
Accelerated
Database Recovery

What does this session cover?

- What is Accelerated Database Recovery?
- Current Database Recovery Process
- Accelerated Database Recovery Components
- Accelerated Database Recovery Process
- Demonstration

Accelerated Database Recovery

Accelerated Database Recovery is a new SQL Server Engine feature that greatly improves database availability by completely redesigning the current SQL Server recovery process.

Benefits of Accelerated Database Recovery

- Fast & Consistent Database Recovery
- Instantaneous Transaction Rollback
- Aggressive Log Truncation
- Available in Standard Edition

How to enable ADR?

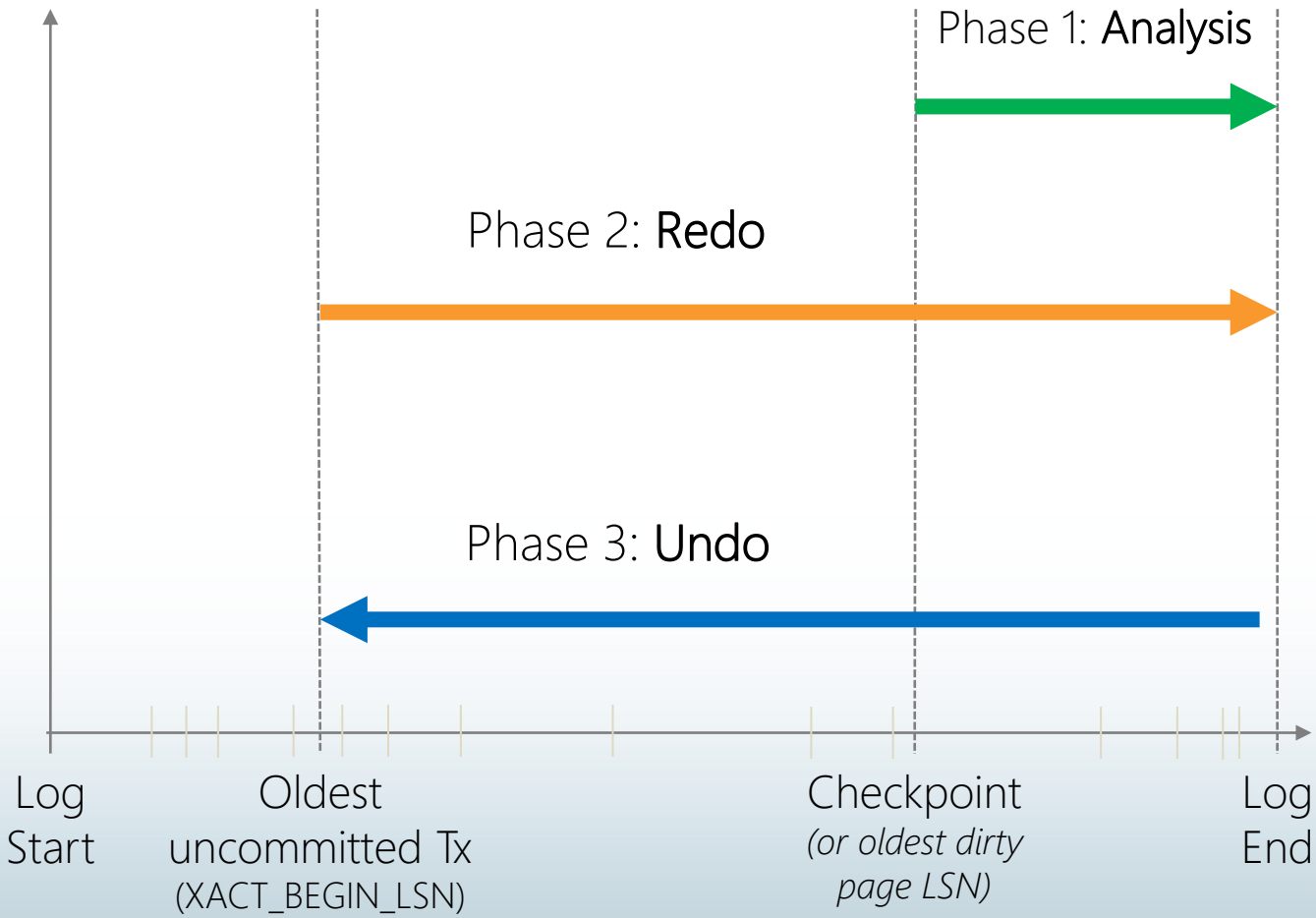
Azure SQL Database

It's ON by default.

