



# Introduction to Accelerated Database Recovery

John Deardurff





## John Deardurff

Microsoft Customer Engineer (Global Technical Team)

Microsoft Certified Trainer (Regional Lead)

MVP: Data Platform (2016 – 2018)

Email: [John.Deardurff@Microsoft.com](mailto:John.Deardurff@Microsoft.com)

Twitter: [@SQLMCT](https://twitter.com/SQLMCT)

Website: [www.SQLMCT.com](http://www.SQLMCT.com)

GitHub: [github.com/SQLMCT](https://github.com/SQLMCT)



# 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
- Reduces contention on TempDB database.

# How to enable ADR?

Azure SQL Database

It's ON by default.

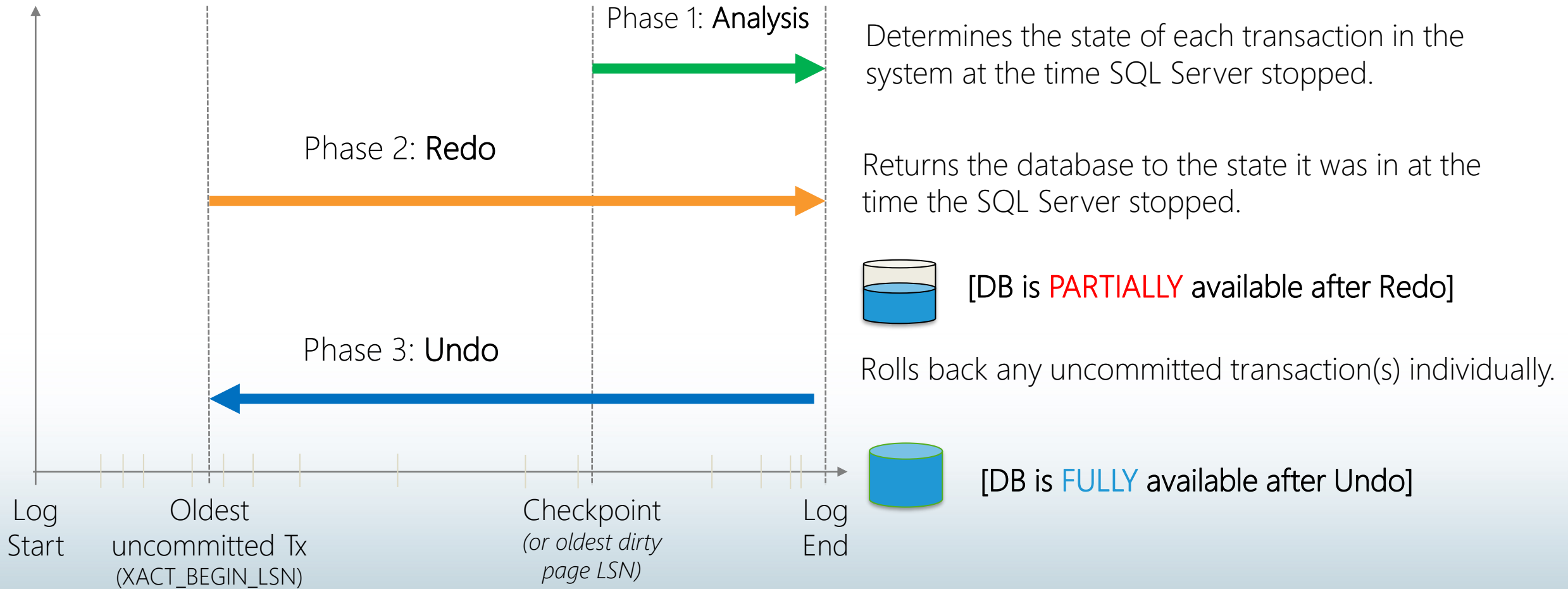
SQL Server 2019

```
ALTER DATABASE <db_name> SET
```

```
ACCELERATED_DATABASE_RECOVERY = ON
```

```
(PERSISTENT_VERSION_STORE_FILEGROUP = [VersionStoreFG])
```

