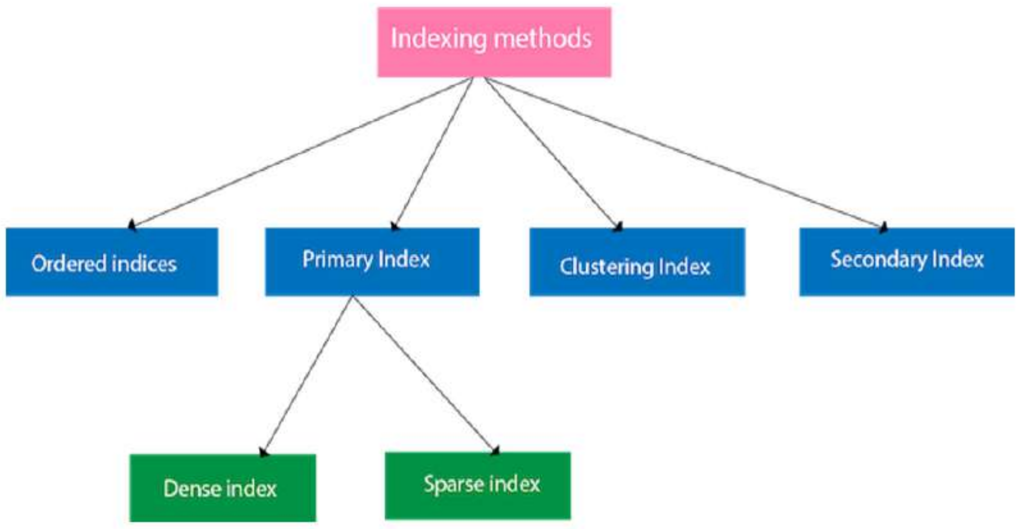
DBMS Mid-2 Important Questions & Answers (Part-II)

1) What are the different types of indexes? Explain.

Ans:-

Indexing Methods



1.Ordered indices

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

- **Example**: 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.

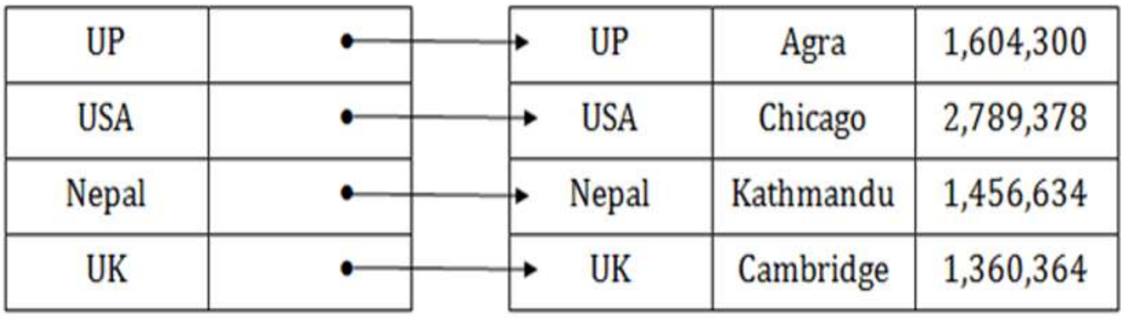
2.Primary Index

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

- The primary index can be classified into two types:
  a. **Dense index**
  b. **Sparse index**

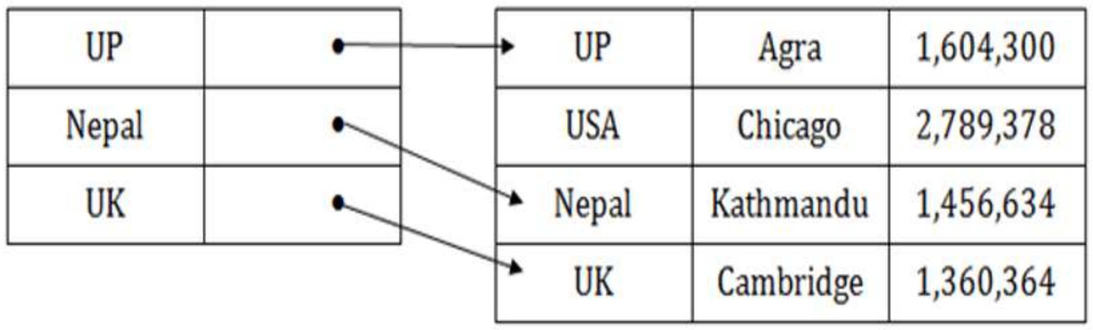
a. Dense index

- 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
USA
Nepal
UK
UP | Agra | 1,604,300
USA | Chicago | 2,789,378
Nepal | Kathmandu | 1,456,634
UK | Cambridge | 1,360,364

b.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.



