12- Transaction - Recovery

School of Computer Science
University of Windsor

Dr Shafaq Khan

Agenda

➢ **Lecture**

- The Need for Recovery
- Transactions and Recovery
- Recovery Facilities
- Recovery Techniques

Introductory Questions

What are the causes of database failure?

What is the purpose of the transaction log file?

What is the purpose of checkpoints during transaction logging?

How to recover from database failure?

Database Recovery

Database Recovery: The process of restoring the database to a correct state in the event of a failure.

To ensure that the database is reliable and remains in a consistent state in the presence of failures.

The Need for Recovery

- ✓**System crashes** due to hardware or software errors, resulting in loss of main memory;
- ✓**Media failures**, such as head crashes or unreadable media, resulting in the loss of parts of secondary storage;
- ✓**Application software errors**, such as logical errors in the program that is accessing the database, that cause one or more transactions to fail;
- ✓**Natural physical disasters**, such as fires, floods, earthquakes, or power failures;
- ✓**Carelessness** or unintentional destruction of data or facilities by operators or users;
- ✓**Sabotage**, or intentional corruption or destruction of data, hardware, or software facilities.

Transactions and Recovery

Transactions is the basic unit of recovery in a database system.

It is the role of the recovery manager to guarantee two of the four ACID properties of transactions, namely **atomicity** and **durability**, in the presence of failures.

The recovery manager has to ensure that, on recovery from failure

all the effects of a given transaction are permanently recorded in the database OR none of them are recorded in the database

```
read(staffNo = x, salary)
salary = salary * 1.1
write(staffNo = x, salary)
```

For the Read operation, the DBMS carries out the following steps:

- find the address of the disk block that contains the record with primary key value x;
- transfer the disk block into a database buffer in main memory
- copy the salary data from the database buffer into the variable salary

For the write operation, the DBMS carries out the following steps:

- find the address of the disk block that contains the record with primary key value x;
- transfer the disk block into a database buffer in main memory
- copy the salary data from the variable salary into the database buffer;
- write the database buffer back to disk

Transactions and Recovery

What is buffer?

- The database buffers occupy an area in main memory from which data is transferred to and from secondary storage.
- Only once the buffers have been flushed to secondary storage can any update operations be regarded as permanent.

Force-writing: The explicit writing of the buffers to secondary storage.

Rollforward:

- If a failure occurs between writing to the buffers and flushing the buffers to secondary storage, the recovery manager must determine the status of the transaction that performed the write at the time of failure.
- If the transaction had issued its **commit**, then to ensure **durability** the recovery manager would have to **redo** that transaction's updates to the database

Rollback:

- If the transaction had **not committed** at the time of failure, then the recovery manager would have to **undo** any effects of that transaction on the database to guarantee transaction **atomicity**.

Recovery Facilities

A DBMS should provide the following facilities to assist with recovery:

1. A backup mechanism:

which makes **periodic backup copies** of the database and the log file to be made at regular intervals.

The DBMS should provide a mechanism to allow backup copies of the database and the log file to be made at regular intervals without necessarily having to stop the system first.

Typically, the backup is stored on offline storage, such as an optical disk.

2. Logging facilities:

which keep track of the current state of transactions and database changes. It contains information about all updates to the database.

A segment of a log file.

Tid	Time	Operation	Object	Before image	After image	pPtr	nPtr
T1	10:12	START				0	2
T1	10:13	UPDATE	STAFF SL21	(old value)	(new value)	I	8
T2	10:14	START				0	4
T2	10:16	INSERT	STAFF SG37		(new value)	3	5
T2	10:17	DELETE	STAFF SA9	(old value)		4	6
T2	10:17	UPDATE	PROPERTY PG16	(old value)	(new value)	5	9
T3	10:18	START				0	11
T1	10:18	COMMIT				2	0
	10.19	CHECKPOINT	T2,T3				
T2	10:19	COMMIT				6	0
T3	10:20	INSERT	PROPERTY PG4		(new value)	7	12
T3	10:21	COMMIT				11	0