SQL Server 2019

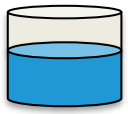
```
ALTER DATABASE <db_name> SET
ACCELERATED_DATABASE_RECOVERY = ON
(PERSISTENT_VERSION_STORE_FILEGROUP = [VersionStoreFG])
```

Current Database Recovery Process



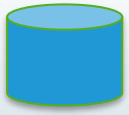
Determines the state of each transaction in the system at the time SQL Server stopped.

Returns the database to the state it was in at the time the SQL Server stopped.



[DB is PARTIALLY available after Redo]

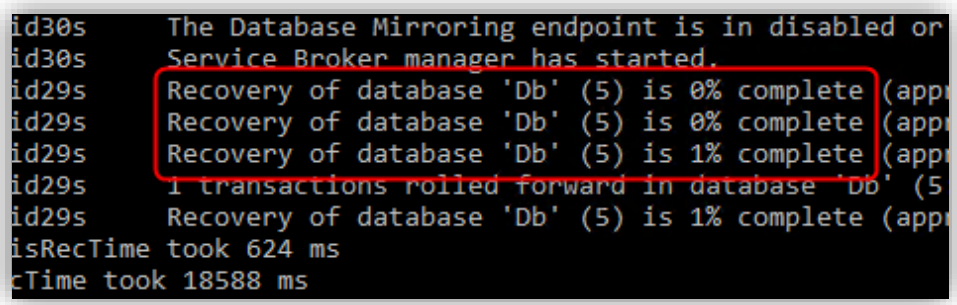
Rolls back any uncommitted transaction(s) individually.



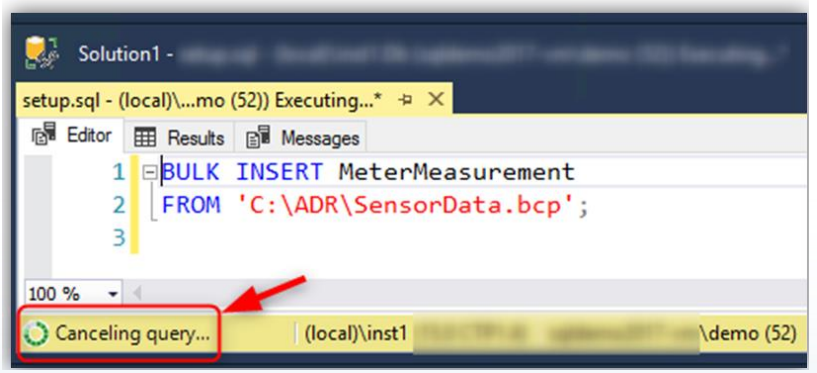
[DB is FULLY available after Undo]

Most common implications

- Recovery time is roughly proportional to the longest running transaction.



- Rolling back large batch operations (such as bulk insert) takes a long time.



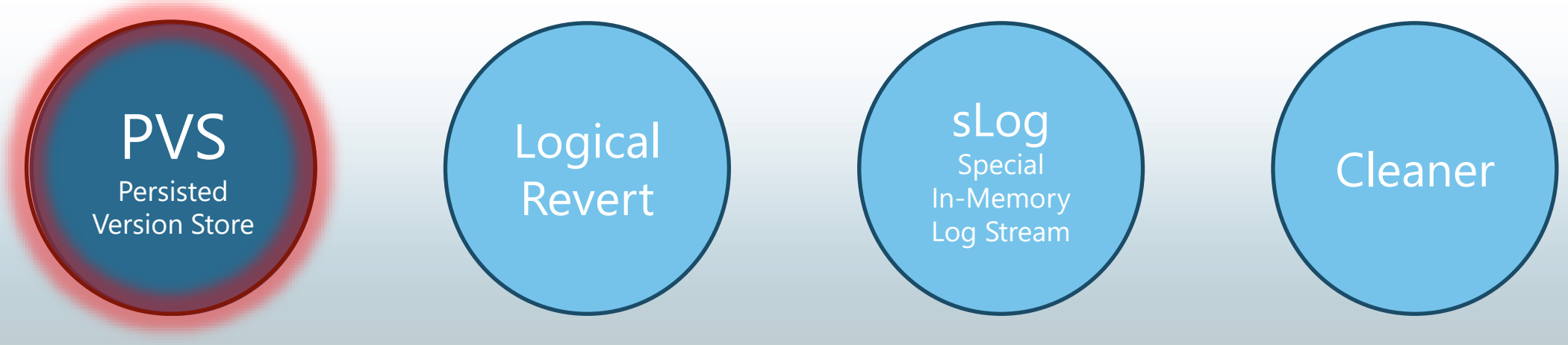
- Transaction log may run out of space during long-running transactions.

```
Error 9002, Severity: 17, State: 4
The transaction log for database 'db' is full.
To find out why space in the log cannot be reused, see the log_reuse_wait_desc column in sys.databases
```

