syntax, verifies relation.
- The relational algebra expression may have many equivalent expressions.
- The relational algebra representation of a query specifies only partially how to evaluate a query.
- Consider a query:

  Select salary from employee where salary > 20000

  Now, equivalent relational algebra expressions are

  a) $\sigma_{salary>20000}(\pi_{salary}(employee))$
  b) $\pi_{salary}(\sigma_{salary>20000}(employee))$

ii) Evaluation

- The query execution engine takes a query evaluation plan, executes plan and returns the answer to query.
- 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 operations.

~~DBMS~~ 19/2/2018 21

Query optimization

- It is an important aspect of a query processing. It is the activity of choosing an efficient execution for processing a query.
- The cost of the query evaluation can vary from different types of queries. Although the system is responsible for constructing the evaluation plan, the user does need not to write their query efficiently.
- Usually, a database system generates an efficient query evaluation plan, which minimizes its cost. This type of task performed by the database system and is known as query optimization.
- For optimizing a query, the query optimizer should have an estimated cost analysis of each operations. It is because the overall operation cost depends on the memory allocations to several operations, execution costs, and so on.
- Finally, after selecting an evaluation plan, the system evaluates the query and produces the output of the query.

Eg:- Seconds Vs days to execute same query

Cost based query optimization

- Generate logically equivalent expressions by using a set of equivalence rules.

- Annotate the expressions to get alternative query evaluation plans.
- Select the cheapest plan based on estimated cost.
- Estimation of query evaluation cost based on statical information from the catalog manager in combination with expected performance algorithms.

Importance of query optimization

- Provides faster query processing.
- Requires less cost per query.
- Provides high performace.
- consumes less memory.

2011 spring

Q6) Define equivalance of expression?

Any two relational expressions are said to be equivalent if both the expressions generate same set of records.

- When two expressions are equivalent we can use them interchangeably. i.e., we can use either of the expression whichever gives better performance.
- We have different equivalent expression for different types of operations. Equivalence rule defines how to write equivalence expression for each of the operation.

i) conjuctive selection operations can be deconstructed into a sequence of individual selections

- $\sigma_{\theta_1 \wedge \theta_2}(E) = \sigma_{\theta_1}(\sigma_{\theta_2}(E))$

ii) selection operations are commutative

- $\sigma_{\theta_1}(\sigma_{\theta_2}(E)) = \sigma_{\theta_2}(\sigma_{\theta_1}(E))$

iii) cascade of projection operations (only final one)

- $\Pi_{A_1}(\Pi_{A_2}(\ldots(\Pi_{A_n}(E))\ldots)) = \Pi_{A_1}(E)$

iv) selections can be combined with cartesian products and theta joins

- $\sigma_\theta(E_1 \times E_2) = E_1 \bowtie_\theta E_2$
- $\sigma_{\theta_1}(E_1 \bowtie_{\theta_2} E_2) = E_1 \bowtie_{\theta_1 \wedge \theta_2} E_2$

v) Theta join (and natural join) operations are commutative

- $E_1 \bowtie_\theta E_2 = E_2 \bowtie_\theta E_1$
- note that order of attributes is ignored

vi) Natural join operations are associative

- $(E_1 \bowtie_{\theta_1} E_2) \bowtie E_3 = E_1 \bowtie (E_2 \bowtie E_3)$

vii) Theta joins operation are associative in the following manner

- $(E_1 \bowtie_{\theta_1} E_2) \bowtie_{\theta_2 \wedge \theta_3} E_3 = E_1 \bowtie_{\theta_1 \wedge \theta_3} (E_2 \bowtie_{\theta_2} E_3)$
- where θ_2 contains attributes only from E_2 and E_3

viii) union and intersection operations are commutative

- $E_1 \cup E_2 = E_2 \cup E_1$
- $E_1 \cap E_2 = E_2 \cap E_1$

ix) Union and intersection operations are associative
- $(E_1 \cup E_2) \cup E_3 = E_1 \cup (E_2 \cup E_3)$
- $(E_1 \cap E_2) \cap E_3 = E_1 \cap (E_2 \cap E_3)$

The selection operation distributes over union, intersection and set difference
- $\sigma_p(E_1 - E_2) = \sigma_p(E_1) - \sigma_p(E_2)$

The projection distributes over the union operation
- $\pi_A(E_1 \cup E_2) = (\pi_A(E_1)) \cup (\pi_A(E_1))$

Note that this is only a selection of equivalence rules.

2012 Fall repeated. 2012 spring repeated.
2013 spring repeated. 2013 spring repeated.

2014 Fall

6a) How the query optimization process is carried out? Explain about cost estimation of query.

Cost estimation of query

- The main aim of query optimization is to choose the most efficient way of implementing the relational algebra operations at the lowest possible cost.
- The query optimizer should not depend solely on heuristic rules, but it should also estimate the cost of executing the different strategies and find out the strategy with minimum cost estimate.
- The cost of an operation is depend of a cardinality of its input.

To estimate the cost of different query execution strategies, the query tree is viewed as containing a series of basic operations which are linked in order to perform the query.

It is also important to know the expected cardinality of an operation's output because this forms the input to the next operation.

- The cost of executing the query includes the following components.
  - Access cost to secondary storage
  - Storage cost
  - Computation cost
  - Memory uses cost
  - Communication cost.

2014 Spring

6b) Show how to derive the following equivalance by a sequence of transformation using equivalance rules.

i) $\sigma_{\theta_1 \wedge \theta_2 \wedge \theta_3}(E) = \sigma_{\theta_1}(\sigma_{\theta_2}(\sigma_{\theta_3}(E)))$

ii) $\sigma_{\theta_1 \wedge \theta_2}(E_1 \bowtie_{\theta_3} E_2) = (\sigma_{\theta_1}(E_1)) \bowtie_{\theta_3} (\sigma_{\theta_2}(E_2))$

Where θ_2 involves only attributes from E_2.

2015 Fall

2016 spring repeated, 2016 Fall repeated

Steps query optimization By Algebric manipulation

An algebric manipulation approach to query optimization often uses the following steps:-

- Selection operation is to be performed as early as possible in order to reduce tuple volume.
- Projections of projection are to be combined, if possible
- Projection over indexed attributes should be done earlier and that over non-indexed attributes should be deferred.
- Expressions are to be rearranged according to the estimated partitioning efficiency of the terms involved.
- Intermediate relations produced in seperate processing sequences must be shared as and when possible.
- If possible; attributes which are controlling the join operation may be sorted earlier.

Steps query optimization with example VVI

query optimization involves three steps they are

i) query tree generation
ii) Plan generation
iii) query plan code generation.

i) query tree generation

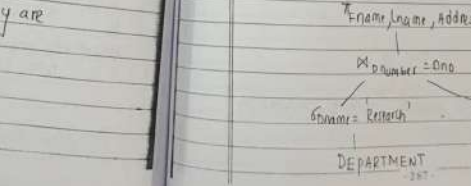
- A query tree is a tree data structure representing a relational algebra expression.
- The tables of the query are represented as leaf nodes. The relational algebra operations are represented as the internal nodes. The root represents as the query as a whole.
- During execution, an internal node is executed whenever its operand tables are available. The node is then replaced by the result table. This process continues for all internal nodes until the root node is executed and replaced by the result table.

For eg:-

consider a following relational algebra

$\pi_{Fname, Lname, Address}(\sigma_{Dname='Research'}(DEPARTMENT) \bowtie_{Dnumber=Dno}(EMPLOYEE))$

The above query tree is



ii) query plan Generation

- After the query tree is generated, a query plan is made. A query plan is an extended query tree that includes access paths for all operations in the query tree. Access paths specify how the relational operations in the tree should be performed.
- For eg:- A selection operation can have an access path that gives details about the use of B+ tree index for selection.

Besides, a query plan also states how the intermediate tables should be passed from one operator to the next, how temporary files tables should be used and how operations should be pipelinned / combined.

iii) Code generation

- Code generation is the final step in query optimization. It is the executable form of the query, whose form depends upon the type of the underlying OS.
- Once the query code is generated, the Execution Manager runs it and produces the results.

dolestring repeated

2017 Fall repeated 2017 Spring

Explain the Heuristic based Choice of evaluation plan for query optimization.

- It uses rule-based optimization approaches for query optimization approaches for query optimization.
- These algorithms have polynomial time and space complexity which is lower than the exponential complexity of exhaustive search-based algorithms.
- However, these algorithms do not necessarily produce the best query plan.

Some of the common heuristic rules are:-

- performs select and project operations before join operation. This is done by moving the select and project operations down the query tree. This reduces the number of tuples available for join.
- perform the most restrictive select / project operations at first before the other operations.
- Avoid cross-product operation since they result in very large-sized intermediate table.

Chapter 8:- File organization and indexing.

2011 Fall

5b) What is file organization? Explain heap file organization starting its advantages and disadvantages.

- It is a method of arranging the records in the file when the file is stored on disk. A relation is typically stored as a file of records.
- The File is a collection of records. Using the primary key, we can access the records. The type and frequency of access can be determined by the type of file organization which was used for a given set of records.
- File organization is a logical relationship among various records. This method defines how file records are mapped onto disk blocks.
- File organization is used to describe the way in which the records are stored in terms of blocks, and the blocks are placed on the storage medium
- The first approach to map the database to the file is to use several files and store only one fixed length record in any given file. An alternative approach is to structure our files so that we can contain multiple lengths for records.
- Files of fixed length records are easier to implement than the files of variable length records.

Objective of File organization

- It contains an optimal selection of records, i.e. records can be selected as fast as possible.
- To perform insert, delete or update transaction on the records should be quick and easy.
- The duplicate records cannot be induced as a result of insert, update or delete.
- For the minimal cost of storage, records should be stored efficiently.

Types of file organization

File organization contains various methods. These particular methods have pros and cons on the basis of access or selection. In the File organization, the programmer decides the best-suited file organization method according to his/her requirements.

Types of file organization

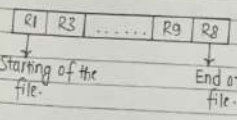
1) Sequential File organization

This method is the easiest method for file organization. In this method, files are stored sequentially.

- This method can be implemented in two ways:

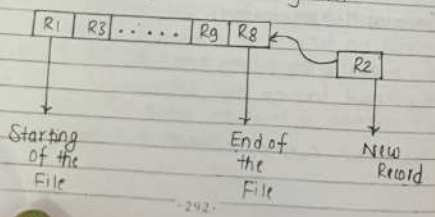
1) Pile File Method:

- It is a quite simple method. In this method, we store the record in a sequence i.e. one after another. Here, the record will be inserted in the order in which they are inserted into tables.
- In case of updating or deleting of any record, the record will be searched in the memory blocks. When it is found, then it will be marked for deleting, and the new record is inserted.



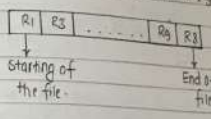
Insertion of the new record:

Suppose we have four records R1, R3 and so on upto R9 and R8 in a sequence. Hence, records are nothing but a row in the table. Suppose we want to insert a new record R2 in the sequence, then it will be placed at the end of the file. Here, records are nothing but a row in any table.



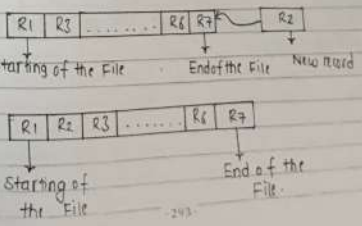
2) Sorted File method

- In this method, the new record is always inserted at the file's end, and then it will sort the sequence is in assending or descending order. sorting of records is based on any primary key or any other key.
- In the case of modification of any record, it will update the record and then sort the file, and lastly, the updated record is placed in the right place.



Insertion of the new Record:

Suppose then is a pre-existing sorted sequence of four records R1, R3 and so on upto R6 and R7. Suppose a new Record R2 has to be inserted in the sequnce, then it will be inserted at the end of the file, and then it will sort the sequence.



⇒ Pros of sequential file organization.

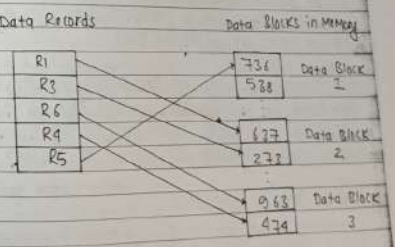
- It contains a fast and efficient method for the huge amount of data.
- In this method, files can be easily stored in cheaper storage mechanism like magnetic tapes.
- It is simple in design. It requires no much effort to store the data.
- This method is used when most of the records have to be accessed like grade calculation of a student, generating the salary slip etc.
- This method is used for report generation or statistical calculations

⇒ Cons of sequential file organization.

- It will waste time as we cannot jump on a particular record that is required but we have to move sequentially which takes our time.
- sorted file method takes more time and space for sorting the records.

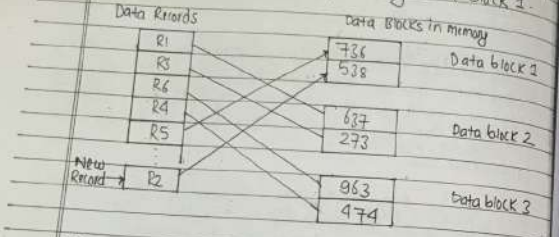
2) Heap File organization

- It is the simplest and most basic type of organization. It works with data blocks. In heap file organization, the records are inserted at the file's end. When the records are inserted, it doesn't require the sorting and ordering of records.
- When the data block is full, the new record is stored in some other block. This new data block need not to be the next data block, but it can select any data block in the memory to store new records. The heap file is also known as unordered file.
- In the file, every record has a unique id, and every page in a file is of the same size. It is the DBMS responsibility to store and manage the new records.



Insertion of a new record:

Suppose we have five records R1, R3, R6, R4 and R5 in a heap and suppose we want to insert a new record R2 in a heap. If the data block 3 is full then it will be inserted in any of the database selected by the DBMS, let's say data block 1.



- If we want to search, update or delete the data in heap file organization, then we need to traverse the data from staring of the file till we get the requested record.
- If the database is very large then searching, updating or deleting of record will be time-consuming because there is no sorting or ordering of records. In the heap file organization, we need to check all the data until we get the requested record.

Pros of Heap file organization.

- It is a very good method of file organization for bulk insertion. If there is a large number of data which needs to load into the database at a time, then this method is best suited.

- In case of small database, fetching and retrieving of records is faster than the sequential record.

Cons of Heap file organization

- This method is inefficient for large databases.
- This method is inefficient for the large database because it takes time to search or modify the record.

2) Hash file organization 4) Bt file organization
5) Index sequential access method 6) cluster file organization.

2011 spring

5a) Explain sequential File organization. What are hash functions explains with example.

Hashing in DBMS

- For a huge database structure, it can be almost next to impossible to search all the index values through all its level and then reach the destination data block to retrive the desired data. Hashing is the effective technique to calculate the direct location of data of a data record on the disk without using index structure.
- Hashing uses search key with hash function as parameters to generate the address of a data record.

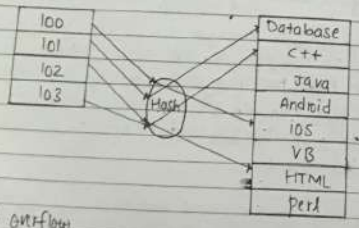
Hash organization

- Bucket - A hash file stores data in bucket format. Bucket is considered a unit of storage. A bucket typically stores one complete disk block, which in turn can store one or more records.

- Hash Function - A hash function, h is a mapping function that maps all the set of search-keys K to the address where actual records are placed. It is a function from search keys to bucket addresses.

Static Hashing

- In static hashing when a search-key value is provided, the hash function always computes the same address. For eg, if mod-4 hash function is used, then it shall generate only 5 values. The output address shall always be same for that function. The number of buckets provided remains unchanged at all times.



Bucket overflow
Operation

- Insertion - When a record is required to be entered using static hash, the hash function h computes the bucket address for search key K, where the record will be stored.
  Bucket address = h(K)

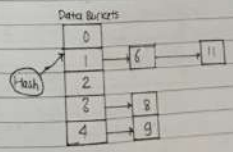
- Search - When a record needs to be retrieved, the same hash function can be used to retrieve the address of the bucket where the data is stored.

- Delete - This is simply a search followed by a deletion operation.

Bucket overflow
The condition of bucket-overflow is known as collision. This is a fatal state for any static hash function. In this case, overflow chaining can be used.

- Overflow Chaining

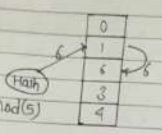
When buckets are full, a new bucket is allocated for the same hash result and is linked after the previous one. This mechanism is called closed Hashing.



- Reasons of bucket overflow
  - Insufficient Buckets
  - Skew in distribution of records. This can occur due to two reasons
    - multiple records have same search-key value

- chosen hash function produces non-uniform distribution of key values.
- Although the probability of bucket overflow can be reduced, it cannot be eliminated; it is handled by using overflow buckets.

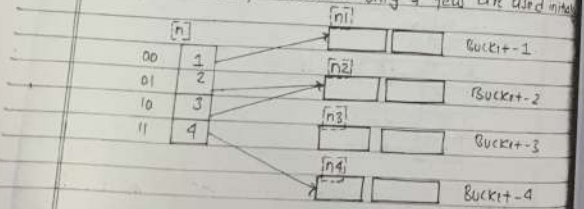
- Linear Probing - when a hash function generates an address at which data is already stored, the next free bucket is allocated to it. This mechanism is called open Hashing.



⇒ Dynamic Hashing

- The problem with static hashing is that it does not expand or shrink dynamically as the size of the database grows or shrinks. Dynamic hashing provides a mechanism in which data buckets are added and removed dynamically and on-demand. Dynamic hashing is also known as extending hashing.

Hash function, in dynamic hashing, is made of to produce a large number of values and only a few are used initially.



organization

The prefix of an entire hash value is taken as a hash index. Only a portion of the hash index value is used for computing bucket addresses. Every hash index has a depth value to signify how many bits are used for computing hash function. These bits can address 2^n buckets. When all these bits are consumed — that is when all the buckets are full — then the depth value is increased linearly and twice the buckets are allocated.

Operation

- Querying - look at the depth value of the hash index and use those bits to compute the bucket address.
- Update - perform a query as above and update the data.
- Deletion - perform a query to locate the desired data and delete the same.
- Insertion - compute the address of the bucket
  - If the bucket is already full.
    - Add more buckets.
    - Add additional bits to the hash value.
    - Re-compute the hash function.
  - else
    - Add data to the bucket.
  - If all the buckets are full, perform the remedies of static hashing.

- Hashing is not favorable when the data is organized in some ordering and the queries require a range of data. When data is discrete and random, hash performs the best.
- Hashing algorithms have high complexity than indexing. All hash operations are done in constant time.

5b) Explain indexing and B-tree file structure.

Indexing

- Indexing is used to optimize the performance of a database by minimizing the number of disk accesses required when a query is processed.
- The index is a type of data structure. It is used to locate and access the data in a database table quickly.

Index structure:

Indexes can be created using some database columns.

Search Key	Data Reference

Fig:- Structure of index.

- The first column in a database is the search key that contains a copy of primary key or candidate key of the table. The values of the primary key are stored in sorted order so that the corresponding data can be accessed easily.
- The second column of the database is the data reference. It contains a set of pointers holding the address of the disk block [illegible]

Indexing Methods.

1) Ordered indices

- The indices are usually sorted to make searching faster. The indices are stored in known as ordered indices.

eg:-

- Suppose we have an employee table with thousands of record and each of which is 10 bytes long. If their IDs start with 1,2,3... and so on and we have to search student with ID-543.
  - In the case of a database with no index, we have to search the disk block from starting till it reaches 543. The DBMS will read the record after reading 543*10 = 5430 bytes.
  - In the case of an index, we will search using indexes and the DBMS will read the record after reading 542*2 = 1084 bytes which are very less compared to the previous case.

2) primary's Indix

- If the indix is created on the basis of the PK of the table, then it is known as primary indexing. These PK are unique to each record and contain 1:1 relation between the records.
- As PK are stored in sorted order, the performance of the searching operation is quite efficient.

- The primary index can be classified into two types they are Dense index and sparse index.

i) Dense index

- The dense index contains an index record for every search key value in the data file. It makes Searching faster.
- In this, the number of records in the index table is same as the number of records in the main table.
- It needs more space to store index record itself. The index records have the search key and a pointer to the actual record on the disk.

UP	→	UP	Agra	1,504,200
USA	→	USA	Chicago	2,789,278
Nepal	→	Nepal	Kathmandu	1,956,624
UK	→	UK	Cambridge	1,366,264

ii) Sparse index

- In the data file, index record appears only for a few items. Each item points to a block.
- In this, instead of pointing to each record in the main table, the index points to the records in the main table in a gap.

UP	→	UP	Agra	1,504,200
Nepal	→	USA	Chicago	2,789,278
UK	→	Nepal	Kathmandu	1,956,624
	→	UK	Cambridge	1,366,264

3) Clustering Index.