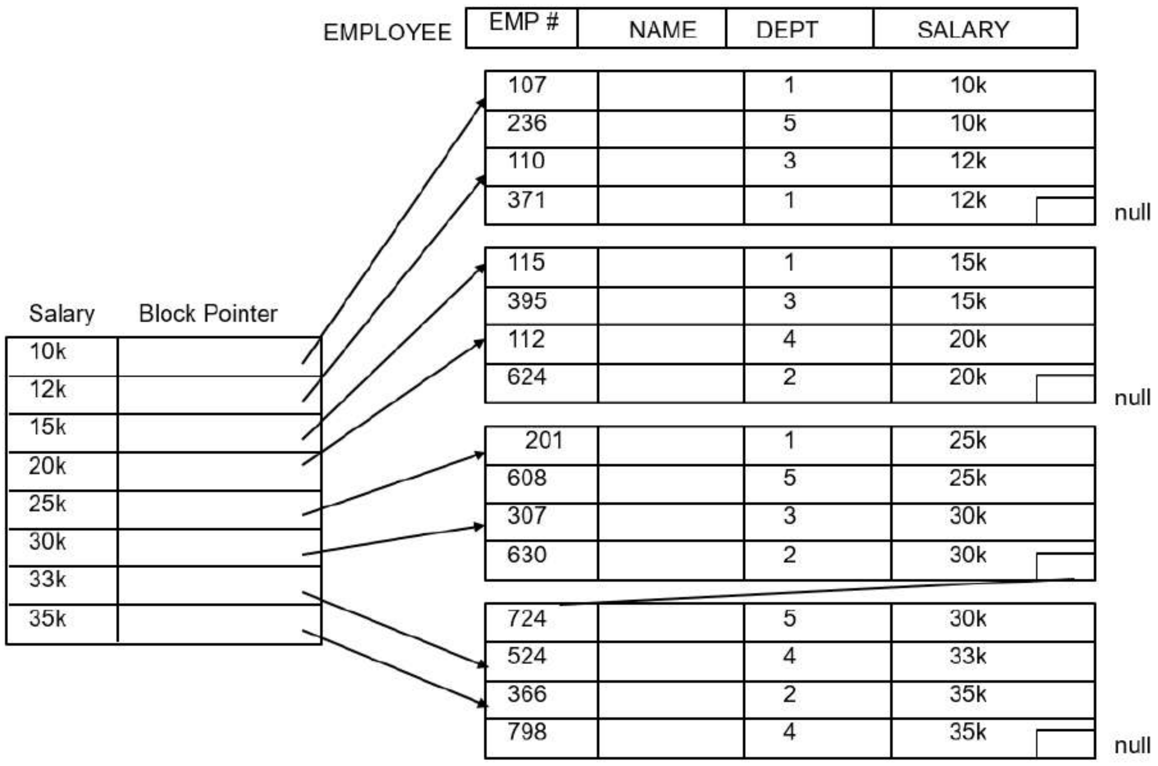
3.Clustering Index

- A clustered index can be defined as an ordered data file. Sometimes the index is created on non-primary key columns which may not be unique for each record.

- 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.

- The records which have similar characteristics are grouped, and indexes are created for these group.

Clustering Index Example....