Recovery Facilities

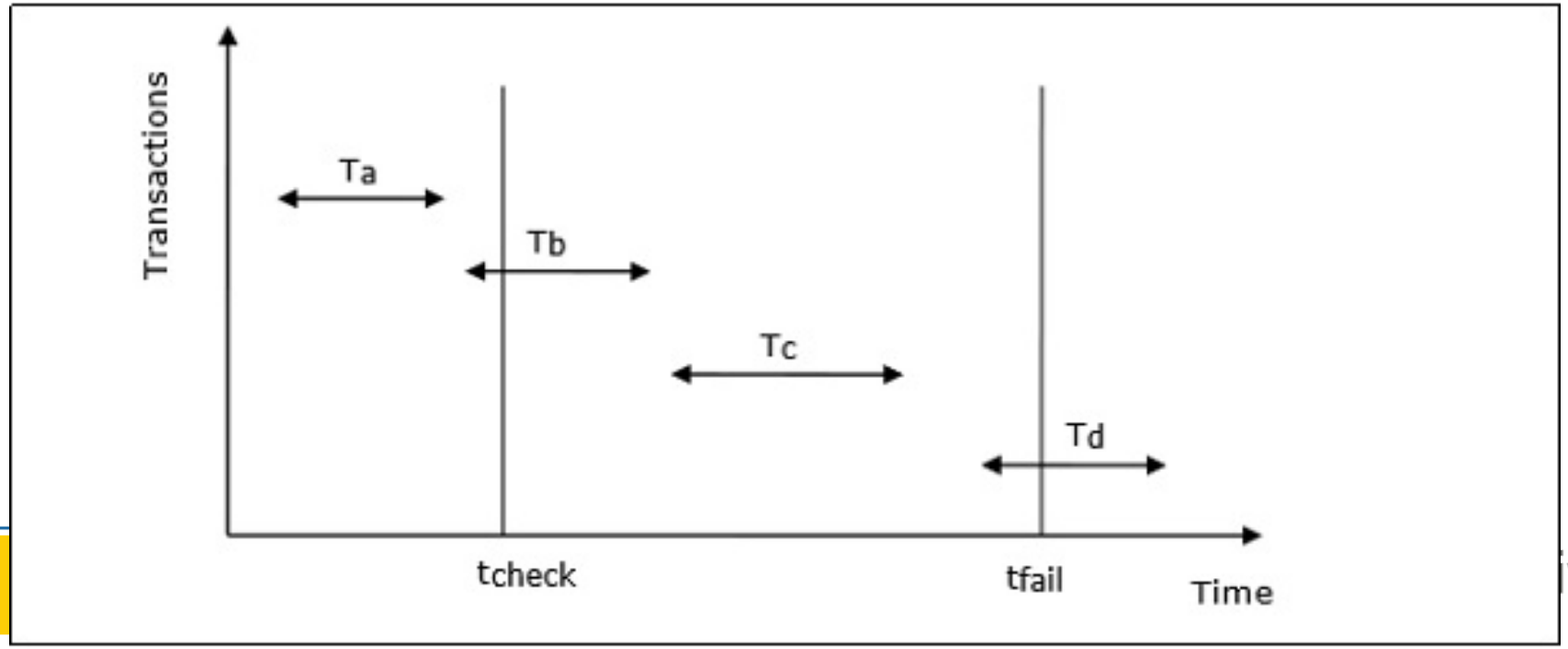
A DBMS should provide the following facilities to assist with recovery:

3. **A checkpoint facility**:

   which enables updates to the database that are in progress to be made permanent;

   The point of synchronization between the database and the transaction log file. All buffers are force-written to secondary storage.

   Checkpoints are scheduled at predetermined intervals.

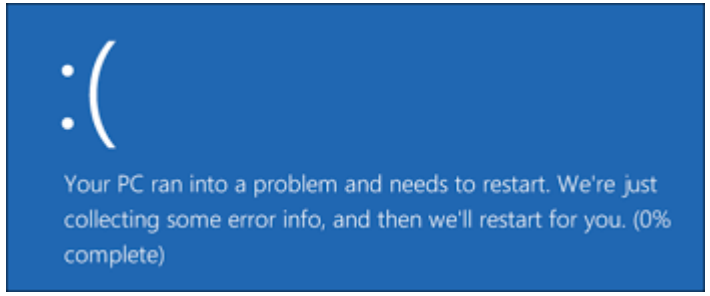


Recovery Techniques

The particular recovery procedure to be used is dependent on the extent of the damage that has occurred to the database.

If the database has been extensively damaged

If the database has not been physically damaged but has become inconsistent



Recovery techniques using deferred update (redo)

1. Updates are not written to the database until after a transaction has reached its commit point.
   - If a transaction fails before it reaches this point → no undoing of changes will be necessary
   - However, it may be necessary to redo the updates of committed transactions as their effect may not have reached the database.

Log entry	Log written to disk	Changes written to database buffer	Changes written on disk
start_transaction(T)	No	N/A	N/A
read_item(T, x)	No	N/A	N/A
write_item(T, x)	No	No	No
commit(T)	Yes	Yes	*Yes
checkpoint	Yes	Undefined	Yes(of committed Ts)

*Yes: writing back to disk may occur not immediately.

- ✓ When a transaction starts, write a transaction start record to the log.
- ✓ When any write operation is performed, write a log record containing all the log data specified previously (excluding the before-image of the update). Do not actually write the update to the database buffers or the database itself.
- ✓ When a transaction is about to commit, write a transaction commit log record, write all the log records for the transaction to disk, and then commit the transaction. Use the log records to perform the actual updates to the database.
- ✓ If a transaction aborts, ignore the log records for the transaction and do not perform the writes.

Recovery techniques using immediate update (redo/undo)

2. Updates are applied to the database as they occur without waiting to reach the commit point.
   - However, these operations are typically recorded in the log on disk by force-writing before they are applied to the database, so that recovery is possible.
   - It may now be necessary to undo the effects of transactions that had not committed at the time of failure.

Log entry	Log written to disk	Changes written to database buffer	Changes written on disk
start_transaction(T)	No	N/A	N/A
read_item(T, x)	No	N/A	N/A
write_item(T, x)	Yes	Yes	*Yes
commit(T)	Yes	Undefined	Undefined
Checkpoint	Yes	Undefined	Yes(of committed Ts)

*Yes: writing back to disk may not occur immediately

- ✓ When a transaction starts, write a transaction start record to the log.
- ✓ When a write operation is performed, write a record containing the necessary data to the log file.
- ✓ Once the log record is written, write the update to the database buffers.
- ✓ The updates to the database itself are written when the buffers are next flushed to secondary storage.
- ✓ When the transaction commits, write a transaction commit record to the log.

Summary

We discussed the need for recovery and the various techniques to recover.

Any Questions