- Clustering index is defined as an an ordered data file. The data file is ordered on a non-key field.
- In this case, to identify the record faster, we will group two or more columns to get the unique value and create index out of them. This method is called a clustering index.

eg:- Suppose a company contains several employees in each department. Suppose we use a clustering index, where all employees which belong to the same Dept_ID are considered within a single cluster, and index pointers point to the cluster as a whole. Here Dept_ID is a nonunique key.

Dept_ID	Pointer
1	
2	
3	
4	
5	

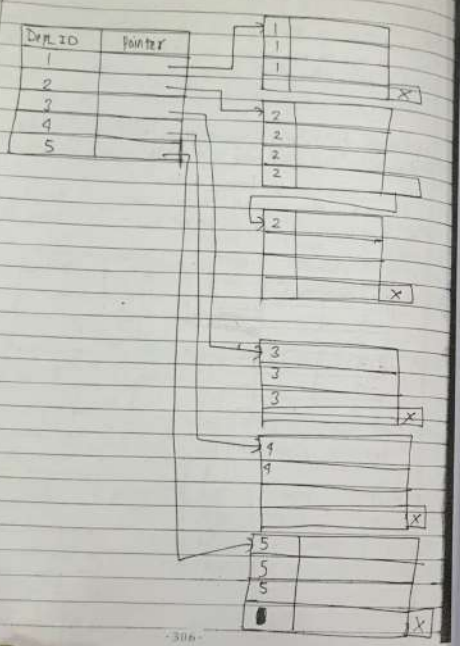
1	Record
1	
1	
1	

2	Record
2	
2	
2	

3	Record
3	
3	
4	

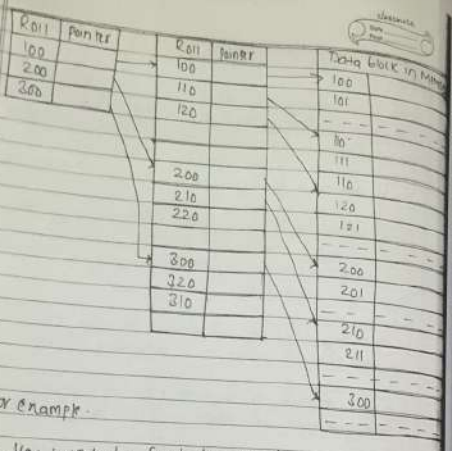
4	Record
5	
5	

The previous schema is little confusing because one disk block is shared by records which belong to the different cluster. If we use seperate disk block for seperate clustrn, then it is called better technique.



2) Secondary Index.

- In the sparse indexing, as the size of the table grows, the size of mapping also grows. These mappings usually kept in the primary memory so that address fetch is faster. Then the secondary memory searches the actual data based on the address got from mapping.
- If the mapping size gets thin fetching the address itself becomes slower. In this case, the sparse index will not be efficient. To overcome this problem, secondary indexing is introduced.
- In secondary indexing, to reduce the size of mapping, another level of indexing is introduced.
- In this method, the huge range for the columns is selected initially so that the mapping size of the first level becomes small.
- The each range is further divided into smaller ranges.
- The mapping of the first level is stored in the primary memory,
- so that the address fetch is faster.
- The mapping of a secondary level is stored in the primary memory and actual data stored in the secondary memory (hard disk).



For example.

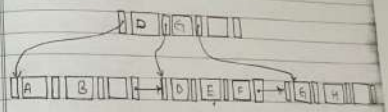
- If you want to find the record of roll 111 in the given diagram, then it will search the highest entry which is smaller than or equal to 111 in the first level index. It will get 100 at this level.
- Then in the second index level, again it does max(111)<= 111. and gets 110. Now using the address 110, it goes to the data block and starts searching each record till it gets 111.
- This is how a search is performed in this method. Insertion, updating or deleting is also done in the same manner.

B+ tree file structure

- A B+ tree is a balanced binary search tree that follows a multi-level index format.
- The leaf node of a B+ tree denotes actual data pointer. B+ tree ensures that all leaf nodes remain at the same height, thus is balanced. Additionally the leaf nodes are linked using a link list, . . a, B+ tree can support random access as well as sequential access.

Structure of a B+ tree

Every leaf node is at equal distance from the root node. B+ tree is of the order of n where 'n' is fixed for every B+ tree.



Internal nodes –

- Internal (non-leaf) node contains at least $\lceil n/2 \rceil$ pointers except the root node.
- At most, an internal node can contain n pointers.

Leaf nodes –

- Leaf nodes contains at least $\lceil n/2 \rceil$ record pointers and $\lceil n/2 \rceil$ key values
- At most, a leaf node can contain n record pointers and n key values.
- Every leaf node contains one block pointer p to point to

next leaf node and forms a linked list.

B+ tree insertion in note copy

⇒ B+ tree deletion

- B+ tree entries are deleted at the leaf nodes
- The target entry is searched and deleted
  - If it is an internal node, delete and replace with the entry from the left position.
- After deletion, underflow is tested,
  - If underflow occurs, distribute the entries from the node left to it
- If distribution is not possible from left, then
  - Distribute from the nodes right to it.
- If distribution is not possible from left or from right, then
  - Merge the node with left and right to it.

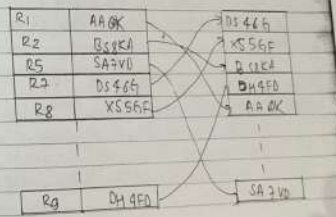
2012 Fall repeated

2012 spring repeated.

2013 Fall

5a) Explain the structure of Index sequential file with the help of a diagram.

- It is an advance sequential file organization. In this method records are stored in the file using the primary key
- An index value is generated for each primary key and mapped with the record.
- The index contains the address of the record in the file.
- It consists of two parts:-
- Data Files - contains the primary key records in sequential Schema
- Index file - contains the PK and its address in the data file.
- If an record has to retrieved on its index value, then the address of the data block is fetched and the record is retrieved from the memory



Data Records

Data Blocks in memory

- Syntax for indexed sequential file organization is :-

```
Input-output section
File-control
Select file-name Assign To dd-name-JCL
organization is indexed
Record key is primary-key
Alternate Record key is sec-key
```

- It consists of a flat file optionally with some internal organization such as fixed or variable length records and pages and one or more associated index files used to quickly position within the file to access specific records without having to traverse all preceding data in the file.

→ pros of ISAM are :-

- In this method, each record has the address of it's data block, searching a record in a huge database is quick and easy.

- This method supports range retrieval and partial retrieval of records. since the index is based on the primary key values, we can retrieve the data for the given range of value. In the same way, the partial value can also be easily searched, i.e. the student name starting with 'JA' can be easily searched.

cons of ISAM

- This method requires extra space in the disk to store the index value.

- when the new records are inserted, then those files have to be reconstructed to maintain the sequence.

- when the record is deleted, then the space used by it needs to be released, otherwise, the performance of the database will slow down.

2013 - Spring repeated.

2014 Fall

5b) When it is preferable to use a dense index rather than a sparse index ? Explain with a suitable example.

- It is preferable to use a dense index instead of a sparse index when the file is not sorted on the indexed field (such as when the index is a secondary index) or when the index file is small compared to the size of memory.

- When the file is not sorted on the indexed field (such as when the index is a secondary index)

- "the file" = the database table (or similar object) being indexed.

- "the indexed field" = the field being considered for the new index. A sparse index relies on the records being sorted in the same order as the indexed field.

example: data in table is sorted on ID:

ID	Last name
1	Smith
2	Francis
3	Jones
4	Zygoski
5	Bahr
6	Josephine
7	Michaels
8	Able

- And let's imagine this is a sparse index on Last Name (with pointers to the appropriate records.

Index
- Able
- Francis
- Jones
- Smith

- Now to find "Bahr", you go to "Able" and then search sequentially. Unfortunately, "Able" is the last record in the file, so, you can't get to "Bahr" from there.

ii) When the index file is small compared to size of memory.

- Basically, if the entire index can be loaded into memory and still have plenty of room to load in the data from the table that's needed, then the database engine should be able to find each record in the index, and go directly to that record in the data table (instead of possibly having to pull in several pages from the table, looking to locate the record it wants).

- If the dense index takes up too much room in the memory, then either you have to read it in parts, or it's going to squeeze the available memory for the rows you need from the data table

- Hence, we can say that on this condition dense index is more preferable than the sparse index.

question vi:

Dense indices are faster in general, but sparse indices requires less space and impose less maintenance for insertion and deletion

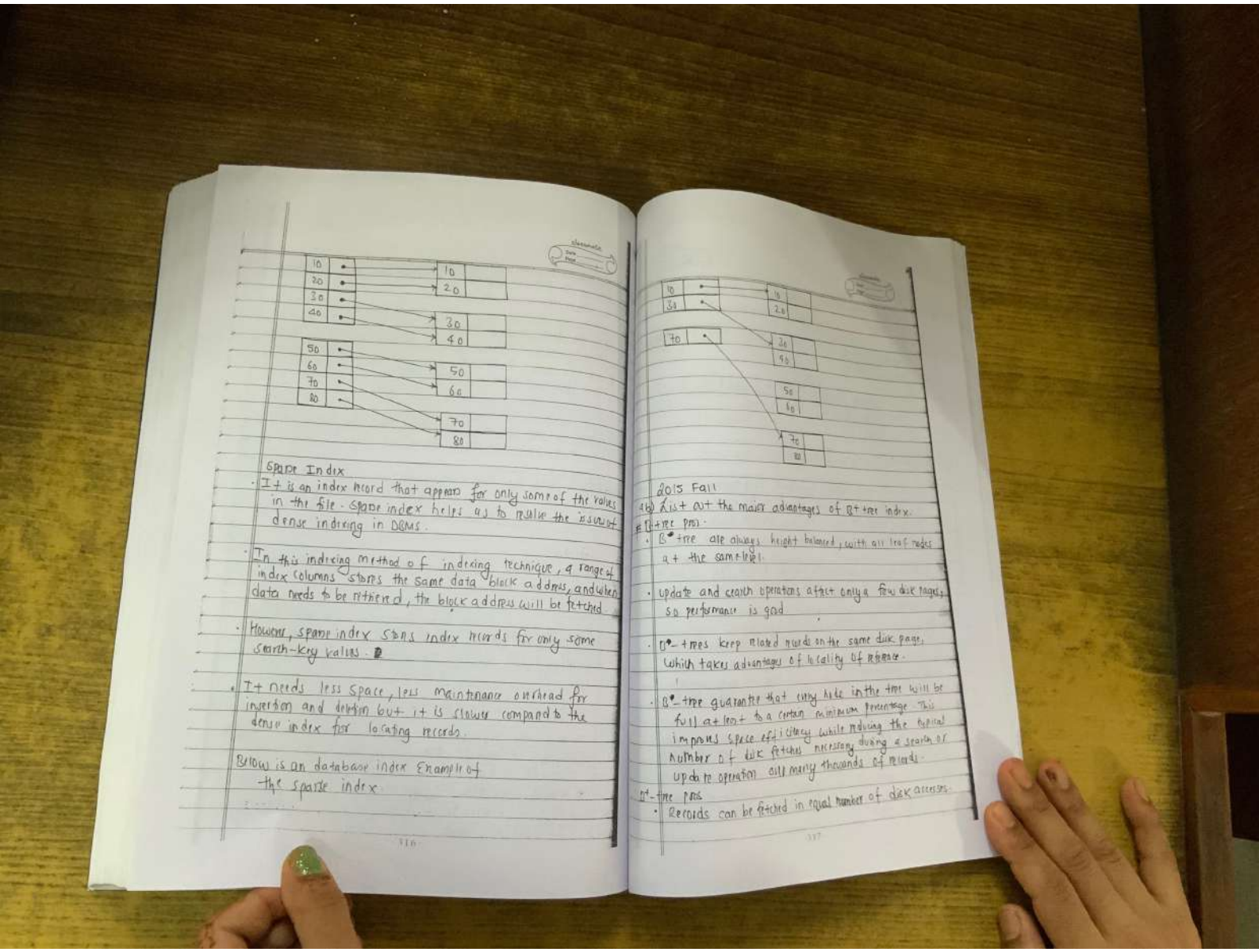
Dense index.

- In a dense index, a record is created for every search key value in the database.

- This helps you to search faster but needs more space to store index records. In this indexing, method records contain search key value and points to the real record on the disk.

- In this, the number of records in the index table is same as the number of records in the main table

- It makes searching faster



Sparse Index

- It is an index record that appears for only some of the values in the file. Sparse index helps us to resolve the issue of dense indexing in DBMS.
- In this indexing method of indexing technique, a range of index columns stores the same data block address, and when data needs to be retrieved, the block address will be fetched.
- However, sparse index stores index records for only some search-key values.
- It needs less space, less maintenance overhead for insertion and deletion but it is slower compared to the dense index for locating records.

Below is an database index Example of the sparse index.

2015 Fall

4(a) List out the major advantages of B+ tree index.

B+ tree pros:

- B+ tree are always height balanced, with all leaf nodes at the same level.
- update and search operations affect only a few disk pages. So performance is good.
- B+ trees keep related records on the same disk page, which takes advantages of locality of reference.
- B+ tree guarantee that every node in the tree will be full at least to a certain minimum percentage. This improves space efficiency while reducing the typical number of disk fetches necessary during a search or update operation with many thousands of records.

B+ tree pros

- Records can be fetched in equal number of disk accesses.

- Height of the tree remains balanced and less as compare to B-tree.
- We can access the data stored in a B+ tree sequentially as well as directly.
- Faster search queries as the data is stored only on the leaf nodes. Keys are used for indexing.

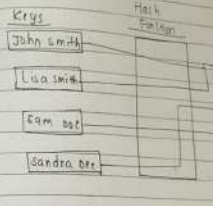
Hash Functions with examples.

hash

- A function that converts function h, is a mapping functions that maps all set of search-keys K to the address where actual records are placed. It is a function from search keys to bucket addresses.
- Choose the hash function that assign search key values to buckets in such a way that distribution is either uniform or Random.
- A good hash function should have the following properties:
  - Efficiently computable.
  - should uniformly distribute the Keys (Each table position equally likely for each key.
- The values returned by a hash function are called hash values, hash codes, digests, or simply hashes.
- The values are usually used to index a fixed-size table called hash table.

use of hash function is to index a hash table is called hashing or scatter storage addressing.

eg:-



- A hash function that maps names to integers from 0 to 15. There is a collision between keys "JohnSmith" and "Sandra Dee".
- Hash functions and their associated hash tables are used in data storage and retrieval applications to access data in a small and nearly constant time per retrieval, and require an amount of storage space only fractionally greater than the total space required for the data or records themselves.
- Case of hash functions relies on statistical properties of key and function interaction:
  worst case behavior is intolerably bad with a non-negligible probability, and average case behavior can be nearly optimal (minimal collision).

2015 spring

2 b) Construct a B+ tree for the following set of key values. (2, 3, 5, 7, 11, 17, 19, 23, 29, 31)
Assume that the tree is initially empty and values are added in ascending order. Construct B+ trees for the case where the number of pointers that will fit in one node is four. Also show the form of the tree after deletion of 23.

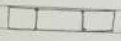
Ans:-

Soln

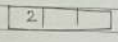
order(n) = 5
max children = 4
min children = $\lceil \frac{n}{2} \rceil = 2$
max key = n-1 = 3
min key = $\frac{n}{2} - 1 = 2$

Here, No of pointers (N) = 4 that means the maximum 3 data can be filled at the node.

When N = 4 then



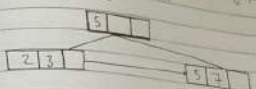
Step I :- Insert 2 in the empty node.



Step II :- Insert 3 and 5 in the node.

Here this has no any space to insert, this is leaf node, so we break it into two nodes.

The 1st partition holding cell value of $(N-1)/2 = 3/2$ = 1.5 ≈ 2 so, and inserting 7 and keeping parent the smallest among second node

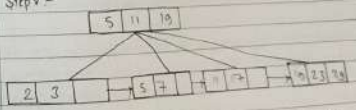


Step III :-

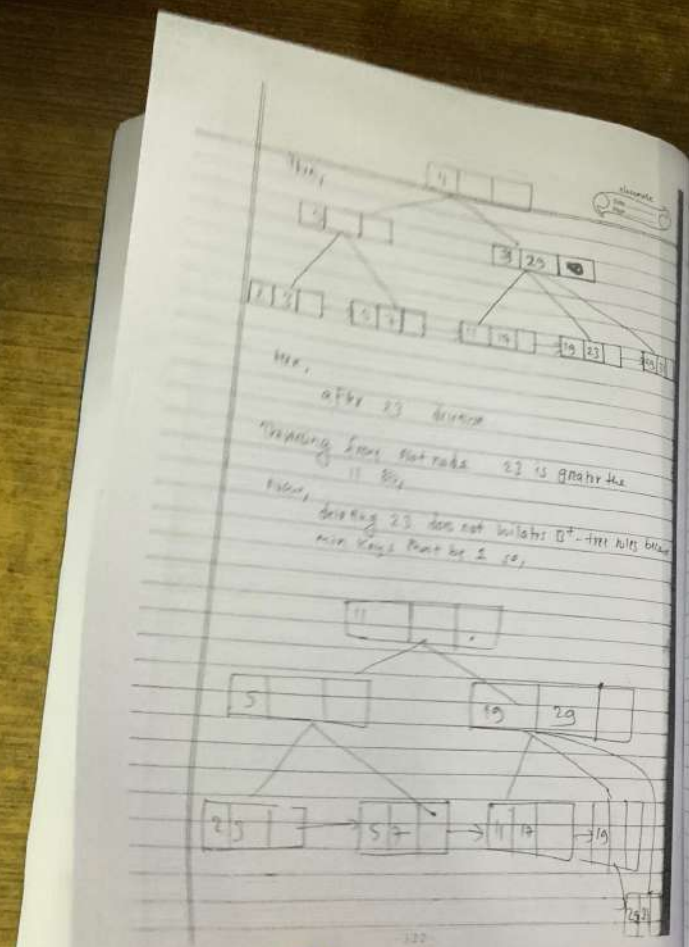
Second node is full so break down the data in two segments

Step IV :-

Step V :-



Here non-leaf node is full so, break it and its partition hold cell value is $\left(\frac{N}{2} - 1\right) = \left(\frac{4}{2} - 1\right) = 1$



2017 Fall

Explain how records of a file are placed and organized into a secondary storage.

The records of a file are placed and organized into storage media by following ways.

1. Sequential files.

- The records in sequential file organization are stored in sequence. A sequence means the records are stored one after the other.

- The records can be retrieved only in the sequence in which they were stored. The practical storage media for sequential files is magnetic tape.

- The major disadvantage of sequential access is that it is very slow. If the last record is to be retrieved, all preceding records are read before reaching the last record.

2. Direct or random files

- The records in direct file organization are not stored in a particular sequence. A key value of a record is used to determine the location to store the record. Each record is accessed directly without going through the preceding records.

- The file organization is suitable for storing data on disks. Direct file organization is much faster than sequential file organization for finding a specific record.

- A problem may occur in this type of files known as synonym. The problem occurs if the same address is calculated to store two or more records.

iii) By Indexed sequential files.

- In indexed sequential file organization, records are stored in ascending or descending order.
- The order is based on a value called key. Additionally, indexed file organization maintains an index in a file.
- An index consists of key values and the corresponding disk address for each record in the file.
- Index refers to the place on a disk where a record is stored.
- The index file is updated whenever a record is added or deleted from the file.
- The records in indexed file organization can be accessed in sequential access or direct access.
- The records in this file type requires more space on storage media.
- This method is slower than direct file organization as it requires to perform an index search.
- ~~In this way, records of file are organized in a secondary~~

Heap File Organization

When a file is created using heap file organization, the OS allocates memory area to that file without any further accounting details.

- File records can be placed anywhere in that memory area.
- It is the responsibility of the software to manage the records.
- Heap File does not support any ordering, sequencing, or indexing on its own.

v) Hash File organization

Hash File organization uses hash function computation on some fields of the records.

- The output of the hash function determines the location of disk block where the records are to be placed.

vi) Clustered File organization

- Clustered file organization is not considered good for large databases.
- In this mechanism, related records from one or more relations are kept in the same disk block, that is, the ordering of records is not based on primary key or search key.

Do this

2017 Spring

5b) Describe the structure of a B+ tree File organization.