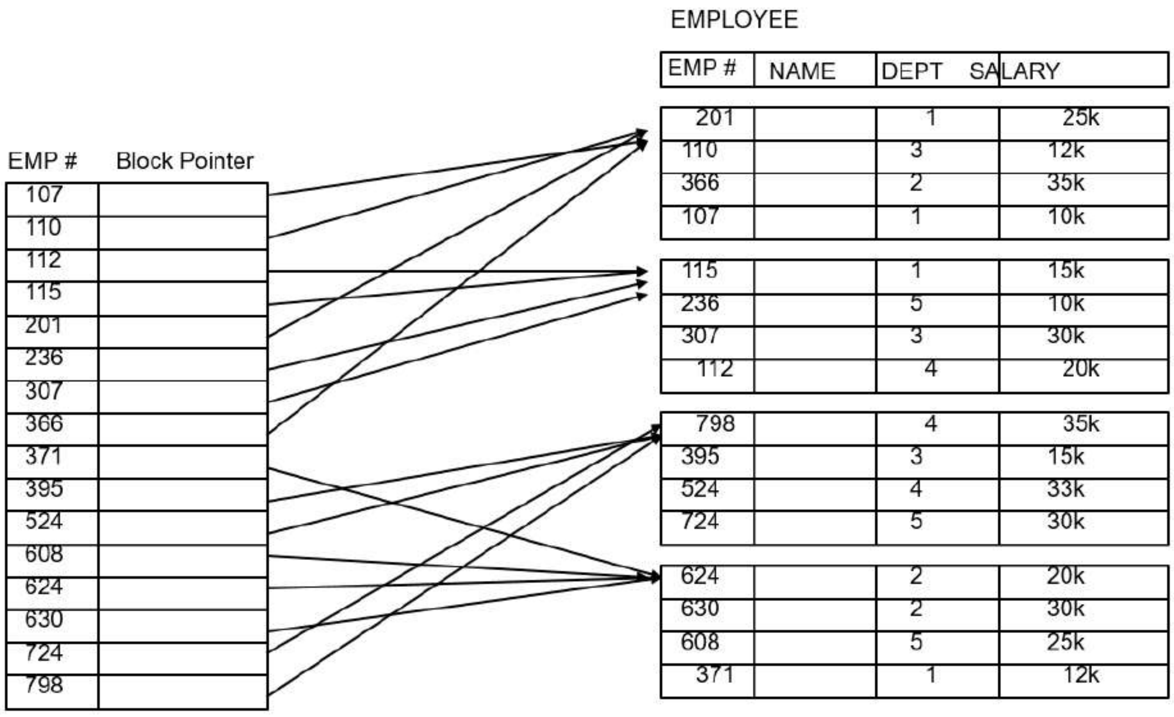
Salary	Block Pointer
10k	
12k	
15k	
20k	
25k	
30k	
33k	
35k	

EMP #	NAME	DEPT	SALARY
107		1	10k
236		5	10k
110		3	12k
371		1	12k
115		1	15k
395		3	15k
112		4	20k
624		2	20k
201		1	25k
608		5	25k
307		3	30k
630		2	30k
724		5	30k
524		4	33k
366		2	35k
798		4	35k

4.Secondary indexing

- 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.
- Then each range is further divided into smaller ranges. The mapping of the first level is stored in the primary memory, so that address fetch is faster.
- The mapping of the second level and actual data are stored in the secondary memory (hard disk).

Secondary indexing Example



EMP #	Block Pointer
107	
110	
112	
115	
201	
236	
307	
366	
371	
395	
524	
608	
624	
630	
724	
798	

EMPLOYEE

EMP #	NAME	DEPT	SALARY
201		1	25k
110		3	12k
366		2	35k
107		1	10k
115		1	15k
236		5	10k
307		3	30k
112		4	20k
798		4	35k
395		3	15k
524		4	33k
724		5	30k
624		2	20k
630		2	30k
608		5	25k
371		1	12k

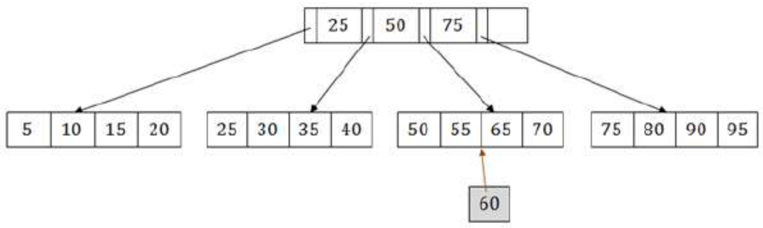
2) Explain the insertions and deletions in B+ Trees.

