TRANSACTION RECOVERY

Recovery algorithms are techniques to ensure database consistency and transaction atomicity and durability despite failures

Recovery using Log records (Log-Based Recovery)

- 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 recovered 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.

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.

1. If the log contains the record <Ti, Start> and <Ti, Commit> or <Ti, Commit>, then the Transaction Ti needs to be redone.
2. If log contains record<T_n, Start> but does not contain the record either <Ti, commit> or <Ti, abort>, then the Transaction Ti needs to be undone.

Recovery algorithms have two parts

1. Actions taken during normal transaction processing to ensure enough information exists to recover from failures
2. Actions taken after a failure to recover the database contents to a state that ensures atomicity, consistency and durability

Example

Begin transaction

Update Acc 1001{balance:=Balance-100};

If any error occurred then Goto **Undo**;

End if;

Update Acc 1002{balance:=balance+100};

If any error occurred then Goto **undo**;

End if;

Commit;

Goto finish;

Undo: rollback;

Finish: return;

Requirement for recovery

- Implicit rollback
- Message handling
- Recovery log
- Statement atomicity
- No nested transaction

Transaction recovery

Database updates are kept in buffer in main memory and not physically written to disk until commit.

System recovery

- **Local failures** –affect only the transaction which the failure has actually occurred.
- **Global failures**- affect all the transaction in progress at the time of failure.
- **System failure** – do not physically damage the DB Eg: power shut down Media failure- cause damage to the DB. Eg: head crash ARIES

Recovery using Checkpoints:
Checkpoints:

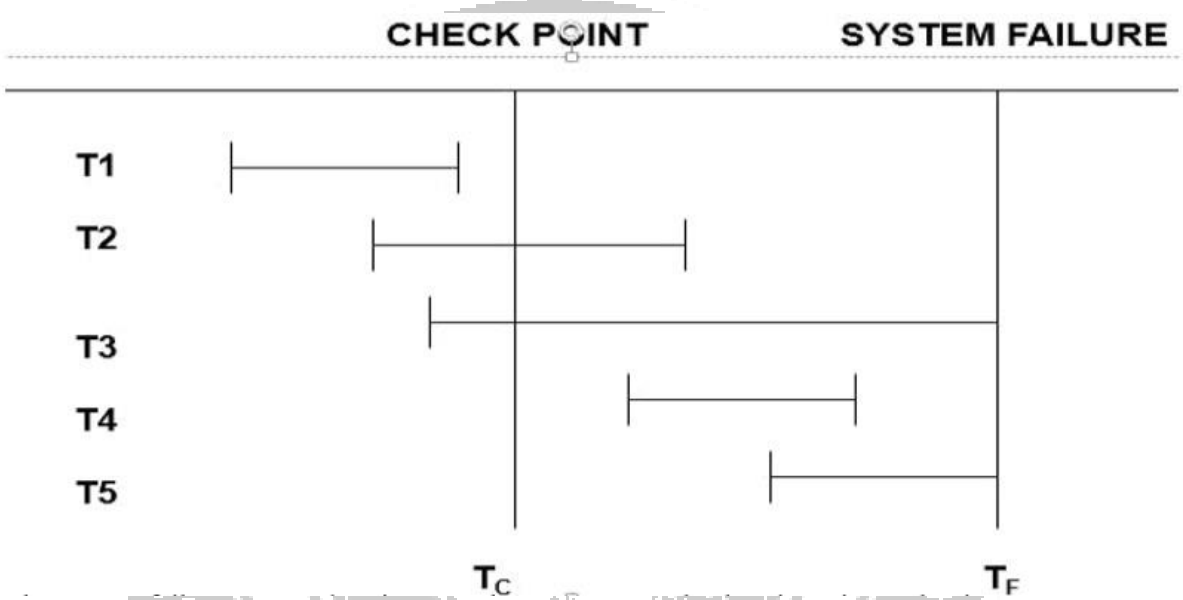
- Keeping and maintaining logs in real time and in real environment may fill out all the memory space available in the system. As time passes, the log file may grow too big to be handled at all. The checkpoint is a type of mechanism where all the previous logs are removed from the system and permanently stored in the storage disk.
- The checkpoint is like a bookmark. While the execution of the transaction, such checkpoints are marked, and the transaction is executed then using the steps of the transaction, the log files will be created.
- When it reaches to the checkpoint, then the transaction will be updated into the database, and till that point, the entire log file will be removed from the file. Then the log file is updated with the new step of transaction till next checkpoint and so on.
- The checkpoint is used to declare a point before which the DBMS was in the consistent state, and all transactions were committed.

Recovery Algorithm

ARIES-Algorithm for Recovery and Isolation Exploiting Semantics

ARIES recovery involves three passes

- Analysis pass: Determines the REDO and UNDO lists.
- Redo pass: Repeats history, redoing all actions from REDO List
- Undo pass: Rolls back all incomplete transactions



The system failure occurred at time Tf , the most recent check point prior to the time Tf was taken at a time Tc

- Start with two list of transaction the UNDO and REDO list
- Search forward through the log starting from check point.
- If begin transaction log record is found for transaction(T) add T to UNDO list.
- If commit log record is found for transaction(T),add T to REDO list
- When the end of log record is reached the UNDO and REDO list is identified

UNDO : T3 T2

REDO : T5 T4