- B+ tree file organization is the advanced method of an indexed sequential access method. It uses a tree-like structure to store records in File.
- It uses the same concept of key-index where the PK is used to sort the records. For each PK, the value of the index is generated and mapped with the record.
- The B+ tree is similar to a binary search tree (BST), but it can have more than two children. In this method, all the records are stored only at the leaf node. Intermediate nodes act as a pointer to the leaf nodes. They do not contain any records.

eg:-

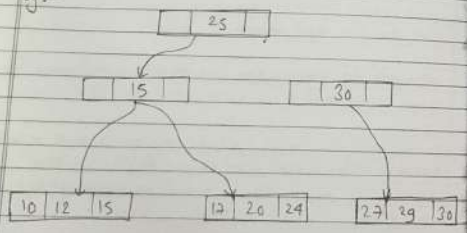


Fig:- B+ tree

326

The above B+ tree shows that:

- There is one root node of the tree, i.e. 25.
- There is an intermediary layer with nodes. They do not store the actual record. They only store the actual they have only pointers to the leaf nodes.
- The nodes to the left of the root node contain the prior value of the root and nodes to the right contain next value of the root. i.e, 15 and 30 respectively.
- There is only one leaf node which have one values i.e. 10, 12, 17, 20, 24, 27 and 29.
- Searching for any record is easier as all the leaf nodes are balanced.
- In this method, searching any record can be traversed through the single path and accessed easily.

→ pros of B+-tree

- In this method, searching becomes very easy as all the records are stored only in the leaf nodes and sorted the sequential linked list.
- Traversing through the tree structure is easier and faster.
- It is a balanced tree structure and any insert/update/delete does not affect the performance of tree.

327

- The size of the B+ tree has no restrictions, so the number of nodes can increase or decrease and the B+ tree structure can also grow or shrink.

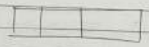
⇒ Cons of B+-tree

- This method is inefficient for the static method.

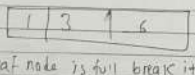
2018 Fall

5a) Construct a B+-tree for the following set of key values:
(1, 3, 6, 7, 11, 17, 19, 23, 30, 32). Assume that the tree is initially empty and values ~~are~~ added in ascending order.
Construct B+ tree with one node is Far. Also show tree after insertion of g.

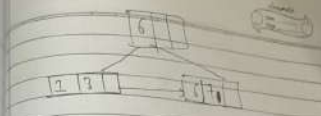
Step I :- empty node



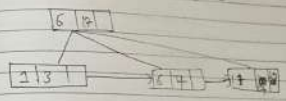
Step II :- Insert 1, 3, 6 then,



Now, the leaf node is full break it into two nodes.
The first partition holding ceil value of $(N-1)/2 = 3/2$
≈ 2 so, inserting 11 and keeping
Parent node the smallest among second node.

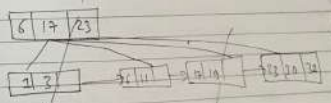


Step III inserting 17 then the leaf node is full so, again, break the node in two segments

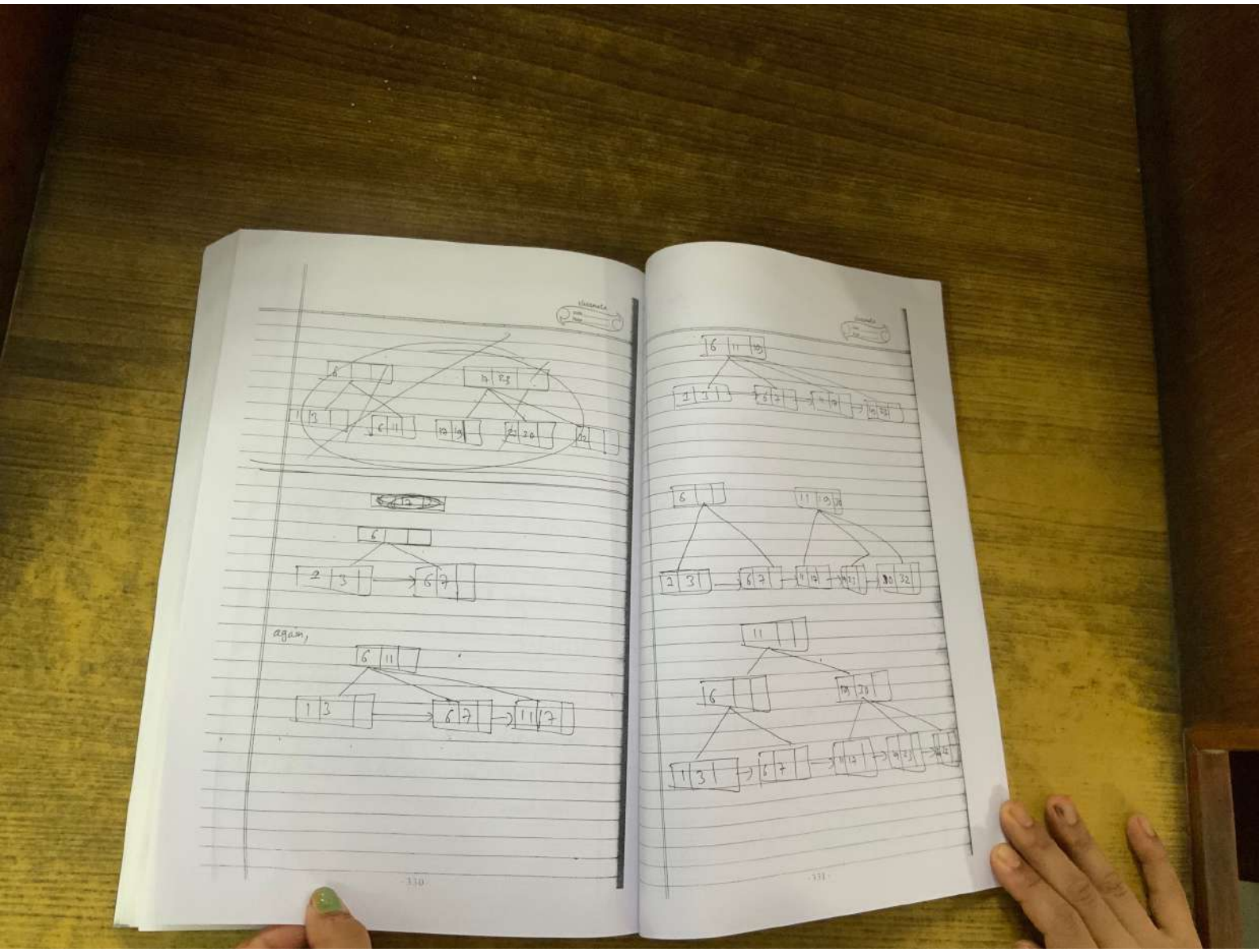


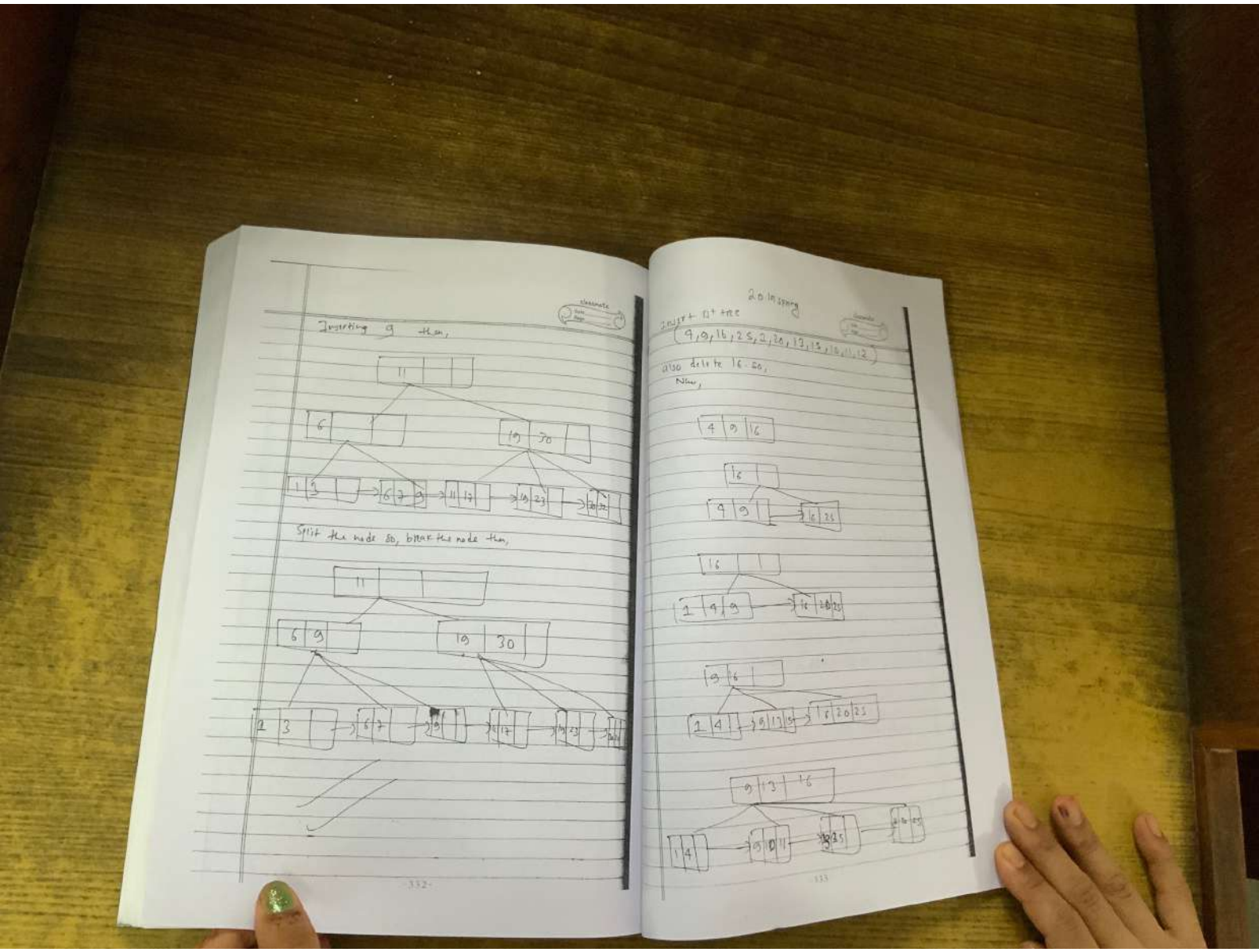
then,

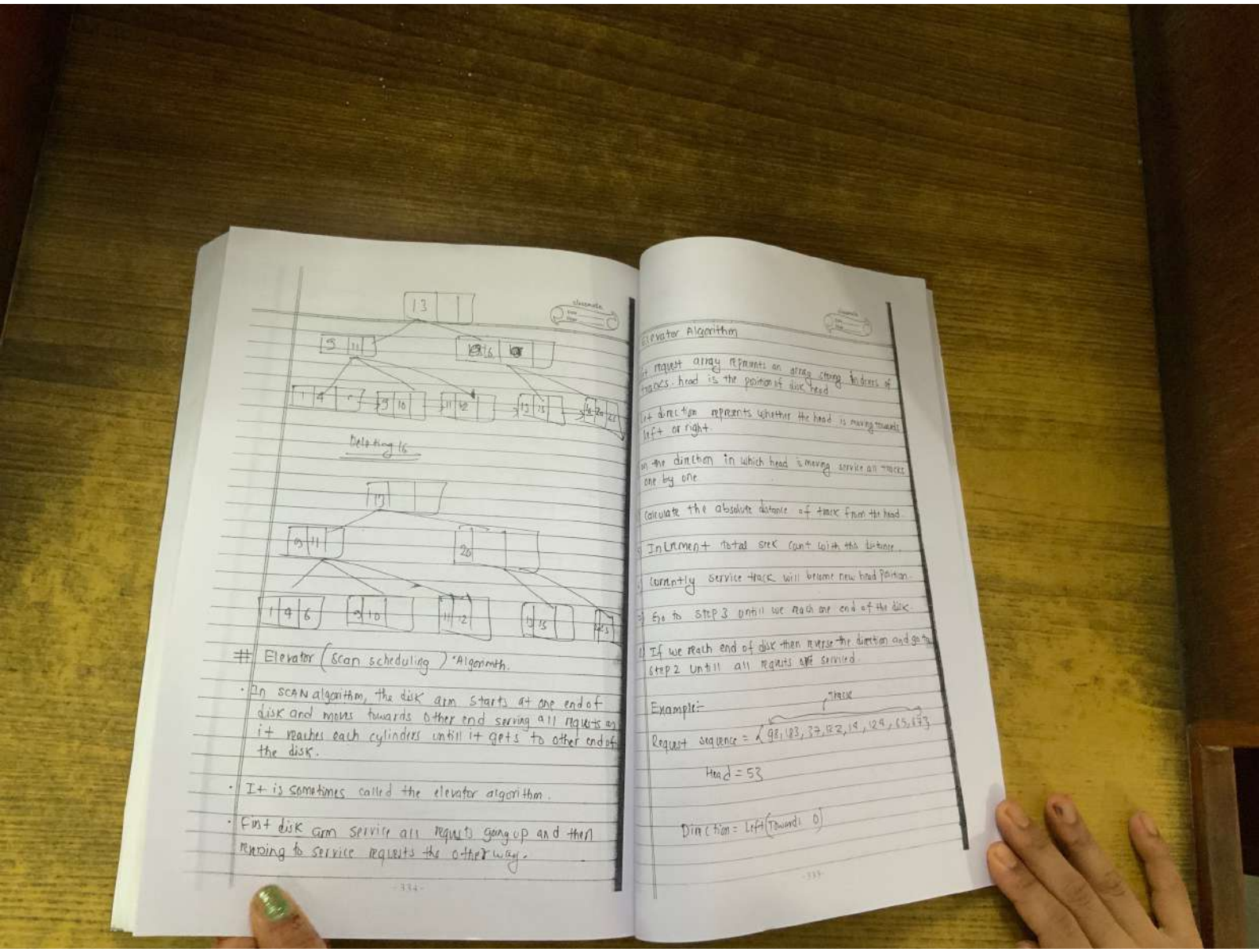
Step IV ~~inserting~~ breaking last internal then,



Here, parent node is also full so, we can say that
1st partition holds ceil value of $\left(\frac{N}{2}-1\right) = \frac{4}{2}-1 = 1$
then,







Elevator (scan scheduling) Algorithm.

- In SCAN algorithm, the disk arm starts at one end of disk and moves towards other end serving all requests as it reaches each cylinders until it gets to other end of the disk.
- It is sometimes called the elevator algorithm.
- First disk arm service all requests going up and then reversing to service requests the other way.

Elevator Algorithm

1. request array represents an array storing indexes of tracks. head is the position of disk head
2. direction represents whether the head is moving towards left or right.
3. in the direction in which head is moving service all tracks one by one
4. Calculate the absolute distance of track from the head
5. Increment total seek count with this distance.
6. Currently service track will become new head position.
7. Go to step 3 until we reach one end of the disk.
8. If we reach end of disk then reverse the direction and go to step 2 until all requests are serviced.

Example:-

Request sequence = { 98, 183, 37, 122, 14, 124, 65, 67 } (these)

Head = 53

Direction = Left (Towards 0)

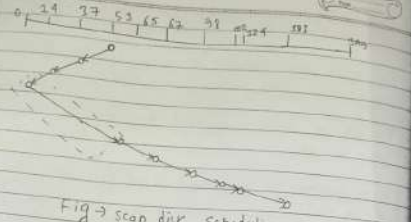


Fig → scan disk scheduling

Now,

Taking positive distance

$= (53-37) + (37-14) + (14-0) + (0+65) + (65+67) + (67+98) + (98+122) + (122+124) + (124+199)$

$= 236$ tracks.

Here there must be one improvement we should do

When we see extreme left head the portion of rectangle in dotted line it's undesirable movement the request is made after 65 only.

How does a mechanical Hard drives work:

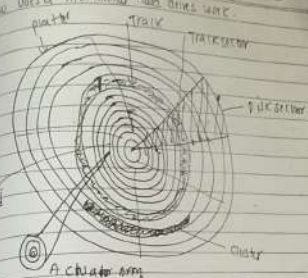


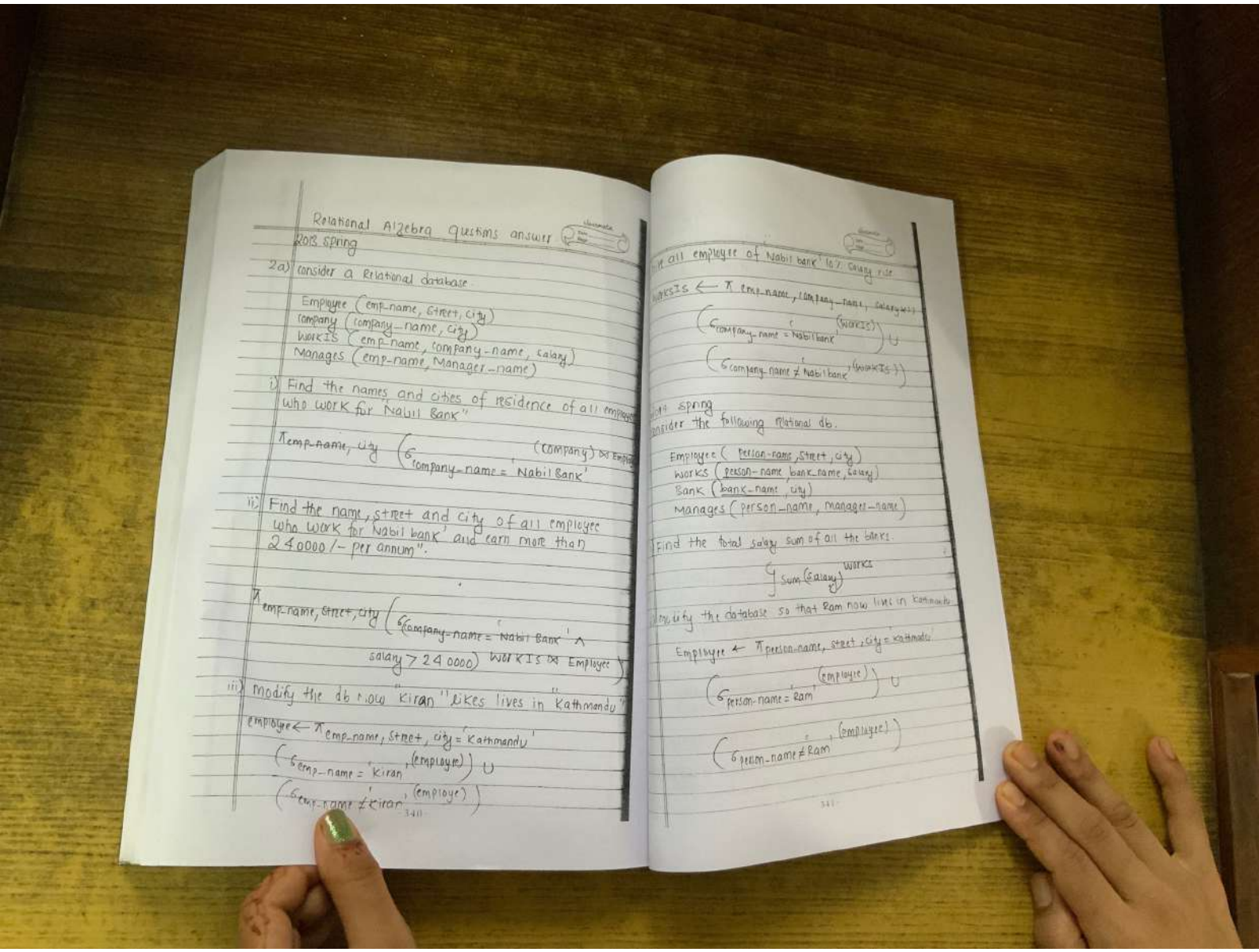
Fig:- simple Hard drive

When you save a document, it gets written somewhere "non-volatile" that keeps its state even when the power is off. How does that work for a hard drive?

- The hard drive contains a spinning platter with a thin magnetic coating.
- A "head" moves over the platter writing 0's and 1's as tiny areas of magnetic north or south on the platter.
- To read the data back, the head goes to the same spot, notices the north and south spots flying by, and so deduces the stored 0's and 1's.

- A modern hard drive can store well over a trillion 0/1 bits per platter, so the individual North/South spots are quite small.
- "Flash" storage is made with chips (no moving parts) and is gradually replacing spinning hard drives like this. Flash chips are what's inside camera SDHC memory cards and USB storage.

Chapter 11
Advance Database concepts



Relational Algebra questions answer

2015 spring

2a) consider a Relational database

Employee (emp_name, Street, city)
company (company_name, city)
workIS (emp_name, company_name, salary)
Manages (emp_name, Manager_name)

i) Find the names and cities of residence of all employees who work for Nabil Bank"

$\Pi_{emp\_name,\ city}\left(\sigma_{company\_name\ =\ 'Nabil\ Bank'}(company) \bowtie Empl\right)$

ii) Find the name, street and city of all employee who work for Nabil bank' and earn more than 240000/- per annum".