Ans:-

B+ Tree Insertion

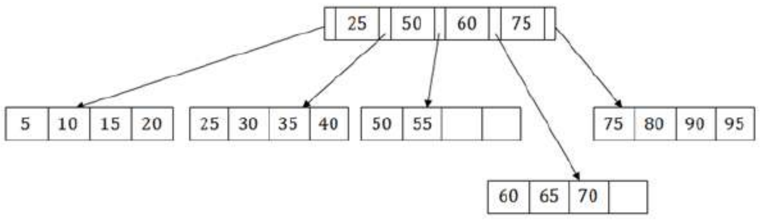
Suppose we want to insert a record 60 in the below structure. It will go to the 3rd leaf node after 55. It is a balanced tree, and a leaf node of this tree is already full, so we cannot insert 60 there.

In this case, we have to split the leaf node, so that it can be inserted into tree without affecting the fill factor, balance and order.



The 3rd leaf node has the values (50, 55, 60, 65, 70) and its current root node is 50. We will split the leaf node of the tree in the middle so that its balance is not altered. So we can group (50, 55) and (60, 65, 70) into 2 leaf nodes.

If these two has to be leaf nodes, the intermediate node cannot branch from 50. It should have 60 added to it, and then we can have pointers to a new leaf node.

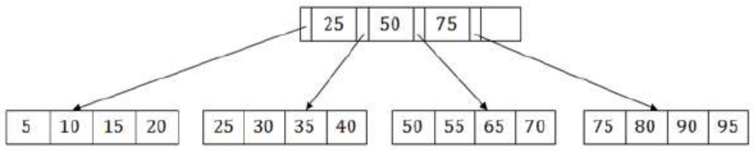


This is how we can insert an entry when there is overflow. In a normal scenario, it is very easy to find the node where it fits and then place it in that leaf node.

B+ Tree Deletion

Suppose we want to delete 60 from the above example. In this case, we have to remove 60 from the intermediate node as well as from the 4th leaf node too. If we remove it from the intermediate node, then the tree will not satisfy the rule of the B+ tree. So we need to modify it to have a balanced tree.

After deleting node 60 from above B+ tree and re-arranging the nodes, it will show as follows:



3) Mention the properties of decomposition. Explain each.

Ans:-

Following are the properties of Decomposition,

1. Lossless Decomposition
2. Dependency Preservation

1. Lossless Decomposition

- Decomposition must be lossless. It means that the information should not get lost from the relation that is decomposed.
- It gives a guarantee that the join will result in the same relation as it was decomposed.

2. Dependency Preservation

- Dependency is an important constraint on the database.
- Every dependency must be satisfied by at least one decomposed table.
- If {A → B} holds, then two sets are functional dependent. And, it becomes more useful for checking the dependency easily if both sets in a same relation.
- This decomposition property can only be done by maintaining the functional dependency.
- In this property, it allows to check the updates without computing the natural join of the database structure.

4) What is Multi valued Dependency? How can it be removed? Give an example.

Ans:-

Explanation – The multivalued dependencies is that, if there is a dependency or relation in a table, then one value has multiple dependencies occur.

Let us consider an example as given below. Consider the following table –

id	department	shift
1	coding	day
2	Hr	day
3	Network	night

In the above table, id 2 has two departments Hr and Network. And shift timing day and night.

When we select the details with the id 2, then it will result the table as follows –

id	department	shift
2	Hr	day
2	Network	night
2	Hr	night
2	Network	day

This means there exist multivalued dependencies. In this, the relation between department and shift is nothing.

This can be rectified by removing the multivalued dependency as, making this data in to two tables as below –

Table 1

id	department
1	coding
2	Hr
2	network

Table 2

id	shift
1	day
2	day
2	night

The 4th normal form is applied to remove the multivalued dependencies in the data table.

5) Discuss about two phase locking protocol and its variants.