Accelerated Database Recovery Components

Persisted Version Store

- Persists row versions in the database itself rather than TempDB.
- The version can be stored in-row or off-row within the database, it will vary according to the row size;
- Versions have the previous state of the data and the Transact-ID of the version;
- Fast UNDO, instead of rolling back the active transactions (traditional recovery process) the row version is marked as Terminated.



Accelerated Database Recovery Components

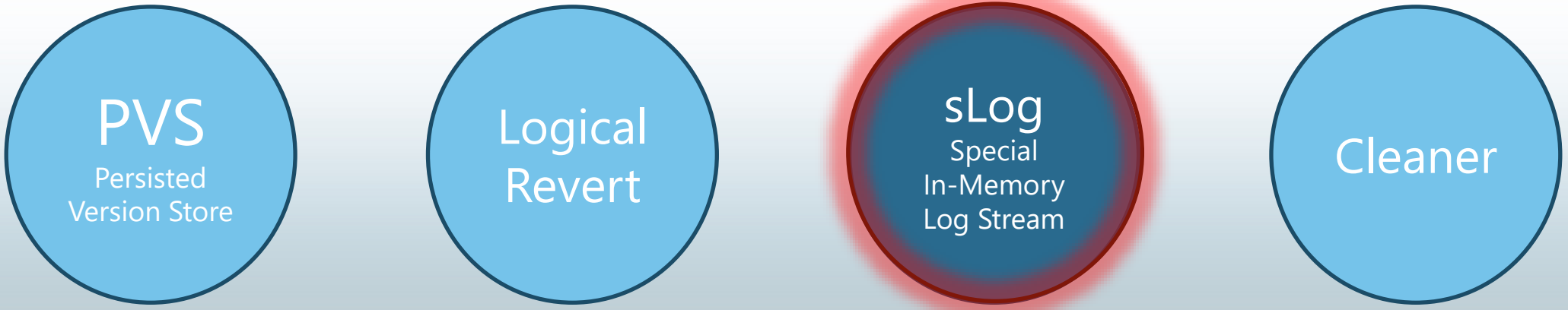
Logical Revert

- Asynchronous process that performs row-level version-based Undo;
- Keeps track of all terminated transactions;
- Performs rollback using recent committed transactions from PVS;
- Release all locks immediately after transaction termination.

Accelerated Database Recovery Components

sLog

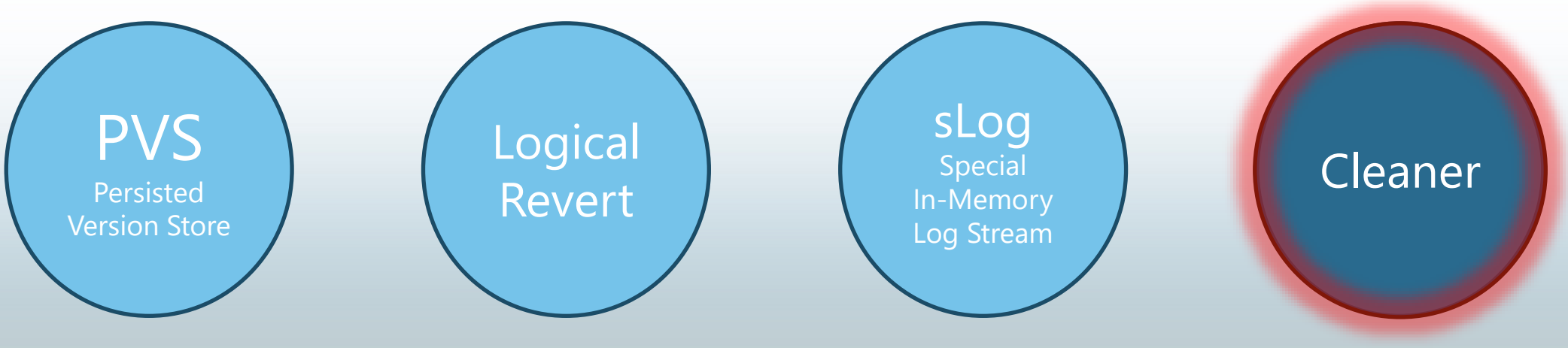
- Secondary in-memory log stream that stores log records for non-versioned operations (e.g.: metadata cache invalidation, lock acquisitions);
- Persisted on disk by been serialized during SQL checkpoint;
- Is periodically truncated as transactions commits;
- It accelerates the redo and undo by processing only the non-versioned operations;