$\Pi_{emp\_name,\ street,\ city}\left(\sigma_{company\_name\ =\ 'Nabil\ Bank'\ \wedge\ salary > 240000}(workIS \bowtie Employee)\right)$

iii) modify the db i.ow "Kiran" likes lives in Kathmandu"

$Employee \leftarrow \Pi_{emp\_name,\ street,\ city\ =\ 'Kathmandu'}\left(\sigma_{emp\_name\ =\ 'Kiran'}(employee)\right) \cup \left(\sigma_{emp\_name\ \neq\ 'Kiran'}(employee)\right)$

all employee of Nabil bank' in 7% salary rise

$workIS \leftarrow \Pi_{emp\_name,\ company\_name,\ salary*1.1}\left(\sigma_{company\_name\ =\ 'Nabil\ Bank'}(workIS)\right) \cup \left(\sigma_{company\_name\ \neq\ 'Nabil\ bank'}(workIS)\right)$

spring

consider the following Relational db.

Employee (person_name, Street, city)
works (person_name, bank_name, salary)
Bank (bank_name, city)
Manages (person_name, manager_name)

Find the total salary sum of all the banks.

$sum(salary)^{works}$

Modify the database so that Ram now lives in Kathmandu

$Employee \leftarrow \Pi_{person\_name,\ street,\ city\ =\ 'kathmandu'}\left(\sigma_{person\_name\ =\ 'Ram'}(employee)\right) \cup \left(\sigma_{person\_name\ \neq\ 'Ram'}(employee)\right)$

iii) Find the street address, name cities of all employee who work for Nepal World bank corporation and earn more than $10,000 per annum.

$\Pi_{person\_name, street, city}(\sigma_{bank\_name = 'Nepal\ world\ bank' \wedge salary > 10000}(works \bowtie employee))$

iv) Delete all tuples in works relation for employee of small Bank corporation.

$works \leftarrow works - (\sigma_{bank\_name = 'small\ Bank\ corporation'}(works))$

2015 Fall

Consider a relational db. 2014 spring same question.

i) Find the names of all employees who wak for Nepal Rastra bank and salary greater than $10,000

$\Pi_{person\_name}(\sigma_{bank\_name = 'Nepal\ Rastra\ bank' \wedge salary > 10000}(employee \bowtie works))$

ii) Find names and cities of residence of all employees who work for Nepal Rastra Bank.

$\Pi_{person\_name, city}(\sigma_{bank\_name = 'Nepal\ Rastra\ bank'}(employee \bowtie works))$

142

nd the names of all employees in the db who live in same city as the company for which they work.

$\Pi_{person\_name}(\sigma_{employee.city = bank.city}((employee \bowtie works) \bowtie Bank))$

ive all the employee of Firstbank corporation a 1% salary rise.

$works \leftarrow \Pi_{person\_name, bank\_name, salary * 1.1}(\sigma_{bankname = 'firstbank'}(works)) \cup (\sigma_{bank\_name \neq firstbank}(works))$

2016 Fall

Student (CRN, Name, Gender, Address, Phone)
Course (courseID, courseName, Hour, TeacherID)
Teacher (TeacherID, TeacherName, office)
Registration (CRN, courseID, Date)

) count the number of student registerd subject in year 2015 gender wise.

Gender γ count(CRN) (Student)

143

ii) Show student details taught by teacher Rohit Shrestha

$\pi_{CRN, Name, Gender, Address, Telephone}$ ($\sigma_{TeacherName = 'Rohit Shrestha'}$ Teacher ⋈ Course ⋈ Registration ⋈ Student)

iii) Delete Student information taught by teacher N. Mathema.

Student ← Student − ($\sigma_{TeacherName = 'N. Metha'}$ (Teacher) ...)

Q. 2016 - Spring

Branch (branch-name, branch-city, assets)

Account (account-number, branch-name, balance)

Customer (customer-id, customer-name, customer-street, customer-city)

Depositor (customer-id, account-number)

loan (loan-number, branch-name, amount)

Borrow (customer-id, loan-number)

Write RA.

...nd all customer either account or loan.

$\pi_{customer\text{-}id}$ (Borrow) ∪ $\pi_{customer\text{-}id}$ (Depositor)

...t the name and city of customer who have their account at the branch location 'Butwal'.

$\pi_{customer\text{-}name, customer\text{-}city}$ ($\sigma_{customer\text{-}city = 'Butwal'}$ (customer))

Delete all account in the branch "B1"

Account ← Account − ($\sigma_{branch\text{-}name = 'B1'}$ (account))

Increase balance by 5% to all branches.

balance ← $\pi_{account\text{-}number, branch\text{-}name, balance * 1.05}$

Find all customer who have an account from atleast the 'Downtown' and the 'uptown' branches.

$\pi_{customer\text{-}id}$ ($\sigma_{branch\text{-}name = 'downtown'}$ ((account ⋈ loan) ⋈ Borrow))

2017 Fall

2a) Consider a relational Algebra.

employee (person-name, street, city)
works (person-name, company-name, salary)
company (company-name, city)
manages (person-name, manager-name)

a) Find the name of all employees who earns more than ~~manager~~ and works at small bank.

$\Pi_{\text{person-name}}(\text{works}) - (\Pi_{\text{works.person-name}}(\text{works} \bowtie (\text{works.salary} \leq \text{works2.salary} \wedge \text{works2.company-name} = \text{'small bank'})\ \rho_{\text{works2}}(\text{works})))$

b) Find the names of all employees who lives in the same city and on the same street as their manager.

$\Pi_{\text{person-name}}((\text{employee} \bowtie (\text{manager-name} = \text{employee2.person-name} \wedge \text{employee.street} = \text{employee2.street} \wedge \text{employee.city} = \text{employee2.city})\ [\rho_{\text{employee2}}(\text{employee})]))$

Find the names of all employees in the db who do not work for NBL company.

$\Pi_{\text{person-name}}(\sigma_{\text{company-name} \neq \text{'NBL company'}}(\text{works}))$

If question is, If people may not work for any company than

$\Pi_{\text{person-name}}(\text{employee}) - \Pi_{\text{person-name}}(\sigma_{(\text{company-name} = \text{'Fin+bank'})}(\text{works}))$

c) Find the names of all employees in the database who earns more than the top earner at 'BL company' in the db same as q no 6)

2018-Fall

Department (DepartmentID, DepartName)

Designation (DesignationID, DesignationName, Salary)

Employee (EmpID, EmpName, Gender, DesignationID, DepartmentID)

Allowance (AllowanceID, AllowanceName)

Allowance-Details (DetailID, EmpID, AllowanceID, Amount)

i) Find the number of employees department-wise.

$$\text{DepartName}\ \mathcal{G}_{\text{count}(\text{DepartmentID})}\ (\text{Department})$$

ii) List the employee details whose total salary is above Rs 50000

$$\pi_{\text{EmpID, Empname, Gender, DesignationID, DepartmentID}}\ (\sigma_{\text{Salary}>50000}\ (\text{Designation}) \bowtie \text{employee})$$

ii) the employee that who are getting house allowance.

$$\pi_{\text{empname}}\ (\sigma_{\text{Allowancename}='\text{house allowance}'}\ (\text{Allowance} \bowtie \text{Allowance-details} \bowtie \text{Employee}))$$

2018 Spring

Consider the relational database

Users (uid, cname, city)

Items (itemid, itemname, city, quantity, price)

Manager (mid, aname, city)

Query (queryno, uid, mid, itemid, query-details, hitratio)

Find all (queryno, uid) pairs for query with hitratio value greater than 500.

$$\pi_{\text{queryno, uid}}\ (\sigma_{\text{hitratio}>500}\ (\text{query}))$$

Find all item names of items in packers ordered with query-details as packers-details.

$$\pi_{\text{itemname}}\ (\sigma_{\text{city}=\text{packers}}\ (\text{query} \bowtie \text{Item}))$$

iii) Find itemids of items ordered through Manager 35 but not through manager 27.

$\pi_{itemid}(\sigma_{mid='35'}(query\ on\ Items)) \cup$

$\pi_{itemid}(\sigma_{mid \neq 27}(query\ on\ Items))$

Data Fail

b) Doctors (DoctorID, DoctorName, Department, Address, Salary)

patients (patientID, patientName, Address, Age, Gender)

Hospitals (patientID, DoctorID, HospitalName, location)

It's sql question.

2019 Spring

project (Project num, Project Name, Project type, Project Manager)

Employee (EmpName, Empnum)

Assigned-ID (Project num, Empnum)

Find Employee details working on a project name starts with 'L'.

$\pi_{EmpName, Empnum}(\sigma_{ProjectName\ like('L\%')}(Project\ no\ assigned\ to\ employee))$

List the employee details who are working under Project manager Rahan

$\pi_{Empnum, Empname}(\sigma_{Project\ Manager='Rahan'}(Project\ no\ assigned\ to\ Employee))$

iii) $\pi_{EMPNAME}(Employee) - \pi_{EMPNAME}(Employee\ assigned\ to\ project)$

List the employees who are still not assigned with any project.

iv) $\pi_{Empname, empno}(\sigma_{Employee.Empnum = Assigned.empnum}(Employee \times (Assigned\ to\ project) \div \pi_{Project}(Assigned\ to\ project)))$

=> List emp details who works for more than one project.

Chapter - 9,10 Crash Recovery, Transaction Processing and concurrency control.

Q) What is remote backup system? Describe Paging shadow page recovery. Why is this recovery technique called no undo/no redo technique.

Ans:-

Remote backup system

- Remote backup system is a mechanism where every bit of real-time data is backed up simultaneously at two distant places. one of them is directly connected to the system and the other one is kept at remote place at them as backup.
- Remote backup system provides high degree of availability, allowing transaction processing to continue even if the primary site is destroyed by fire, flood or earthquake.
- Data and log records from a primary sites are continously backup to remote backup site. If the primary site fails, the remote backup sites takes over transaction processing, after executing certain recovery actions.
- Remote backup provides a sense of security in case the primary location where the database is located gets destroyed.
- Remote backup can be offline or real-time or online. In case it is offline it is maintained manually.

Working principle of Remote backup system

- The remote backup site is stored at different state or point to prevent the damage on the remote backup site when the primary site is damaged.
- The primary site or local server is connected to the backup site through internet connection (WAN/LAN)
- The remote site is kept synchronized with primary site, as updates are performed at primary site. The synchronization is achieved by sending all log records from primary site to the remote backup site.
- When the primary site fails, the remote backup site takes over processing.

Architecture of Remote backup system

- Primary site :- The primary site are used in traditional-transactional processing system such as centralized or client-server systems. These system are those that run on a single computer system and do not interact with other systems.
- Network :- Remote backup software primarily bridges a connection between local network/IT environments with a remote backup location. The backup data is copied on a scheduled or specific routine by the backup software and is transferred and copied to the remote location over WAN, Internet or VPN connection.

- Backup :- A backup, or data backup is a copy of computer data taken and stored elsewhere so that it may be used to restore the original data after a data loss event. A backup system contains at least one copy of all data after its loss from data deletion or corruption, or to recover data from an earlier time. Backups provide a simple form of disaster recovery.
- Log Records :- Log records contains old values and new values for all update data items. The new values are used in case the updates need to be redone after system crash. The old values are used to roll back the updates of the transaction if the transactions aborted during normal operations, as well as to roll back the updates of the transaction in case the system crashed before the transaction committed.

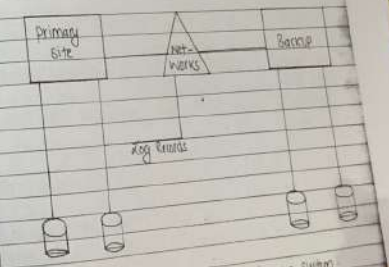


Figure :- Architecture of Remote backup system.

Example of Remote backup system.

- Subscription backup services provided by commercial data centers
- Backup to an offsite FTP server over the Internet.

Advantages of Remote backup system.

- It provides a sense of security incase the primary location where database is located gets damaged.
- Manage and secure digital data information.
- Remote access to backed up data from any location.
- It can recover even if all data at the primary site are lost.
- It can be offline or real-time or online. In case it is offline it is maintained manually.

Disadvantages of Remote backup system.

- Depending on the available network bandwidth, the restoration of data can be slow. Because data is stored offsite, the data may be recovered either via the internet or via a disk shipped from the online backup service provider.
- Some backup service providers have no guarantee that stored data will be kept private.
- If the encryption password is lost, data recovery will be impossible, However, with managed services this should not be a problem.
- It is possible that a remote backup service provider could go out of business or be purchased, which may affect the accessibility of one's data or the cost to continue using the service.

-330-

Shadow Paging and it's technique

- It is an alternative to log based recovery.
- It is a recovery technique that is used to recover database. In this technique, database is considered as made up of fixed size of logical units of storage which are referred as pages.
- pages are mapped into physical blocks of storage, with help of the page table which allows one entry for each logical page of database.
- This method uses two page tables named current page table and shadow page table.
- To start with, both the page tables are identical
  - only current page table is used for data item accesses during execution of the transaction.
- Whenever any page is about to be written for first time
  - A copy of this page is made onto an unused page.
  - The current page table is then made to point to the copy
  - The update is performed on the copy
- To commit a transaction
  - Flush all modified pages in main Memory to disk.
  - Output current page table to disk.
  - Make the current page table the new shadow page table as follows:
    - Keep a pointer to shadow page table at a fixed known location on disk.
    - Simply update the pointer to point to current page table on disk.

337

- once pointer to shadow page table has been written, transaction is committed.
- Non recovery is needed after a crash, new transactions can start right away using shadow page table.
- pages not pointed to from current/shadow page table should be free.

→ • Advantages of shadow paging over log based are :-
- No overhead of writing log records.
- Recovery is trivial. It makes undoing the effect of the executing transaction very simple.

→ • Disadvantages of shadow paging are :-
- coping the entire page table is very expensive.
  • can be reduced by using a page table structured like B+ tree. Here, no need to copy entire tree, only need to copy paths in the tree that lead to update leaf nodes.
- commit overhead is high even with above extension. need to flush every updated page and page table.
- Data gets fragmented. (related pages gets seperated on disk)
- After every transaction completion old version of modified data need to be garbage collected.
- hard to extend algorithm to allow transactions to run concurrently.

• Shadow Paging

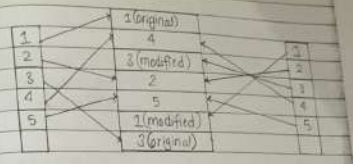


Fig:- Shadow Paging

⇒ Since, the recovery do not involves the undo function to reverse the mistake, such as deleting the wrong word in a sentence and it also does not involves redo function to restore any actions that are previously undo using the function called undo. so, we can say that shadow paging is called as no redo/no undo techniques.

6b) What is serializability? What are concurrency control protocols with it's benifits? Also Explain the problems occurred with concurrency?

Ans:-

18/5/96 29

6a) Serializability and Serializable Schedule

- Serializability is a concept that helps us to check which schedules are serializability.
- When multiple transactions are running concurrently then there is a possibility that the database may be left in an inconsistent state. A serializable schedule is the one that always leaves the database in consistent state.
- Serializability in DBMS decides if an interleaved non-serial schedule is serializable or not.
- A serial schedule is always a serializable schedule because in serial schedule, a transaction only starts when the other transaction finished execution. However a non-serial schedule needs to be checked for serializability.
- A non-serial schedule of 'n' number of transactions is said to be serializable schedule, if it is equivalent to the serial schedule of those 'n' transactions executes at a time and the other starts when the already running transactions finished.
- eg of serializability:- consider 2 schedules; schedule 1 and 2.

Schedule 1

Transaction 1	Transaction 2
• A1	
• A2	• B1
	• B2

Schedule 2

Transaction 1	Transaction 2
• $A1_0$	• B_0
• $A2_0$	• $B(A)$

18/5/96 30

Here, in schedule-2, 1, 2 is an order of transactions for the schedule2 that portrays an interleaved execution.

In the given above examples:

- Schedule 1 is a serial schedule consisting of Transaction1 and Transaction2 where the operation on data items A (A1 and A2) are performed first and later the operations on data item B (B1 and B2) are carried out serially.
- schedule 2 is a non-serial schedule consisting of Transaction1 and Transaction2 where the operations are interleaved.
- In the given scenario, schedule2 is serializable if the output obtained from both schedule2 and Schedule1 are equivalent to one another. In a nutshell, a transaction within a given non-serial schedule is serializable if its outcome is equivalent to the outcome of the same transaction when executed serially.

Rules of Serializability:-

- ordering of read/write is important.
- If two transactions only read data item they do not conflict and order is not important.
- If two transactions either read or write completely separate data items, they do not conflict and order is not important.
- If one transaction write a data item and another reads or writes same data item, order of execution is important.

Types of Serializability:-

18/5/96 81

1) Conflict Serializability :-

- Instructions I_i and I_j of transactions T_i and T_j respectively conflicts:
  - If and only if there exists same item Q accessed by both I_i and I_j and at least one of these instructions wants to write Q.
- If the schedule S can be transformed into a schedule S' by a series of swapping of non-conflicting instructions, then S and S' are conflict equivalent.
- A schedule S is conflict serializable if it is conflict equivalent to a serial schedule.

Instructions I_i	Instructions I_j	Result
Read(Q)	Read(Q)	No conflict
Read(Q)	Write(Q)	conflict
Write(Q)	Read(Q)	conflict
Write(Q)	Write(Q)	conflict

- eg 1 :- Let we have schedule A and schedule B from

Fig- Schedule A

T_1	T_2
Read(A)	
Write(A)	
	Read(A)
	Write(A)
Read(B)	
Write(B)	
	Read(B)
	Write(B)

Fig- Schedule B

T_1	T_2
Read(A)	
Write(A)	
Read(B)	
Write(B)	
	Read(A)
	Write(A)
	Read(B)
	Write(B)

18/5/96 82

Here, schedule A can be transformed into serial schedule B by series of swaps of non conflict instructions.

Hence, schedule A is called conflict serializable

- eg2 :- Let we have schedule x and schedule y as follows

Fig- schedule x

T_1	T_2
Read(a)	
	Write(a)
Write(a)	

Fig- schedule y

T_1	T_2
Read(a)	
Write(a)	
	Write(a)

Here, we can't transform schedule x into schedule y by swapping of non conflict instructions.

Hence, schedule x is non conflict serializable

ii) View serializability

- Two schedule are said to be view equivalent if following three conditions held :-
  - For each data item Q, if transaction T_i reads the initial value of Q in schedule S then transaction T_i in schedule S' must also read initial value of Q

18/5/95 33

- For each data item Q, if T_i executes read(Q) in S and the value was produced by write(Q) of T_j then read(Q) of T_i in S' must also read values of Q produced by same write(Q) if T_j.
- For each data item Q, the transaction that performs the final write(Q) operation in S' must perform the final write(Q) operation in S.

- A schedule 'S' is view serializable if it is view equivalent to a serial schedule.

- E.g.1 Let we have schedule A and schedule B as follows

T1	T2	T3
Read(Q)		
	Write(Q)	
Write(Q)		Write(Q)

Fig:- Schedule A

T1	T2	T3
Read(Q)		
Write(Q)		
	Write(Q)	
		Write(Q)

Fig:- Schedule B

364

18/5/95 34

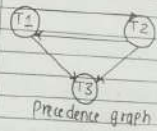
- Here, schedule A is view serializable schedule because it is view equivalent to serial schedule B, as transaction T3 performs Read(Q) i.e. reads the initial value of Q in both schedules and also,
- Here, write(Q) of T2 and T3 are called blind writes because it is performed without having performed Read(Q).
- Every conflict serializable schedule is also view serializable but not vice-versa.

Testing for Serializability:-

- Construct a directed graph called precedence graph from S
- Directed Graph consists of a pair G=(V,E) where,
  - V is set of vertices; it consist of all transaction in the schedule
  - E is a set of edges. Set of edges consists of all edges $T_i \rightarrow T_j$ for which one of three conditions holds
    - * T_i executes write(Q) before T_j executes read(Q).
    - * T_i executes read(Q) before T_j executes write(Q).
    - * T_i executes write(Q) before T_j executes write(Q).
- If an edge $T_i \rightarrow T_j$ exists in precedence graph, then in any serial schedule S' equivalent to S, T_i must appear before T_j.
- If the precedence graph has a cycle then the schedule is not conflict serializable.
- If the precedence graph has not any cycle then schedule is conflict serializable.

eg 1

T1	T2	T3
Read(A)		
	write(A)	
write(A)		
		write(A)

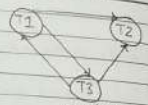


Precedence graph

Here, as graph contains cycle, above schedule is non conflict sertalizable.

eg 2

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



Precedence graph

Here, as graph contains cycle, above schedule is not conflict serilizable.

some advantages of serializability are:-

- It helps to preserve the consistency and concurrency of a database.
- It helps to obtain an equivalent output as of a serial schedule for the same 'n' number of transactions.
- It is a classical concurrency schema which assumes that all accesses to the database are done using read and write operations also it helps that the schedules are serializable or not.