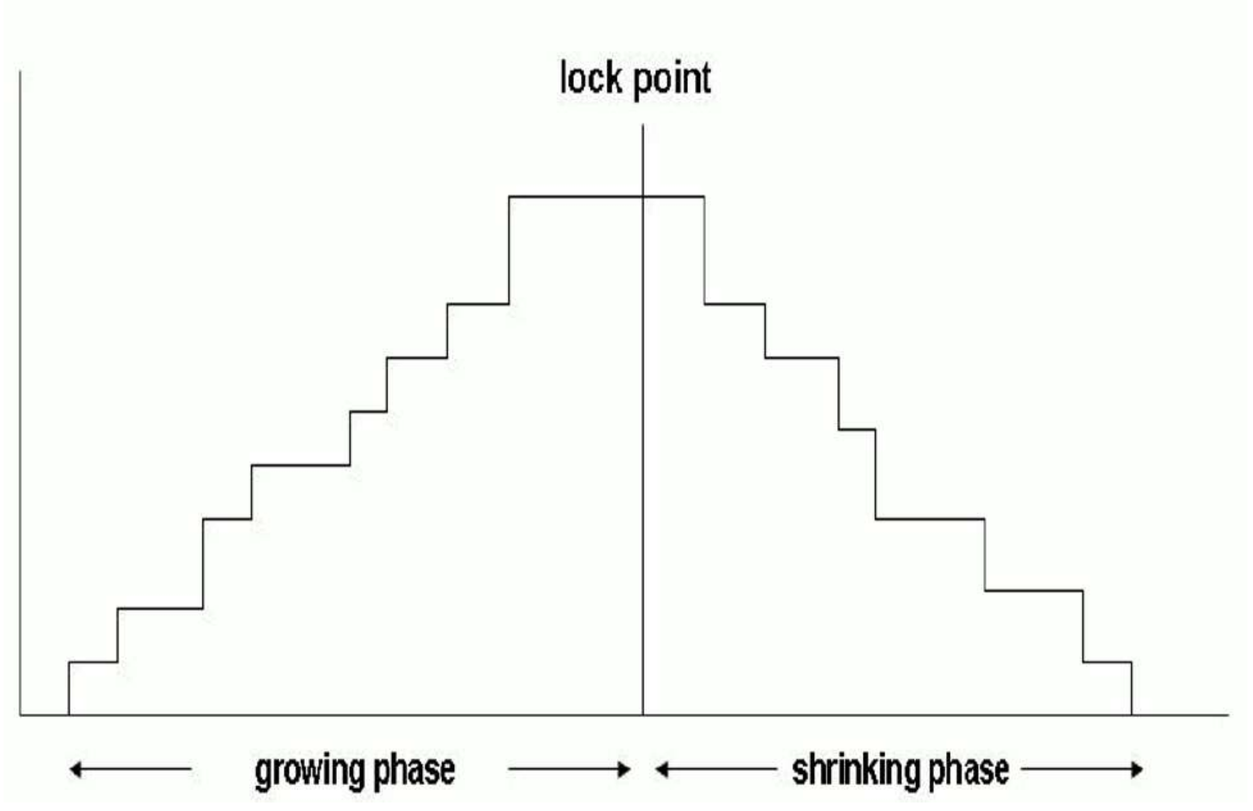
Ans:-

Two Phase Locking

- The Two Phase Locking Protocol defines the rules of how to acquire the locks on a data item and how to release the locks.
- The Two Phase Locking Protocol assumes that a transaction can only be in one of two phases.

Growing Phase:

- In this phase the transaction can only acquire locks, but cannot release any lock.
- The transaction enters the growing phase as soon as it acquires the first lock it wants.
- It cannot release any lock at this phase even if it has finished working with a locked data item.
- Ultimately the transaction reaches a point. This point is called **Lock Point**.



Shrinking Phase:

- After Lock Point has been reached, the transaction enters the shrinking phase.
- In this phase the transaction can only release locks, but cannot acquire any new lock.
- The transaction enters the shrinking phase as soon as it releases the first lock after crossing the Lock Point.
- Here it keeps releasing all the acquired locks.
- There are two different versions of the Two Phase Locking Protocol.