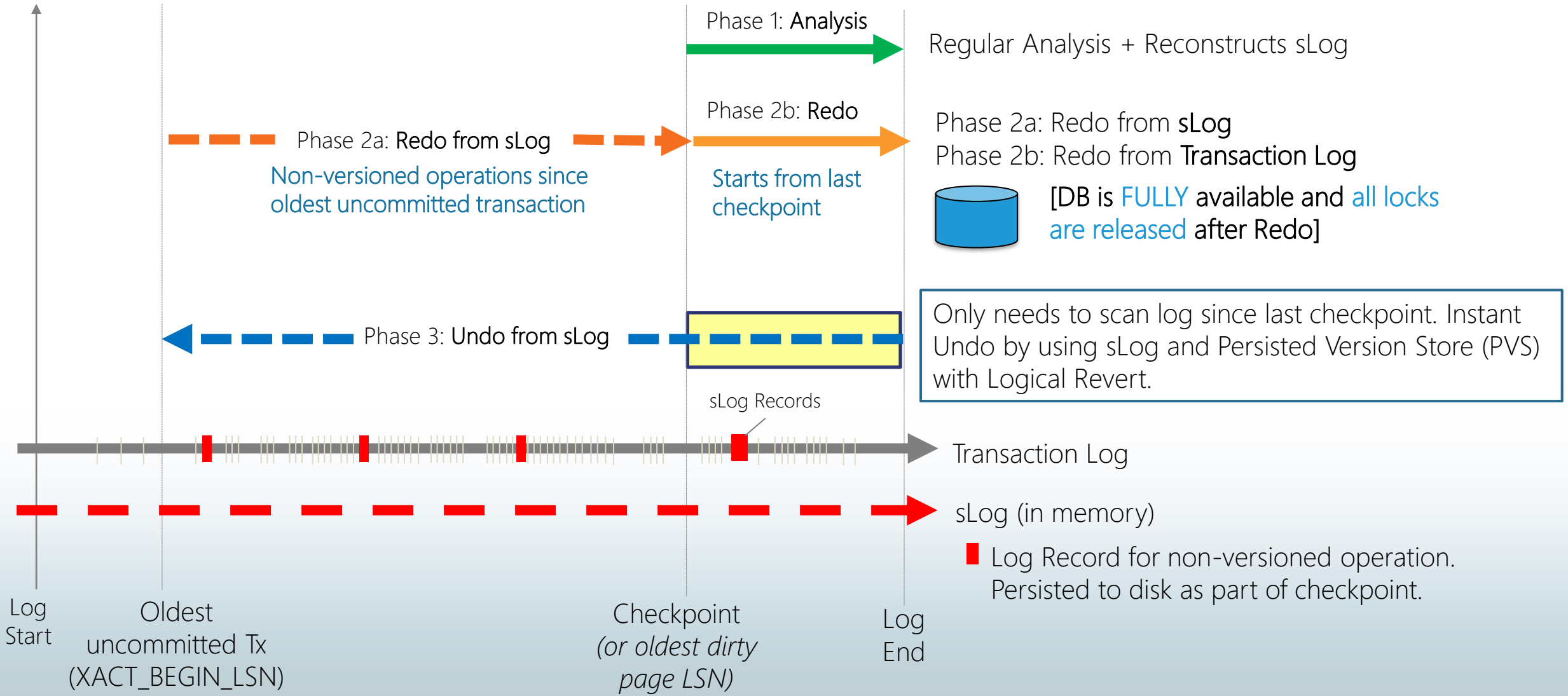
Accelerated Database Recovery Components

Cleaner

- Asynchronous process that periodically cleans row versions that are not needed.