10/12/18 Page 25

concurrency control protocols with its importance in transaction

- concurrency control is a process of managing simultaneous operations without conflicting with each other.
- It ensures that database transactions are performed concurrently and accurately to produce correct results without violating data integrity of the respectively.
- It is used to address such conflicts, which mostly occur with a multi-user system. Therefore it is the most important element for proper functioning of a database where two or more database transactions are executed simultaneously, which require access to the same data.
- The concurrency control protocols ensures the atomicity, consistency, isolation, durability and serializability of the concurrent execution of the database transactions.
- Therefore, these protocols are categorized as:-

i) Lock Based concurrency control protocol

ii) Time stamp concurrency control protocol.

iii) Validation Based concurrency control protocol.

10/12/21 Page 26

i) Lock Based concurrency control protocol.

- In this type of protocol, any transaction cannot read or write data until it acquires an appropriate lock on it. There are two types of lock:-

i) Shared lock:-

- It is also known as a Read-only lock. In a shared lock data item can only read by the transaction.
- It can be shared between the transactions because when the transaction holds a lock, then it can't update the data on the data item.

ii) Exclusive lock.

- In the exclusive lock, the data item can be both reads as well as written by the transactions.
- This lock is exclusive, and in this lock, multiple transactions do not modify the same data simultaneously.

There are four types of lock protocols available:-

1) Simplistic lock protocol

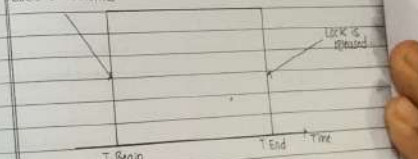
- It is the simplest way of locking the data while transaction. Simplistic lock-based protocols allow all the transactions to get the lock on the data before insert or delete or update on it.

- It will unlock the data item after completing the transaction.

10/12/21 Page 27

2) pre-claiming lock protocol

- Pre-claiming lock protocols evaluate the transaction to list out the data items on which they need locks.
- Before initiating an execution of the transaction, it requests DBMS for all the lock on all those data items.
- If all the locks are granted then this protocol allows the transaction to begin. When the transaction is completed then it releases all the lock.
- If all the locks are not granted then this protocol allows the transaction to roll back and wait until all the locks are granted.

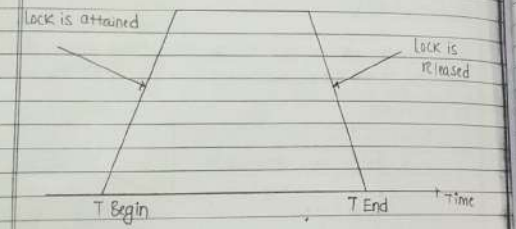


3) Two-phase locking (2PL)

- The two-phase locking protocol divides the execution phase of the transaction into three parts.

10/12/18 Page 28

- In the first part, when the execution of the transaction starts, it seeks permission for the lock it requires.
- In the second part, the transaction acquires all the locks. The third phase is started as soon as the transaction releases its first lock.
- In the third phase, the transaction cannot demand any new locks. It only releases the acquired locks.



There are two phases of 2PL:-

i) Growing phase:- In the growing phase, a new lock on the data item may be acquired by the transaction, but none can be released.

ii) Shrinking phase:- In the shrinking phase, existing lock held by the transaction may be released, but no new locks can be acquired.

10/12/18 Page 29

In the below example, if lock conversion is allowed then the following phase can happen

a) Upgrading of lock (from S(a) to X(a)) is allowed in growing phase

b) Downgrading of lock (from X(a) to S(a)) must be done in Shrinking Phase.

eg:-

	T1	T2
0	Lock-S(A)	
1		Lock-S(B)
2	Lock-X(B)	
3	——	——
4	Unlock(A)	
5		Lock-X(C)
6	Unlock(B)	
7		Unlock(A)
8		Unlock(C)
9	——	——

The following way shows how unlocking and locking work with 2PL

Transaction (T1):-

- Growing Phase: From step 2-3
- Shrinking Phase: From step 5-7
- Lock point: at 3

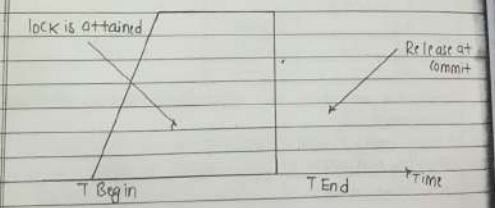
Transaction (T2):-

10/12/2018 Page : 30

- Growing phase : from step 2-6
- shrinking Phase : from step 8-9
- Lock point : at 6

4) Strict Two-Phase locking (Strict-2PL)

- The first phase of Strict-2PL is similar to 2PL. In the first phase, after acquiring all the locks, the transaction continues to execute normally.
- The only difference between 2PL and Strict 2PL is that Strict 2PL does not release a lock after using it.
- Strict-2PL waits until the whole transaction to commit, and then it releases all the locks at a time.
- Strict-2PL protocol does not have shrinking Phase of lock release.



- It does not have cascading abort as 2PL does.

10/12/2018 Page : 31

Timestamp ordering Protocol:

- The Timestamp Ordering protocol is used to order the transactions based on their Timestamps. The order of transaction is nothing but the ascending order of the transaction creation.
- The priority of the older transaction is higher that's why it executes first. To determine the Timestamp of the transaction, this Protocol uses System time or logical counter.
- The lock-based protocol is used to manage the order between conflicting pairs among transactions at the execution time. But Timestamp based protocols start working as soon as a transaction is created.
- Let's assume there are two transactions T1 and T2. Suppose the transaction T1 has entered the system at 007 times and transaction T2 has entered the system at 009 times. T1 has the higher priority, so it executes first as it is entered the system first.
- The timestamp ordering protocol also maintains the timestamp of last 'Read' and 'Write' operation on a data.

Basic Timestamp ordering protocol works as follows -

(a) Check the following condition whenever a transaction T_i issues a Read(x) operation -

10/12/21 page 82

- If W_TS(x) > TS(T_i) then operation is rejected.
- If W_TS(x) <= TS(T_i) then operation is executed.
- Timestamps of all the data items are updated.

b) check the following condition whenever a transaction (T_i) issues a write(x) operations:-

- If TS(T_i) < R_TS(x) then operation is rejected.
- If TS(T_i) < W_TS(x) then the operation is rejected and T_i is rolled back otherwise the operation is executed.

Where,

- TS(T_i) denotes the Timestamp of the transaction T_i.
- R_TS(x) denotes the Read time-stamp of data-item x.
- W_TS(x) denotes the write time-Stamp of data-item x.

Advantages and disadvantages of TO protocols:

- TO protocol ensures serializability since the precedence graph is as follows:

[Transaction with smaller TS] → [Transaction for with larger TS]

Image:- Precedence Graph for TS ordering.

- TS protocol ensures freedom from dead lock that means no transaction ever waits.
- But the schedule may not be recoverable and may not even be cascade-free.

10/12/21 page 83

3) Validation Based Concurrency protocol:

Validation Phase is also known as optimistic concurrency control technique. In the validation based protocol, the transaction is executed in the following three phases:

a) Read Phase:-

- In this phase, the transaction(T) is read and executed.
- It is used to read the values of various data items and stores them in temporary local variables.
- It can perform all the write operations on temporary variables without an update to the actual database.

b) Validation Phase:-

- In this phase, the temporary variable value will be validated against the actual data to see if it violates the serializability.

c) Write Phase:-

- If the validation of the transaction is validated, then the temporary results are written to the database or system otherwise the transaction is rolled back.

Here, each Phase has the following different timestamps:-

10/12/18 page 34

Start (T_i) :- It contains the time when T_i started its execution.

Validation (T_i) : It contains the time when (T_i) finishes its read phase and starts its validation phase.

Finish (T_i) : It contains the time when T_i finishes its write phase.

- This protocol is used to determine the time stamp for the transaction for serialization using the time stamp of the validation phase, as it is the actual phase which determines if the transaction will commit or rollback.
- Hence TS(T) = validation(T).
- The serializability is determined during the validation process. It can't be decided in advance.
- While executing the transaction, it ensures a greater degree of concurrency and also less number of conflicts.
- Thus it contains transactions which have less number of rollbacks.

Benifits/advantages of concurrency. Importance of it :-

- Reduced wasting time response time or turn around time.
- Increased throughput or resource utilization.
- If we run only one transaction at a time than the acid property is sufficient but it is possible that when multiple transactions are

10/12/18 page 35

executed concurrently then database may become inconsistent

- overlapping with the input-output activity with CPU also makes the response time better.
- But interleaving of instruction between transaction may also lead to many problems due to which concurrency control is required.

Drawbacks of concurrency

- It is required to protect multiple applications from one another.
- Additional performance overheads and complexities in OS are required for switching among applications.
- It is required to coordinate multiple applications through additional mechanisms.
- Sometimes running too many applications concurrently leads to severely degraded performance.

Problems due to concurrency

There are many which may occur due to concurrency.

1) Dirty read problems

- If a transaction reads an uncommitted temporary value written by some other transaction than it is called dirty read problem.
- In this one transaction read item updated by another uncommitted transaction that may be future be aborted or failed.
- In such cases, the read value disappears from the database upon abort this is turned on dirty read the reading transaction end with incorrect results.

10/10/18 Page 36

eg:-

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

The values of item X which is read by T2 is called dirty read because this data can be created by a transactions that has not been committed yet.

2) Loss update problem / Write-write problem.

- This problem occur when two transactions access the same data item and have their operations interleaved in a way that makes the value of some database items incorrect.
- If there are two write operations of the different transaction in same data values and in between them there are no read operations then the second write over the first.

Consider a schedule below:-

eg:-

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

- Here, is a blind write that means write without a read. Here the changes made by transaction T1 are lost which is updated by a transaction T2.

10/10/18 Page 37

3) Unrepeatable and phantom read problem

- When a transaction cannot repeat the read operations because the variable is edited the variable by some other transactions then this problem is called Unrepeatable read problem. In this problem at different instances of time a transaction read gives different values it is because data item might have been updated by another transaction.
- This causes a problem while execution of some aggregate by a transaction and due to changes in the values of the data item by another transaction it leads to incorrect results when a transactions read values of data twice and another transaction's updates data item in between then the results of two read operations will differ.

eg:-

T1	T2
R(A)	
	R(A)
Delete(A)	
	R(A)

4) Incorrect summary problem

- When one of the transactions is checking on aggregate summary function while other transactions are updating then this problem is called incorrect summary problem.
- The aggregate functions may calculate some values before they updated and others after they are updated.

5b) What is data dictionery storage ? Mention the information's which are kept in it by defining file categories.

Data dictionery Storage

- Storing the relational schemas and other metadata about the relations in a structure is known as Data dictionery or system catalog.
- A data dictionery is like the A-Z dictionery of the relational database system holding all information of each relation in the database.

→ The types of information a system must store are :-

- Name of the relations.
- Name of the attributes of each relation.
- Lengths and domains of attributes.
- Name and definations of the views defined on the database.
- Various integrity constraints.

→ With this, the system also keeps the following data based on users of the system :-

- Name of authorized users.
- Accounting and authorization information about users.
- The authentication information for users, such as passwords or other related information.

→ In addition to this, the system may also store some statistical and descriptive data about the relations, such as :-

- Number of tuples in each relation.
- Method of storage for each relation, such as clustered or non-clustered.

→ A system may also store the storage organization, whether sequential, hash, or heap. It also notes the location where each relation is stored :-

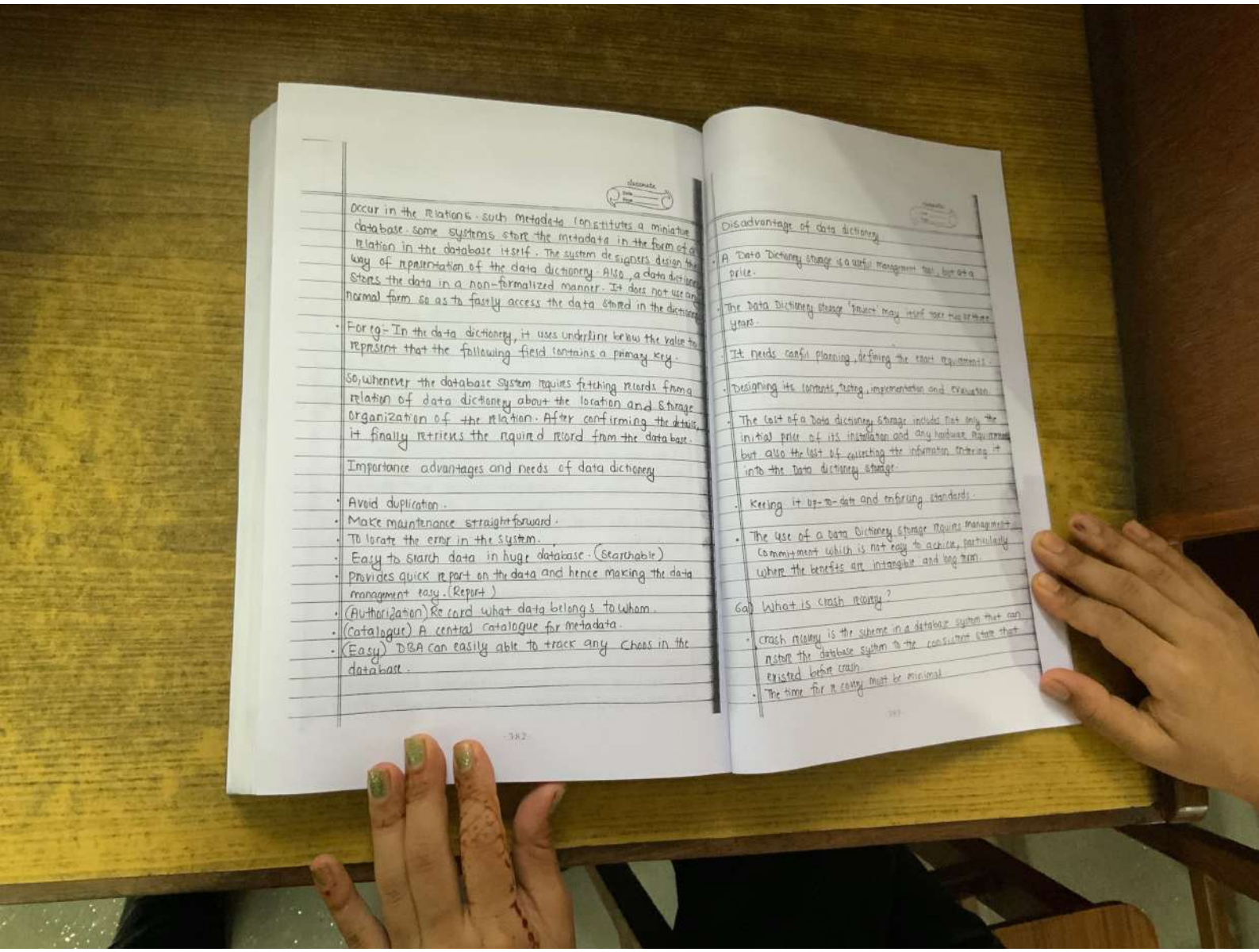
- If relations are stored in the files of the OS, the data dictionery note, and stores the names of file.

- If the database stores all the relations in a single file, the data dictionery notes and store the blocks of containing records of each relation in a data structure similar to a linked list.

→ At last, it also stores the information regarding each index of all the relations :-

- Name of the index.
- Name of the relation being indexed.
- Attributes on which the index is defined.
- The type of index formed.

⇒ All the above information or metadata is stored in a data dictionery. The data dictionery also maintains updated information whenever stored.



occur in the relations. such metadata constitutes a miniature database. some systems store the metadata in the form of a relation in the database itself. The system designers design the way of representation of the data dictionary. Also, a data dictionary stores the data in a non-formalized manner. It does not use any normal form so as to fastly access the data stored in the dictionary.

- For eg:- In the data dictionary, it uses underline below the value to represent that the following field contains a primary key.

So, whenever the database system requires fetching records from a relation of data dictionary about the location and storage organization of the relation. After confirming the details, it finally retrieves the required record from the database.

Importance advantages and needs of data dictionary

- Avoid duplication.
- Make maintenance straightforward.
- To locate the error in the system.
- Easy to search data in huge database. (Searchable)
- provides quick report on the data and hence making the data management easy. (Report)
- (Authorization) Record what data belongs to whom.
- (catalogue) A central catalogue for metadata.
- (Easy) DBA can easily able to track any chaos in the database.

Disadvantage of data dictionary

- A Data Dictionary storage is a useful management tool, but at a price.
- The Data Dictionary storage 'project' may itself take many years.
- It needs careful planning, defining the exact requirements.
- Designing its contents, testing, implementation and evaluation.
- The cost of a Data dictionary storage includes not only the initial price of its installation and any hardware requirements but also the cost of converting the information existing in into the Data dictionary storage.
- Keeping it up-to-date and enforcing standards.
- The use of a Data Dictionary storage requires management commitment which is not easy to achieve, particularly where the benefits are intangible and long term.

Q2) What is crash recovery?

- Crash recovery is the scheme in a database system that can restore the database system to the consistent state that existed before crash.
- The time for recovery must be minimal.

- This is done by rolling back incomplete transactions and completing committed transactions that were still in memory when the crash occurred in the below figure.

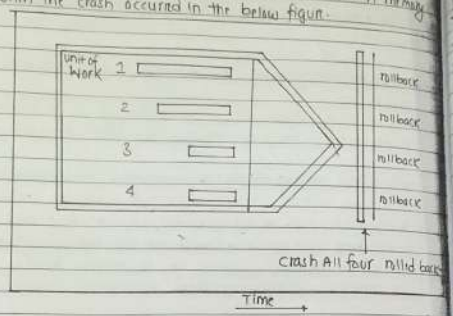


Fig :- Rolling back units of work (crash recovery)

conditions that can necessitate a crash recovery include:-

- A power failure on the machine, causing the database manager and the database partitions on it to go down.
- A hardware failure such as memory, disk, cpu, or network failure.
- A serious operating system error that causes the Db2® instance to end abnormally.

Importance of crash recovery

- Needed for achieving atomicity and durability
  - need to abort transactions or restart them
  - Need to recover from crashes
- crash recovery algorithms had major impact beyond database
  - Algorithms are interesting in their own right
- Logging for crash recovery has significant impact on DBMS performance.

Fig:- Taxonomy of crash recovery algorithms

6b) What is transaction?

- A transaction is a single logical unit of work which accesses and possibly modifies the contents of a database.
- Transaction access data using read and write operations.
- In order to maintain consistency in a database, before and after the transaction, certain properties are followed.

→ Types of Transactions:

* Based on Application areas
- Non-distributed vs. distributed.
- Compensating transactions.
- Transactions Timing.
- On-line vs. batch.

* Based on Actions
- Two-step
- Restricted
- Action model.

* Based on structure
- Flat or simple transactions: It consists of a sequence of primitive operations executed between a begin and end operations.
- Nested transactions: A transaction that contains other transactions

(work flow)

Q.No 5 10/12/21 Page 22

Q.No 5

In order to maintain consistency in database, before and after transactions the acid properties are followed- ACID properties.

ACID is a set of properties of database transactions intended to guarantee data validity despite errors, power failures and other mishaps. In the context of databases, a sequence of database operations that satisfies the ACID properties is called a transaction.

1) Atomicity:-

- By this, we mean that either the entire transaction takes place at once or doesn't happen at all. There is no midway i.e. transactions do not occur partially. Each transaction is considered as one unit and either runs to completion or is not executed at all.
- It involves the following two operations:
  - Abort: If a transaction aborts, changes made to database are not visible.
  - Commit: If a transaction commits changes made are visible.
- Atomicity is also known as the "All or Nothing Rule".

Consider the following transaction T consisting of T1 and T2: Transfer of 100 from account 'X' to account 'Y'.

10/2/2018 page 23

Before: X: 500	Y: 200
Transaction T	
T1	T2
Read(x)	Read(y)
x := x-100	y := y+100
Write(x)	Write(y)
After: X: 400	Y: 300

- If the transaction fails after completion of T1 but before completion of T2. (say, after write(x) but before write(y)), then amount has been deducted from x but not added to y. This results in an inconsistent database state. Therefore, the transaction must be executed in entirety in order to ensure correctness of database state.

ii) Consistency :-

- This means that integrity constraints must be maintained so that the database is consistent before and after the transaction.
- It refers to the correctness of a database.
- Referring to the example above, The total amount before and after the transaction must be maintained.
Total before T occurs = 500+200 = 700. Total after T occurs = 400+300 = 700, Therefore, database is consistent.
Inconsistency occurs in case T1 completes but T2 fails. As a result T is incomplete.

10/10/18 page 24

i) Isolation :-

- This property ensures that multiple transactions can occur concurrently without leading to the inconsistency of database state.
- This property ensures that the execution of transactions concurrently will result in a state that is equivalent to a state achieved these were executed serially in some order.
Let x = 500, y = 500
consider two transactions T and T''.