- **Strict Two Phase Locking Protocol**
- **Rigorous Two Phase Locking Protocol**

Strict Two Phase Locking Protocol

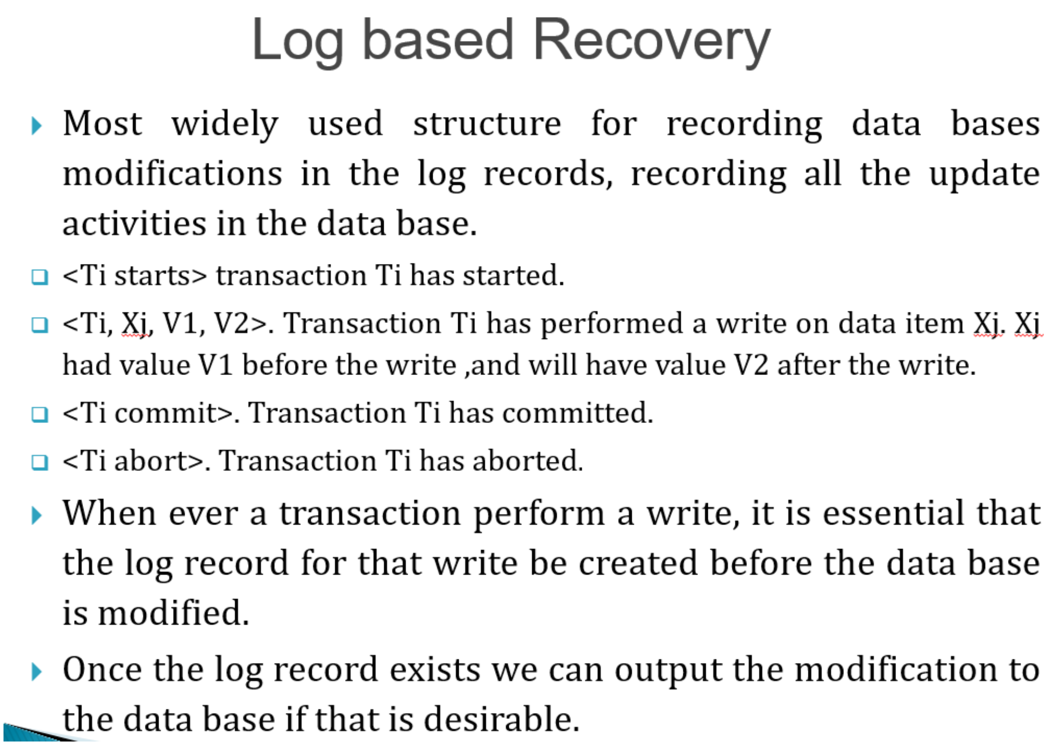
- In this protocol, a transaction may release all the shared locks after the Lock Point has been reached.
- But it cannot release any of the exclusive locks until the transaction commits.
- This protocol helps in creating cascade less schedule.

Rigorous Two Phase Locking Protocol

- A transaction is not allowed to release any lock (either shared or exclusive) until it commits.
- This means that until the transaction commits, other transaction might acquire a shared lock on a data item on which the uncommitted transaction has a shared lock.
- But cannot acquire any lock on a data item on which the uncommitted transaction has an exclusive lock

6) What is Log Based Recovery Technique.

Ans:-



Log based Recovery

- Most widely used structure for recording data bases modifications in the log records, recording all the update activities in the data base.
  - <Ti starts> transaction Ti has started.
  - <Ti, Xj, V1, V2>. Transaction Ti has performed a write on data item Xj. Xj had value V1 before the write ,and will have value V2 after the write.
  - <Ti commit>. Transaction Ti has committed.
  - <Ti abort>. Transaction Ti has aborted.
- When ever a transaction perform a write, it is essential that the log record for that write be created before the data base is modified.
- Once the log record exists we can output the modification to the data base if that is desirable.

- 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 (redoTi) sets the value of all data items updated by the transaction to the new values.

Recovery procedure has two operations.

- **undo(Ti)** restores the value of all data items updated by Ti to their old values, going backwards from the last log record for Ti
- **redo(Ti)** sets the value of all data items updated by Ti to the new values, going forward from the first log record for Ti

When recovering after failure:

- Transaction Ti needs to be undone if the log contains the record <Ti start>, but does not contain the record <Ti commit>.
- Transaction Ti needs to be redone if the log contains both the record <Ti start> and the record <Ti commit>.

<u>**All the best**</u>

<u>**Now do the**</u>

<u>**Rest…**</u>