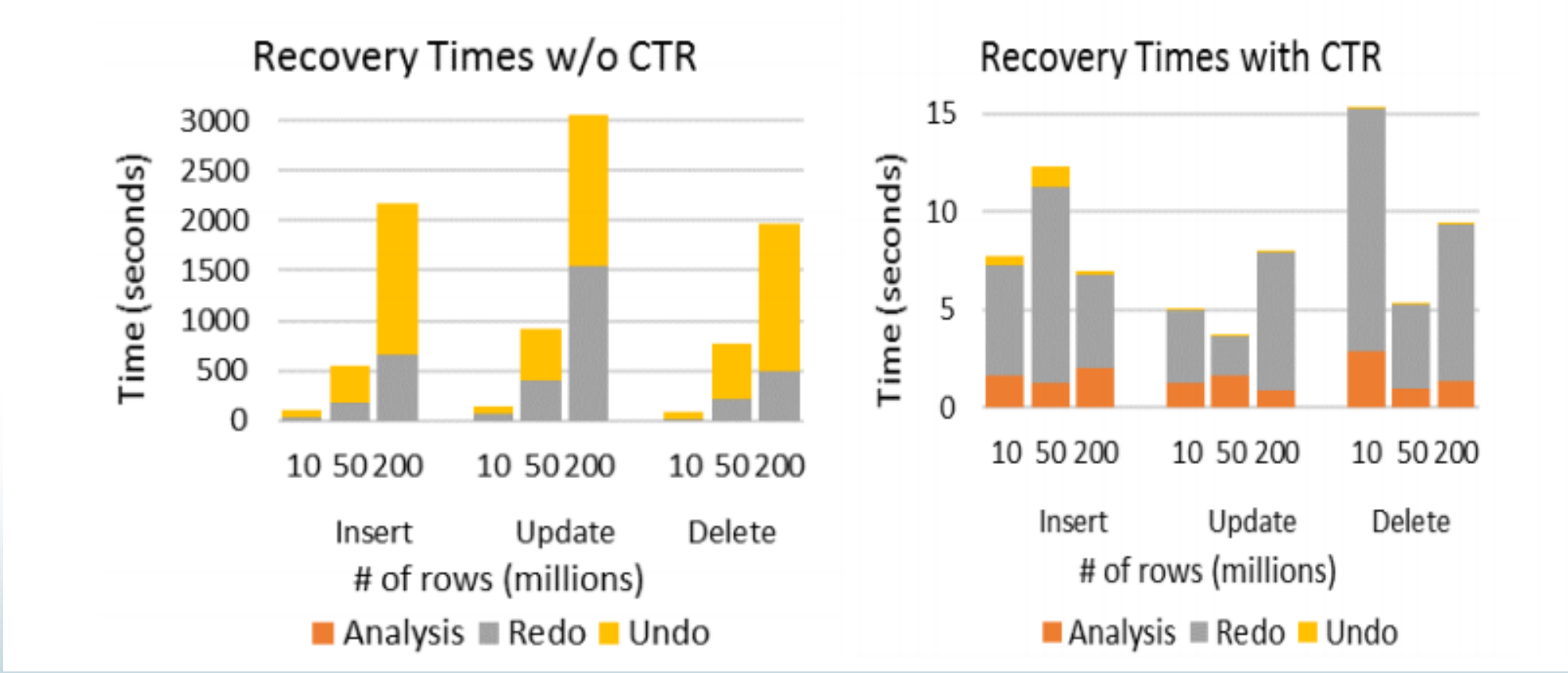


Accelerated Database Recovery process



Recovery Time Comparison

[Constant Time Recovery in SQL Server]



Accelerated Dabase Recovery FAQ

Will my database be larger?

- Yes. Monitor to determine difference.
- According to the CTR whitepaper, 50 million modifications add 1GB to database.

Will it affect performance?

- It depends. Write-heavy (OLTP) workloads are most susceptible.
- According to the CTR whitepaper, 13.8% utilization for Update heavy workloads, 2.4% for normal workloads.

How is PVS different than the version store in TempDB?

- PVS stores versions in the user database rather than TempDB
- If ADR is enabled, PVS is used to support SNAPSHOT and READ_COMMITTED_SNAPSHOT_ISOLATION transactions

How does this affect Availability Groups?

- PVS and log records replicate to secondaries, secondary communicates oldest versions needed to primary
- ADR can speed up failover because Undo becomes fast
- If the secondary must be restarted without ADR, TempDB is lost so versions are lost and queries must wait for data to commit on primary, with ADR, versions are persisted, so no delay before queries can be served

Demo Time

Questions?