T	T''
Read(x)	Read(x)
x := x*100	Read(y)
Write(x)	z := x+y
Read(y)	Write(z)
y := y-50	
Write(y)	

Suppose T has been executed till Read(y) and then T'' starts. As a result, interleaving of operations takes place due to which T'' reads correct value of x but incorrect value of y and sum computed by T'': (x+y = 50,000+450 = 50,450) is thus not consistent with the sum at end of transaction T: (x+y = 50,000+450 - 50,450). This results in database inconsistency, due to a loss of 50 units. Hence transactions must take place in isolation and changes should be visible only after they have been made to the main memory.

10/2018 18546 Page 24

iv) Durability :-

- This property ensures that once the transaction has completed execution, the updates and modifications to the database are stored in and written to disk and they persist even if a system failure occurs.

- These updates now become permanent and are stored in non-volatile memory. The effects of the transactions, thus are never lost.

- The ACID properties, in totality, provide a mechanism to ensure correctness and consistency of a database in a way such that each transaction is a group of operations that acts as a single unit, produces consistent results, acts in isolation from other operations and updates that it makes are durably stored.

2012 Fall

5b) Describe about failure classification? Write down Differences between Deferred Database modification and Immediate Database modification.

To find that where the problem has occurred, we generalize a failure into the following categories.

i) Transaction failure :-

- The transaction failure occurs when it fails to execute or when it reaches a point from where it can't go any further.

- If a few transaction or process is hurt then this is called as transaction failure.

- Reasons for transaction failure could be -

a) Logical errors :- If a transaction cannot complete due to some code error or an internal error condition, then the logical error occurs.

b) Syntax error :- It occurs when the DBMS itself terminates an active transaction because the database system is not able to execute it. For eg:- The system aborts an active transaction in case of deadlock or resource unavailability.

ii) System crash

- System failure can occur due to power failure or other hardware or software failure. Example operating system error.

- Fail-stop assumption : In the system crash, non-volatile storage is assumed not to be corrupted.

iii) Disk Failure

- It occurs where hard-disk drives or storage drives used to fail frequently. It was a common problem in the early days of technology evolution.

- Disk failure occurs due to the formation of bad sectors, disk head crash, and unreachability to the disk or any other failure, which destroy all or part of disk storage.

Deferred Modification	Immediate Modification
• In deferred modification, the changes are not applied immediately to the database.	• In immediate modification, the changes are applied directly to the database.
• The log files contains all the changes that are to be applied to the database.	• The log file contains both old as well as new values.
• In this method once rollback is done all the records of log file are discarded and no changes are applied to the database.	• In this method once rollback is done the old values are restored into the database using the records of the log file.
• Concepts of buffering and caching are used in deferred modification method.	• Concept of shadow paging is used in immediate modification method.
• It is called NO-UNDO/REDO technique.	• It is also called UNDO/REDO technique.
• The major disadvantage of this method is that it requires a lot of time for recovery in case of system failure.	• The major disadvantages of this method is that there are frequent I/O operations while the transaction is active.
• The deferred modification technique occurs if the transaction does not modify the database until it has	• The immediate modification technique occurs if database modification occurs while the transaction is

Q) What do you mean by a schedule? When schedule is called serializable?

- A series of operation from one transaction to another transaction is known as schedule.
- It is used to preserve the order of the operation in each of the individual transaction.

Types of Schedule in DBMS

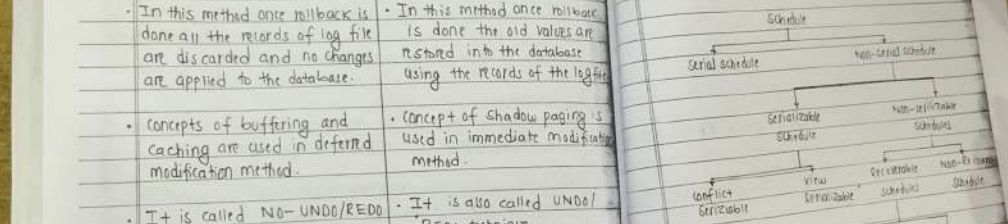


Fig:- Hierarchy of schedules

i) Serial schedules:-

- Schedules in which the transactions are executed non-interleaved, ie a serial schedule is one in which no transaction starts until a running transaction has ended are called serial schedules.

eg:- consider the following schedule involving two transactions T1 and T2.

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

Where, R(A) denotes that a read operation is performed on some data item A.

This is serial schedule since the transactions perform serially in the order T1 → T2

and R(B) denotes that a Read operation is performed on some data item A

similiarly,

W(A) denotes that a write operation is performed on some data A. and,

W(B) denotes that a write operation is performed on some data B.

-394-

Non-Serial schedule:

If interleaving of operations is allowed then there will be non-serial schedule.

- It contains many possible orders in which the system can execute the individual operations of the transactions.

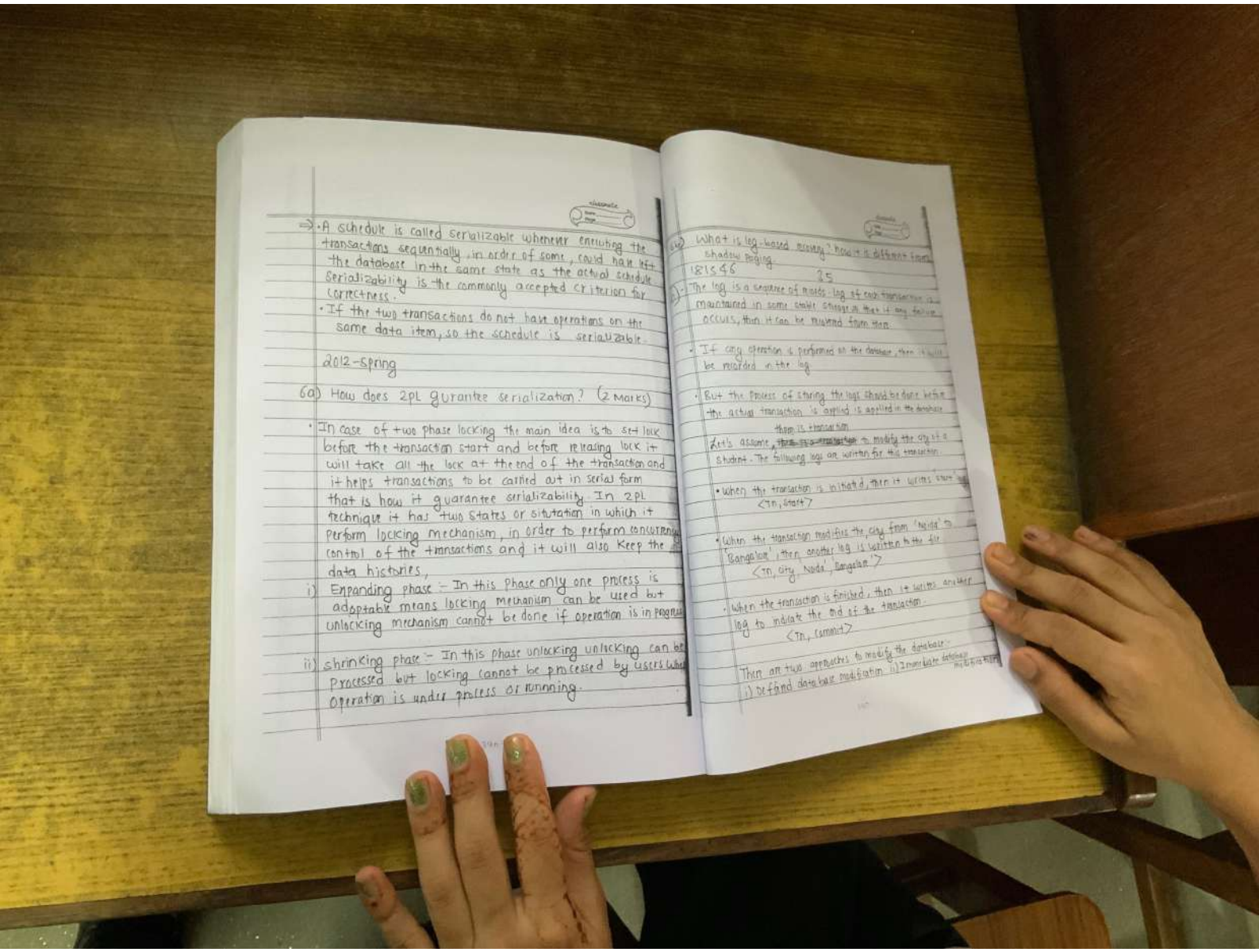
eg:- consider the following transaction T1 and T2 in schedule.

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

In this schedule

- There are two transaction T1 and T2 executing concurrently.
- The operation of T1 and T2 are interleaved
- So, the schedule is called non-serial schedule.

Here, non-serial schedule is divided into many other hierarchy shown in fig above



⇒ • A schedule is called serializable whenever executing the transactions sequentially in order of some, could have left the database in the same state as the actual schedule. Serializability is the commonly accepted criterion for correctness.
• If the two transactions do not have operations on the same data item, so the schedule is serializable.

2012-Spring

6a) How does 2PL guarantee serialization? (2 Marks)

• In case of two phase locking the main idea is to set lock before the transaction start and before releasing lock it will take all the lock at the end of the transaction and it helps transactions to be carried out in serial form that is how it guarantee serializability. In 2PL technique it has two states or situation in which it perform locking mechanism, in order to perform concurrency control of the transactions and it will also keep the data histories,

i) Expanding phase :- In this phase only one process is adoptable means locking mechanism can be used but unlocking mechanism cannot be done if operation is in progress

ii) shrinking phase :- In this phase unlocking unlocking can be processed but locking cannot be processed by users when operation is under process or running.

6b) What is log-based recovery? how it is different from shadow paging

1815 46 25

1) • The log is a sequence of records. log of each transaction is maintained in some stable storage so that if any failure occurs, then it can be restored from there.

• If any operation is performed on the database, then it will be recorded in the log.

• But the process of storing the logs should be done before the actual transaction is applied in the database.

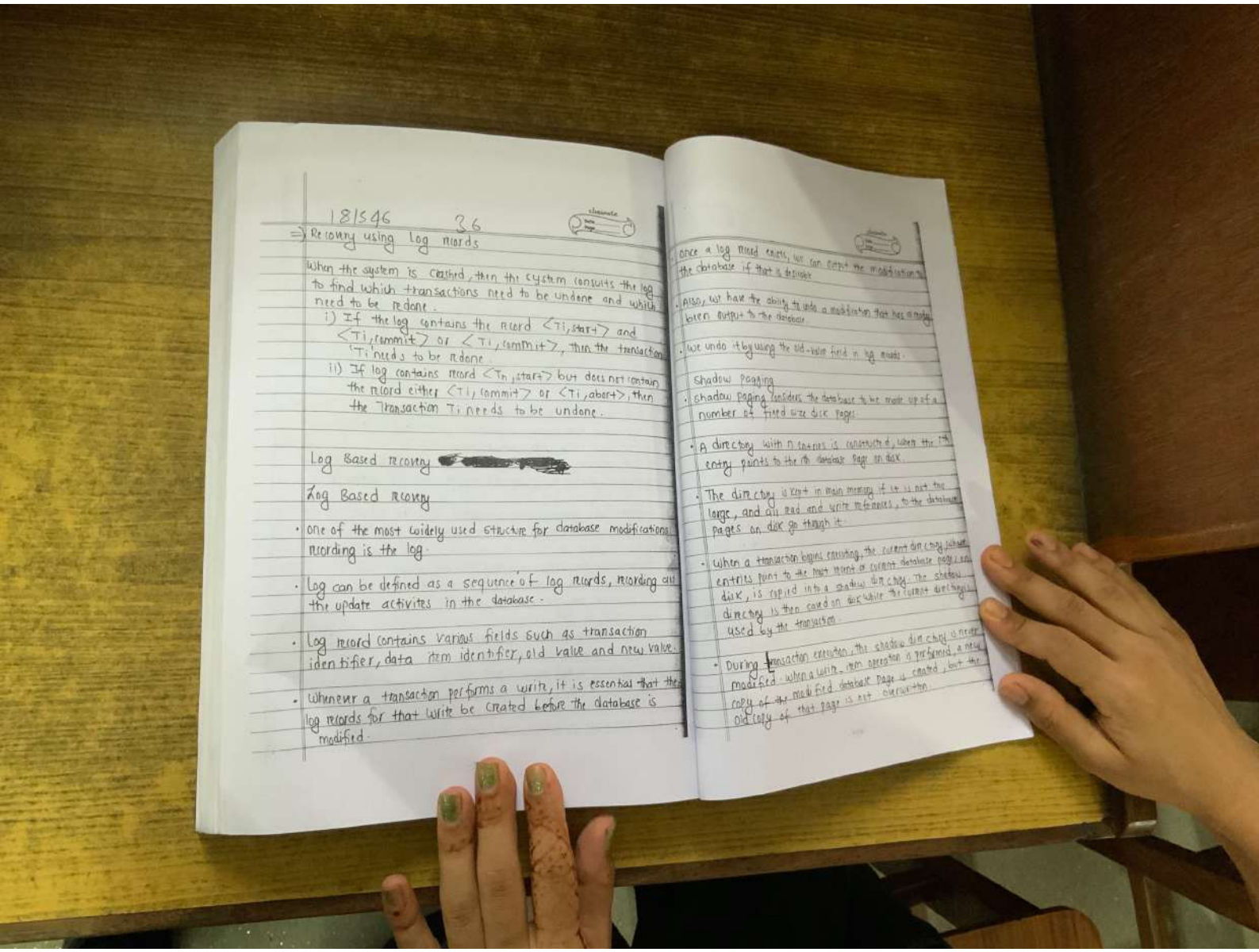
Let's assume, there is a transaction to modify the city of a student. The following logs are written for this transaction.

• When the transaction is initiated, then it writes 'start' log.
<Tn, start>

• When the transaction modifies the city from 'Noida' to 'Bangalore', then another log is written to the file.
<Tn, city, Noida, Bangalore>

• When the transaction is finished, then it writes another log to indicate the end of the transaction.
<Tn, commit>

There are two approaches to modify the database:
i) Deferred database modification ii) Immediate database modification



181546 36

=) Recovery using log records

When the system is crashed, then the system consults the log to find which transactions need to be undone and which need to be redone.

i) If the log contains the record <Ti, start> and <Ti, commit> or <Ti, commit>, then the transaction Ti needs to be redone.

ii) If log contains record <Tn, start> but does not contain the record either <Ti, commit> or <Ti, abort>, then the Transaction Ti needs to be undone.

Log Based Recovery [redacted]

Log Based Recovery

- one of the most widely used structure for database modifications recording is the log.
- Log can be defined as a sequence of log records, recording all the update activites in the database.
- Log record contains various fields such as transaction identifier, data item identifier, old value and new value.
- Whenever a transaction performs a write, it is essential that the log records for that write be created before the database is modified.

- Once a log record exists, we can output the modification to the database if that is desired.
- Also, we have the ability to undo a modification that has already been output to the database.
- We undo it by using the old-value field in log records.

Shadow Paging

- Shadow paging considers the database to be made up of a number of fixed size disk pages.
- A directory with n entries is constructed, where the ith entry points to the ith database page on disk.
- The directory is kept in main memory if it is not too large, and all read and write references to the database pages on disk go through it.
- When a transaction begins executing, the current directory whose entries point to the most recent or current database pages on disk, is copied into a shadow directory. The shadow directory is then saved on disk while the current directory is used by the transaction.
- During transaction execution, the shadow directory is never modified. When a write item operation is performed, a new copy of the modified database page is created, but the old copy of that page is not overwritten.

- Instead, the new page is written elsewhere, on some previously unused disk block. The current directory entry is modified to the new disk block, whereas the shadow directory is not modified and continues to point to the old unmodified disk block.

2013 Fall

5b) What is Stable storage? (2 Marks)

- A Stable storage is a storage in which information is never lost.
- Stable storage devices are theoretically impossible to obtain. But, we must use some technique to design a stable storage system in which the chances of data loss are extremely low.
- Stable storage is a classification of computer data storage technology that guarantees atomicity for any given write operation and allows software to be written that is robust against some hardware and power failures.
- During the recovery from a failure each of the physical blocks is examined. The data present in the stable storage is safe unless a failure destroys all the copies.
- The data that is present in the stable storage is guaranteed to be safe unless a failure destroys all the copies.

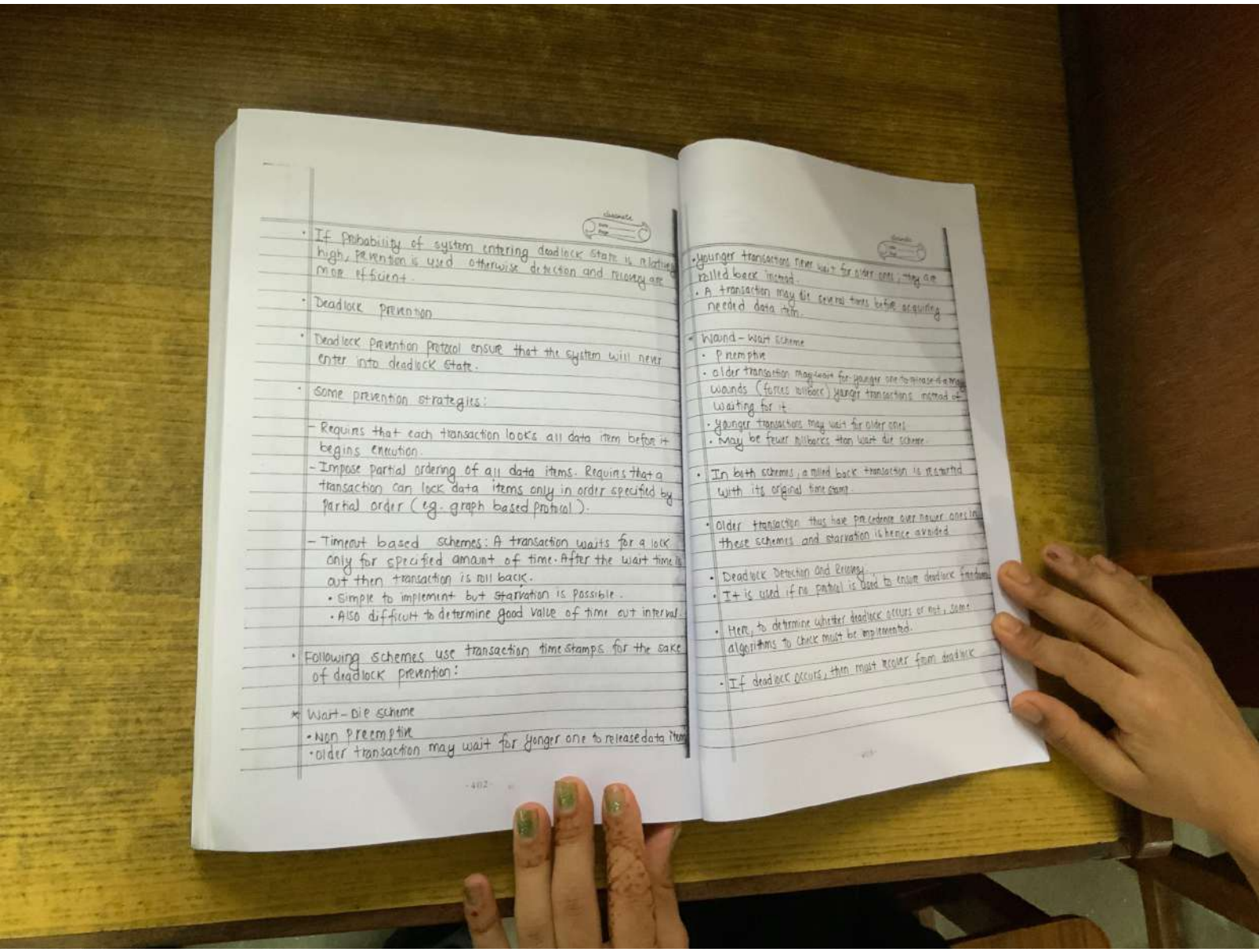
-400-

Describe the dead lock handling mechanism

- Dead lock Handling
- System is deadlocked if there is a set of transactions such that every transaction in the set is waiting for another transaction in the set.
- Let T1 : write(x) write(y)
  T2 write(y) write(x)
- Now, schedule with deadlock

T1	T2
Lock-X(x)	
write(x)	
	Lock-X(y)
	write(y)
	{wait for Lock-X on x}
	write(x)
{wait for Lock-X on y}	
write(y)	

- To deal with deadlock we can use 1. Deadlock prevention Protocol (which ensures that system will never enter a deadlock state) 2. Deadlock Detection and Recovery system once scheme (which try to recover system once it entered dead lock state)
- Both methods may result in transaction rollback



- If probability of system entering deadlock state is relatively high, prevention is used otherwise detection and recovery are more efficient.
- Deadlock prevention
- Deadlock prevention protocol ensure that the system will never enter into deadlock state.
- Some prevention strategies:
  - Requires that each transaction locks all data item before it begins execution.
  - Impose partial ordering of all data items. Requires that a transaction can lock data items only in order specified by partial order (eg. graph based protocol).
  - Timeout based schemes: A transaction waits for a lock only for specified amount of time. After the wait time is out then transaction is roll back.
    - Simple to implement but starvation is possible.
    - Also difficult to determine good value of time out interval.
- Following schemes use transaction timestamps for the sake of deadlock prevention:
- Wait-die scheme
  - Non preemptive
  - older transaction may wait for younger one to release data item

- younger transactions never wait for older ones; they are rolled back instead.
- A transaction may die several times before acquiring needed data item.
- Wound-wait scheme
  - Preemptive
  - older transaction may wait for younger one to release data item. Older transactions wounds (forces rollback) younger transactions instead of waiting for it.
  - younger transactions may wait for older ones
  - May be fewer rollbacks than wait die scheme.
- In both schemes, a rolled back transaction is restarted with its original time stamp.
- Older transaction thus have precedence over newer ones in these schemes and starvation is hence avoided.
- Deadlock Detection and Recovery.
- It is used if no protocol is used to ensure deadlock freedom.
- Here, to determine whether deadlock occurs or not, some algorithms to check must be implemented.
- If deadlock occurs, then must recover from deadlock.

Deadlock detection

- Deadlocks can be described as a wait for graph which consists of a pair $G = (V, E)$ where V is the set of vertices (transactions) and E is a set of edges; each edge is ordered pair $T_i \rightarrow T_j$
- If $T_i \rightarrow T_j$, then there is a directed edge from T_i to T_j. Here, T_i is waiting for T_j to release data item.
- When transaction T_i request a data item held by T_j then $T_i \rightarrow T_j$ is inserted in wait for graph. This edge is removed only when T_i no longer holding data item needed by T_j.
- The system is in deadlock state if and only if wait for graph has a cycle.
- The system invokes a deadlock detection algorithm periodically to look for a cycle.

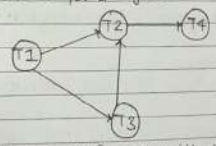


Fig:- wait for graph without cycle

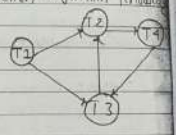


Fig:- wait for graph with cycle.

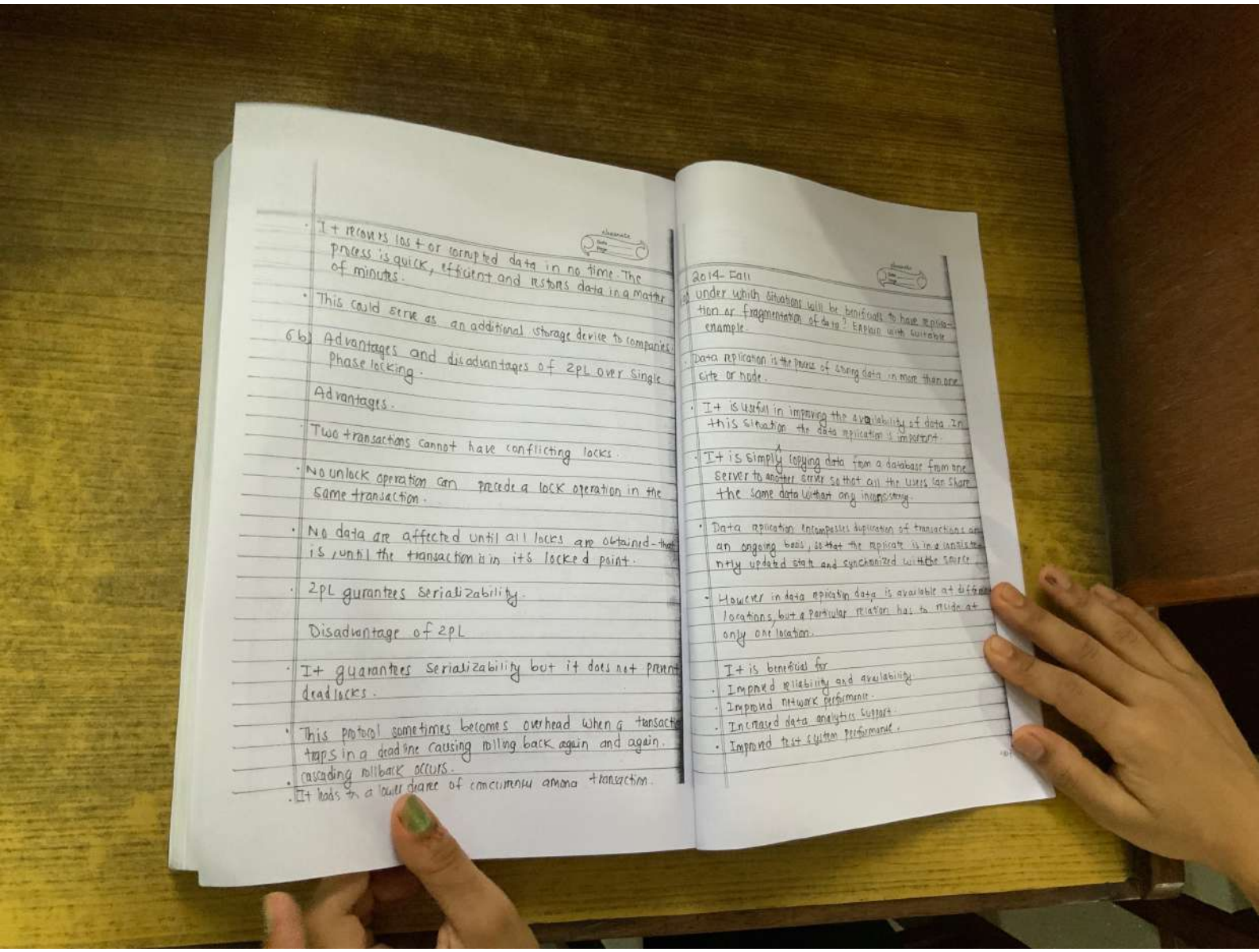
Deadlock Recovery

- When deadlock is detected:-
- some transactions will have to rollback to break deadlock. (Select a victim)
- Rollback - determine how far to rollback transaction.
- Total Rollback: abort transaction and then restart it.
- Partial Rollback: rollback transaction only as far as necessary to break the deadlock, and is more effective
- Starvation happens if same transactions is always chosen as victim.
- must ensure that transaction can be picked as a victim only a small no of times.

2013 - spring

5a) Why database recovery is needed? (3)

- Data recovery can retrieve or recover the user's personal photos, files etc which are precious and important to them.
- This provides ease off any information of file getting lost. This will provide companies being run by computers to securely keep confidential files until backed up. No issues of information lost in any cases after crashing or any virus or any event of device problems.
- It is important in sense that we are secured that even after crashing or any virus or malware attacks, our data could be still salvaged.



- It recovers lost or corrupted data in no time. The process is quick, efficient and restors data in a matter of minutes.
- This could serve as an additional storage device to companies.

6b) Advantages and disadvantages of 2PL over Single Phase locking.

Advantages.

- Two transactions cannot have conflicting locks.
- No unlock operation can precede a lock operation in the same transaction.
- No data are affected until all locks are obtained - that is, until the transaction is in its locked point.
- 2PL guarantees Serializability.

Disadvantage of 2PL

- It guarantees Serializability but it does not prevent deadlocks.
- This protocol sometimes becomes overhead when a transaction traps in a deadline causing rolling back again and again.
- cascading rollback occurs.
- It leads to a low degree of concurrency among transaction.

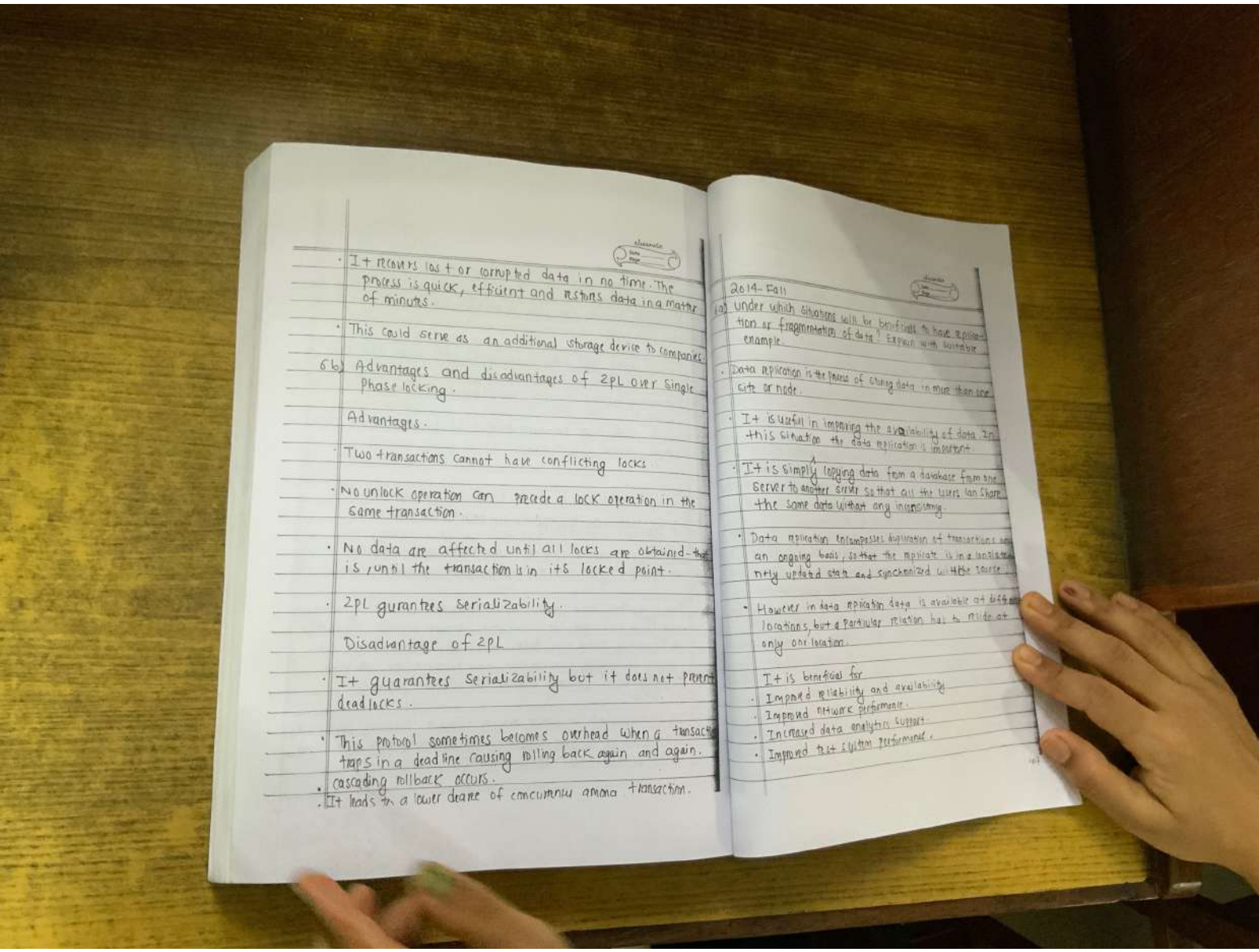
2014- Fall

6a) under which situations will be beneficials to have replication or fragmentation of data? Explain with suitable example.

- Data replication is the process of storing data in more than one site or node.
- It is useful in improving the availability of data. In this situation the data replication is important.
- It is simply copying data from a database from one server to another server so that all the users can share the same data without any inconsistency.
- Data replication encompasses duplication of transactions on an ongoing basis, so that the replicate is in a consistently updated state and synchronized with the source.
- However in data replication data is available at different locations, but a particular relation has to reside at only one location.

It is beneficial for

- Improved reliability and availability.
- Improved network performance.
- Increased data analytics support.
- Improved test system performance.



- It recovers lost or corrupted data in no time. The process is quick, efficient and restors data in a matter of minutes.
- This could serve as an additional storage device to companies.

6b) Advantages and disadvantages of 2PL over Single Phase locking.

Advantages.

- Two transactions cannot have conflicting locks.
- No unlock operation can precede a lock operation in the same transaction.
- No data are affected until all locks are obtained - that is, until the transaction is in its locked point.
- 2PL guarantees serializability.

Disadvantage of 2PL

- It guarantees serializability but it does not prevent deadlocks.
- This protocol sometimes becomes overhead when a transaction traps in a deadline causing rolling back again and again.
- cascading rollback occurs.
- It leads to a lower degree of concurrency among transaction.

2014- Fall

6a) Under which situations will be beneficial to have replication or fragmentation of data? Explain with suitable example.

- Data replication is the process of storing data in more than one site or node.
- It is useful in improving the availability of data. In this situation the data replication is important.
- It is simply copying data from a database from one server to another server so that all the users can share the same data without any inconsistency.
- Data replication encompasses duplication of transactions on an ongoing basis, so that the replicate is in a consistently updated state and synchronized with the source.
- However in data replication data is available at different locations, but a particular relation has to reside at only one location.

It is beneficial for

- Improved reliability and availability.
- Improved network performance.
- Increased data analysis support.
- Improved test system performance.

some commonly used replication technique are -
- snapshot replication.
- Near-real-time replication.
- pull replication.

eg:- data can be copied between two on-premises hosts, between hosts in different locations, to multiple storage devices on the same host, or to or from a cloud-based host.

⇒ advantages of data replication.

- To provide a consistent copy of data across all the database nodes.
- To increase the availability of data.
- The reliability of data is increased through data replication
- Data replication supports multiple users and gives high performance
- To remove any data redundancy, the databases are updated with out dated or incomplete data.
- To perform faster execution of queries.
- since replicas are created there are chances that the data is found itself where the transaction is executing which reduces the data movement.

⇒ Disadvantages of data Replication.

- More storage space is needed as storing the replicas of same data at different sites consumes more space.
- Data replication becomes expensive when the replicas at all different sites need to be updated.
- Maintaining data consistency at all different sites involves complex measures.

18/9/96

Deferred Database Modification

- The deferred database modification scheme records all modifications to the log, but defers all the writes to after partial commit.
- Assume that the transaction execute serially
- Transaction starts by writing <Ti, start> record to log
- A write(x) operations results in a log record <Ti, x, V> and old value is not needed in this scheme
- The write is not performed on x at this time, but is deferred.
- When Ti partially commits, <Ti, commit> is written to the log
- Finally, the log records are read and used to actually execute the previously deferred writes.
- During recovery after a crash, a transaction needs to be redone if and only if both <Ti, start> and <Ti, commit> are there in the log
- Redoing a transaction Ti (redo Ti) sets the value of all data items updated by the transaction to the new values.
- Crashes can occur while
  - The transaction is executing the original updates or
  - while recovery action is being taken

18154S 38

eg. Transaction T_0 and T_1 (T_0 executes before T_1) :-

T_0 : read(A)	T_1 = read(C);
A := A-50;	C := C-100;
write(A);	write(C);
read(B);	
B := B+50;	
write(B);	

Immediate database modification.

- The Immediate database Modification technique allows database modifications to be output to the database while the transaction is still in the active state.
- update log record must be written before database item is written.
  - we assume that the log record is output directly to stable storage.
  - can be extended to postpone log records output, so long as prior to execution of an output(B) operations for a data block B, all log records corresponding to items B must be flushed to stable storage.
- output of updated blocks can take place at any time before or after transaction commit.
- order in which blocks are output can be different from the order in which they are written.

18154S 39

eg:- Immediate DB modification Recovery

Below we show the log as it appears at three instants of time.

(a)	(b)	(c)
<T_0 start>	<T_0 start>	<T_0 start>
<T_0, A, 1000, 950>	<T_0, A, 1000, 950>	<T_0, A, 1000, 950>
<T_0, B, 2000, 2050>	<T_0, B, 2000, 2050>	<T_0, B, 2000, 2050>
	<T_0 commit>	<T_0 commit>
	<T_1 start>	<T_1 start>
	<T_1, C, 700, 600>	<T_1, C, 700, 600>
		<T_1 commit>

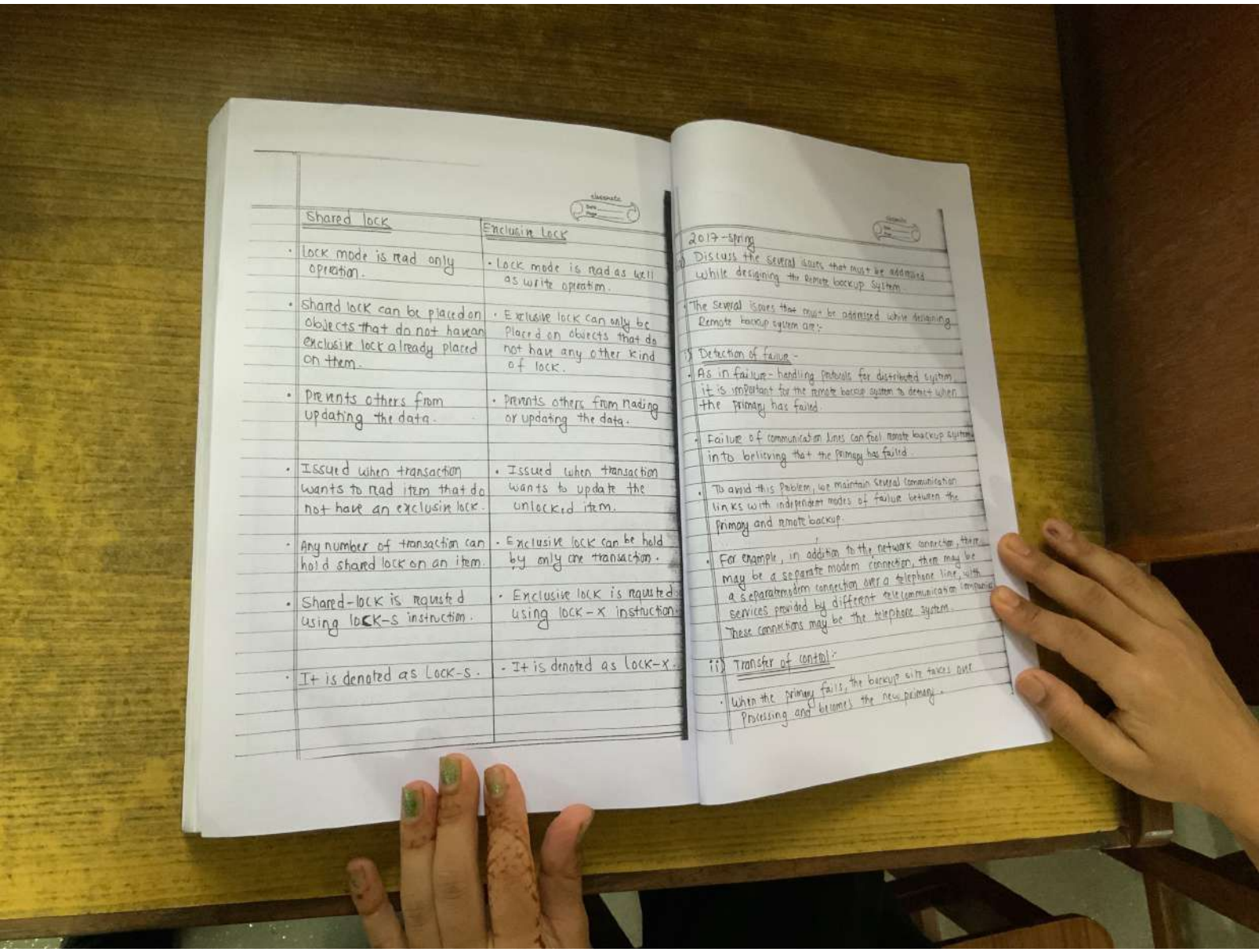
Recovery actions in each case above are:-

a) undo(T_0): B is restored to 2000 and A to 1000 and B are set to 950 and 2050 respectively.

b) undo(T_1) and redo(T_0): C is restored to 700, and then A and B are set to 950 and 2050 respectively.

c) redo(T_0) and redo(T_1): A and B are set to 950 and 2050 respectively. Then C is set to 600.

6b) Difference between Shared lock and Exclusive lock



Shared lock	Exclusive Lock
• Lock mode is read only operation.	• Lock mode is read as well as write operation.
• Shared lock can be placed on objects that do not have an exclusive lock already placed on them.	• Exclusive lock can only be placed on objects that do not have any other kind of lock.
• Prevents others from updating the data.	• Prevents others from reading or updating the data.
• Issued when transaction wants to read item that do not have an exclusive lock.	• Issued when transaction wants to update the unlocked item.
• Any number of transaction can hold shared lock on an item.	• Exclusive lock can be held by only one transaction.
• Shared-lock is requested using lock-s instruction.	• Exclusive lock is requested using lock-x instruction.
• It is denoted as Lock-s.	• It is denoted as Lock-x.

2017-spring

Q) Discuss the several issues that must be addressed while designing the remote backup system.

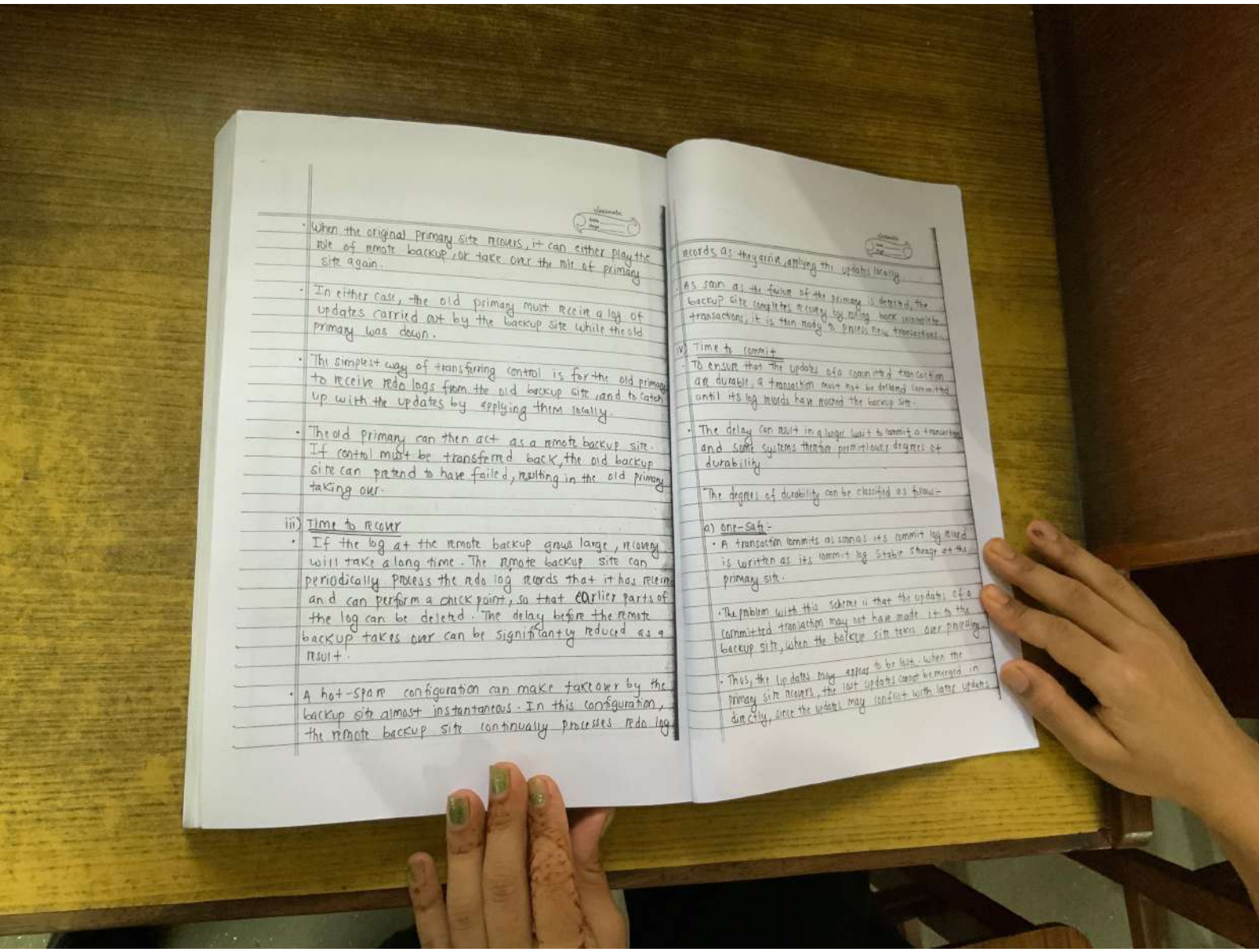
The several issues that must be addressed while designing Remote backup system are:-

i) Detection of failure -

- As in failure-handling protocols for distributed system, it is important for the remote backup system to detect when the primary has failed.
- Failure of communication lines can fool remote backup system into believing that the primary has failed.
- To avoid this problem, we maintain several communication links with independent modes of failure between the primary and remote backup.
- For example, in addition to the network connection, there may be a separate modem connection, there may be a separate modem connection over a telephone line, with services provided by different telecommunication companies. These connections may be the telephone system.

ii) Transfer of control:-

- When the primary fails, the backup site takes over Processing and becomes the new primary.



- When the original primary site recovers, it can either play the role of remote backup, or take over the role of primary site again.
- In either case, the old primary must receive a log of updates carried out by the backup site while the old primary was down.
- The simplest way of transferring control is for the old primary to receive redo logs from the old backup site, and to catch up with the updates by applying them locally.
- The old primary can then act as a remote backup site. If control must be transferred back, the old backup site can pretend to have failed, resulting in the old primary taking over.

iii) Time to recover

- If the log at the remote backup grows large, recovery will take a long time. The remote backup site can periodically process the redo log records that it has received and can perform a checkpoint, so that earlier parts of the log can be deleted. The delay before the remote backup takes over can be significantly reduced as a result.
- A hot-spare configuration can make takeover by the backup site almost instantaneous. In this configuration, the remote backup site continually processes redo log records as they arrive, applying the updates locally.
- As soon as the failure of the primary is detected, the backup site completes recovery by rolling back incomplete transactions; it is then ready to process new transactions.

iv) Time to commit

- To ensure that the updates of a committed transaction are durable, a transaction must not be declared committed until its log records have reached the backup site.
- The delay can result in a longer wait to commit a transaction, and some systems therefore permit lower degrees of durability.
- The degrees of durability can be classified as follow-

a) one-safe-

- A transaction commits as soon as its commit log record is written at its commit log to the stable storage at the primary site.
- The problem with this scheme is that the updates of a committed transaction may not have made it to the backup site, when the backup site takes over processing.
- Thus, the updates may appear to be lost. When the primary site recovers, the lost updates cannot be merged in directly, since the updates may conflict with later updates.

performed at the backup site. Thus, human intervention may be required to bring the database to a consistent state.

b) Two-very-safe :-

- A transaction commits as soon as its commit log record is written to stable storage at the primary and the backup site.
- The problem with this scheme is that transaction processing cannot proceed if either the primary or the backup site is down.
- Thus, availability is actually less than in the single-site case, although the probability of data loss is much less.

c) Two-safe :-

- This scheme is the same as two-very-safe if both primary and backup sites are active. If only the primary is active, the transaction is allowed to commit as soon as its commit log record is written to stable storage at the primary site.
- This scheme provides better availability than does two-very-safe, while avoiding the problem of lost transactions faced by the one-safe scheme.
- It results in a slower commit than the one-safe scheme, but the benefits generally outweigh the cost.

2018-Spring

(a) Explain the different states in a transaction (5 marks)

In a database, the transaction can be in one of the following states.

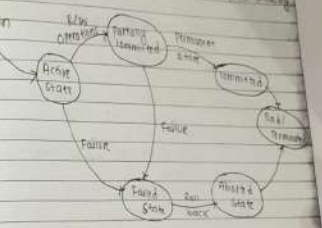
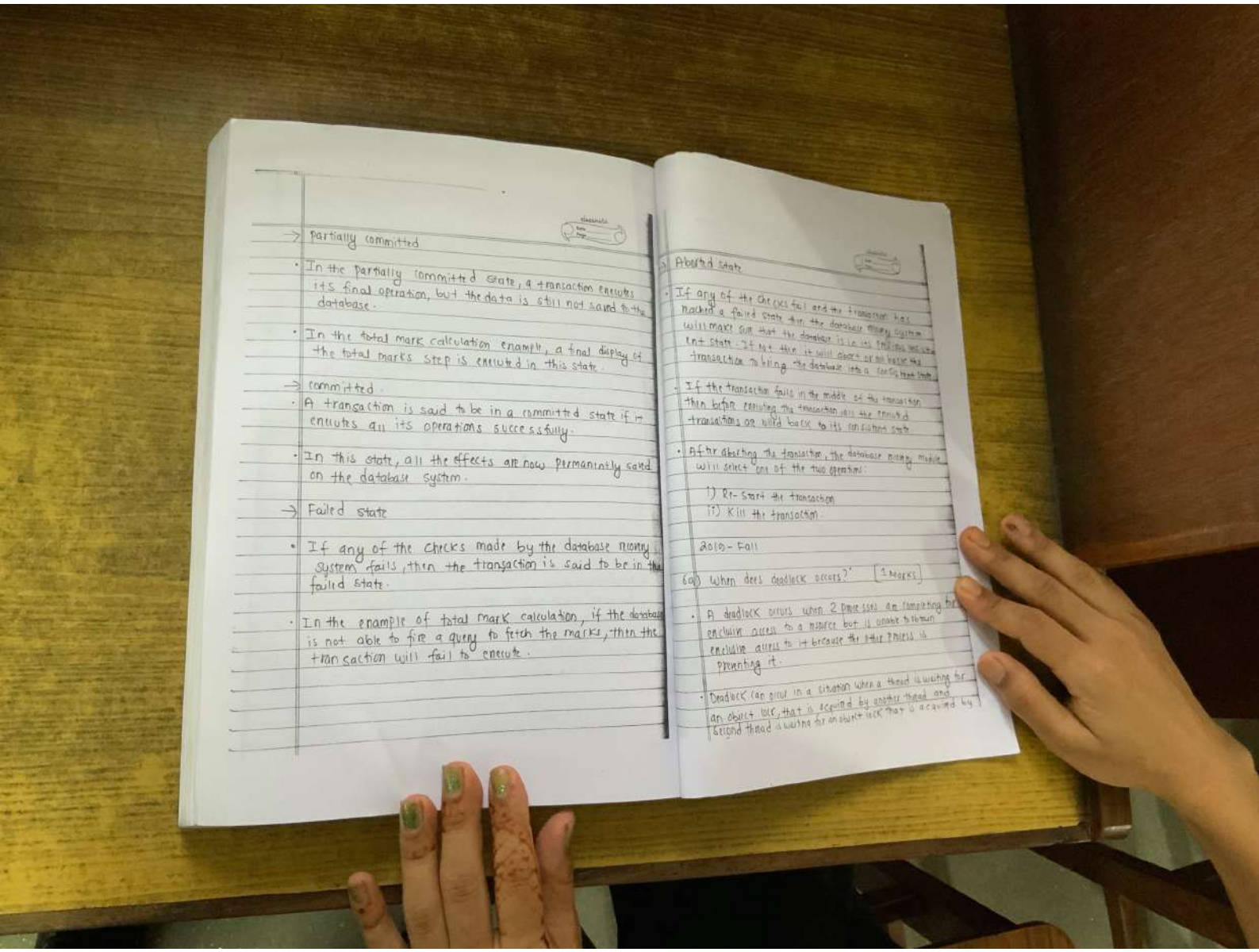


Fig:- Transaction states in DBMS

→ Active state

- The active state is the first state of every transaction. In this state, the transaction is being executed.
- Foreg:- Insertion or deletion or updating a record is done here. But all the records are still not saved to the database.



→ Partially committed

- In the partially committed state, a transaction executes its final operation, but the data is still not saved to the database.
- In the total mark calculation example, a final display of the total marks step is executed in this state.

→ committed

- A transaction is said to be in a committed state if it executes all its operations successfully.
- In this state, all the effects are now permanently saved on the database system.

→ Failed state

- If any of the checks made by the database recovery system fails, then the transaction is said to be in the failed state.
- In the example of total mark calculation, if the database is not able to fire a query to fetch the marks, then the transaction will fail to execute.

→ Aborted state

- If any of the checks fail and the transaction has reached a failed state then the database recovery system will make sure that the database is in its previous consistent state. If not then it will abort or roll back the transaction to bring the database into a consistent state.
- If the transaction fails in the middle of the transaction then before executing the transaction all the executed transactions are rolled back to its consistent state.
- After aborting the transaction, the database recovery module will select one of the two operations:
  i) Re-start the transaction
  ii) Kill the transaction.

2019 - Fall

6a) When does deadlock occurs? [1 Marks]

- A deadlock occurs when 2 processes are competing for exclusive access to a resource but is unable to obtain exclusive access to it because the other process is preventing it.
- Deadlock can occur in a situation when a thread is waiting for an object lock, that is acquired by another thread and second thread is waiting for an object lock that is acquired by

6 b) What are data fragmentations? State the various fragmentation with examples.

Data fragmentations

- Fragmentation is the task of dividing a table into a set of smaller tables.
- The subsets of the tables are called fragments.
- Fragmentation should be done in a way so that the original table can be reconstructed from the fragments.
- This is needed so that the original table can be reconstructed from the fragments whenever required. The requirement is called "reconstructiveness."

The types of data fragmentations are:-

i) Vertical Fragmentation -

- In vertical fragmentation, the fields or columns of a table are grouped into fragments.
- In order to maintain reconstructiveness, each fragment should contain the primary key field(s) of the table.
- Vertical fragmentation can be used to enforce privacy of data.

- For eg -

Student (Regd_No, Name, course, address, semester, Fees, marks)

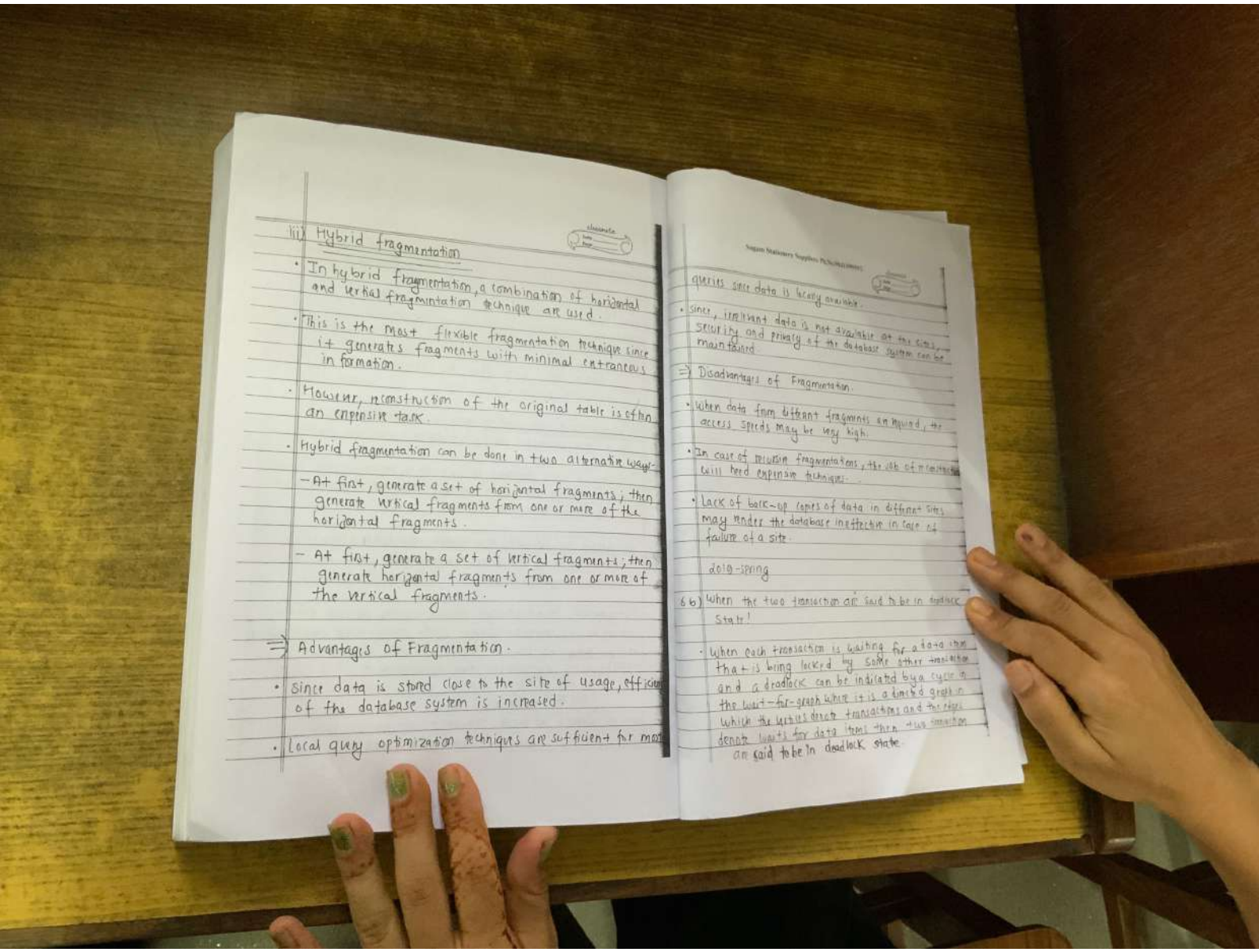
Now, fees are maintained in the accounts section. In this case, the designer will fragment the database as follows -

```
Create table std_Fees As select Regd_No, Fees
From Student;
```

ii) Horizontal Fragmentation groups the types of a table in accordance to values of one or more fields

- It should also conform to the rule of reconstructiveness. Each horizontal fragment must have all columns of the original base table.
- Horizontal fragmentation can further be divided into two techniques: primary Horizontal fragmentation and derived horizontal fragmentation.
- For example, the student Scheme, if the details of all students of computer Science course needs to be maintained at the school of computer science, then the designer will horizontally fragment the database as:

```
Create comp_std as
Select * From student
Where Course = "Computer Science";
```



iii) Hybrid fragmentation

- In hybrid fragmentation, a combination of horizontal and vertical fragmentation technique are used.
- This is the most flexible fragmentation technique since it generates fragments with minimal extraneous information.
- However, reconstruction of the original table is often an expensive task.
- Hybrid fragmentation can be done in two alternative ways-
  - At first, generate a set of horizontal fragments; then generate vertical fragments from one or more of the horizontal fragments.
  - At first, generate a set of vertical fragments; then generate horizontal fragments from one or more of the vertical fragments.

⇒ Advantages of Fragmentation.

- Since data is stored close to the site of usage, efficiency of the database system is increased.
- Local query optimization techniques are sufficient for most queries since data is locally available.
- Since, irrelevant data is not available at the sites, security and privacy of the database system can be maintained.

⇒ Disadvantages of Fragmentation.

- When data from different fragments are required, the access speeds may be very high.
- In case of recursive fragmentations, the job of reconstruction will need expensive techniques.
- Lack of back-up copies of data in different sites may render the database ineffective in case of failure of a site.

data-spring

6 b) When the two transaction are said to be in deadlock state?

- When each transaction is waiting for a data item that is being locked by some other transaction and a deadlock can be indicated by a cycle in the wait-for-graph where it is a directed graph in which the vertices denote transactions and the edges denote waits for data items then the two transaction are said to be in deadlock state.

A deadlock occurs when the first process locks the first resource at the same time as the second process locks the second resource. The deadlock can be resolved by cancelling and restarting the first process.

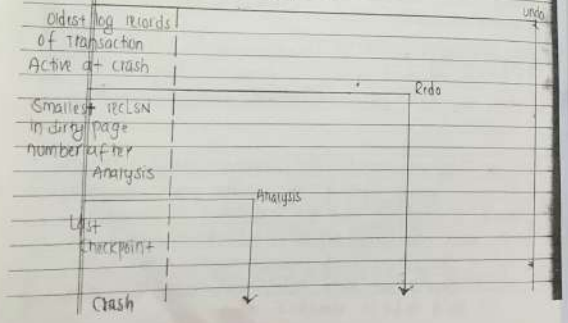
2020 Fall

6(a) What are the various crash recovery algorithm?

- Crash recovery includes three phases.

- Analysis :- Determines which transactions committed since checkpoint and which ones failed.
- REDO all actions.
  - (repeat history)
- UNDO effects of uncommitted transactions (the active transactions at the time of the crash.

Crash recovery phases.



Classification of mass movement.

Types :

Landslide	Debris flow	Slope failure
Movement of large sediment block, which has clear slide surface, large dimension. slow of contenuous movement mainly affected by underground.	Movement of deposited or eroded sediment along the stream. Rapid movement including large volume of water through the stream.	Movement of weathered surface soil layer/rock of steep slope small dimension at rapid movement.

Landslide	Slope failure
Geology:- very often occurs in places with specific geology	Not much related to the geology
Characteristic of soil:- Take place with a sliding surface mainly in clayey soil	Take place in weathered surface of soil/rock in sandy soil.
Condition of Movement :- Usually the velocity is low continuous, recurrent.	Velocity is extremely high, rapid movement occurs suddenly.
Condition of soil mass :- Disturbance to solid mass is small	Soil mass is disturbed.
Cause of movement :- Influence by ground water.	Influence by surface & sub- surface water (rain fall intensity)
Scale of moving mass :- large	Small
Gradient of slope :- Gentle	Step

Difference bet" landslide of slope failure.