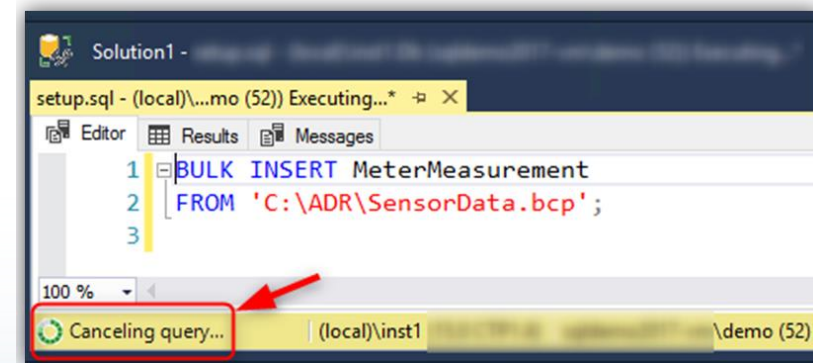
# Current Database Recovery Process (ARIES)



# 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.

```
id30s The Database Mirroring endpoint is in disabled or
id30s Service Broker manager has started.
id29s Recovery of database 'Db' (5) is 0% complete (app
id29s Recovery of database 'Db' (5) is 0% complete (app
id29s Recovery of database 'Db' (5) is 1% complete (app
id29s 1 transactions rolled forward in database 'Db' (5
id29s Recovery of database 'Db' (5) is 1% complete (app
isRecTime took 624 ms
cTime took 18588 ms
```



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.
- 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.



**PVS**  
Persisted  
Version Store

Logical  
Revert

sLog  
Special  
In-Memory  
Log Stream

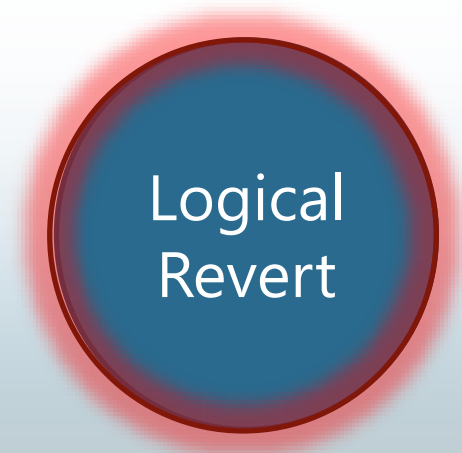
Cleaner



# Accelerated Database Recovery Components

## Logical Revert

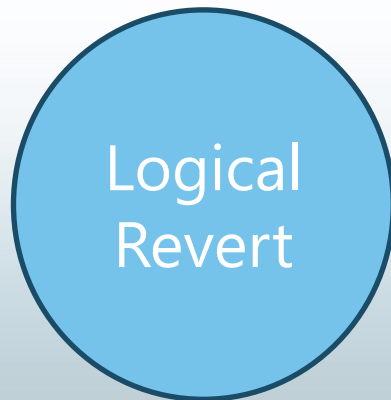
- Asynchronous process that performs row-level version-based Undo;
- Keeps track of all terminated transactions;
- Performs rollback using recent committed transactions from PVS;
- Release all locks immediately after transaction termination.



# Accelerated Database Recovery Components

## sLog

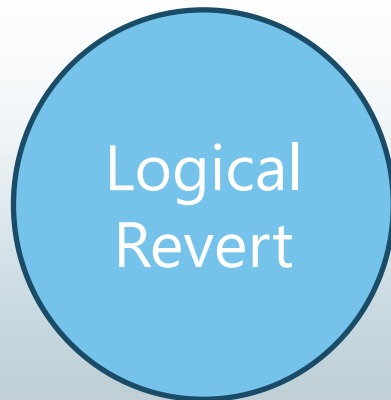
- Secondary in-memory log stream that stores log records for non-versioned operations (Example: Bulk operations, Lock acquisitions, etc...);
- Persisted on disk by been serialized during SQL checkpoint;
- Is periodically truncated as transactions commits;
- It accelerates the redo and undo by processing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



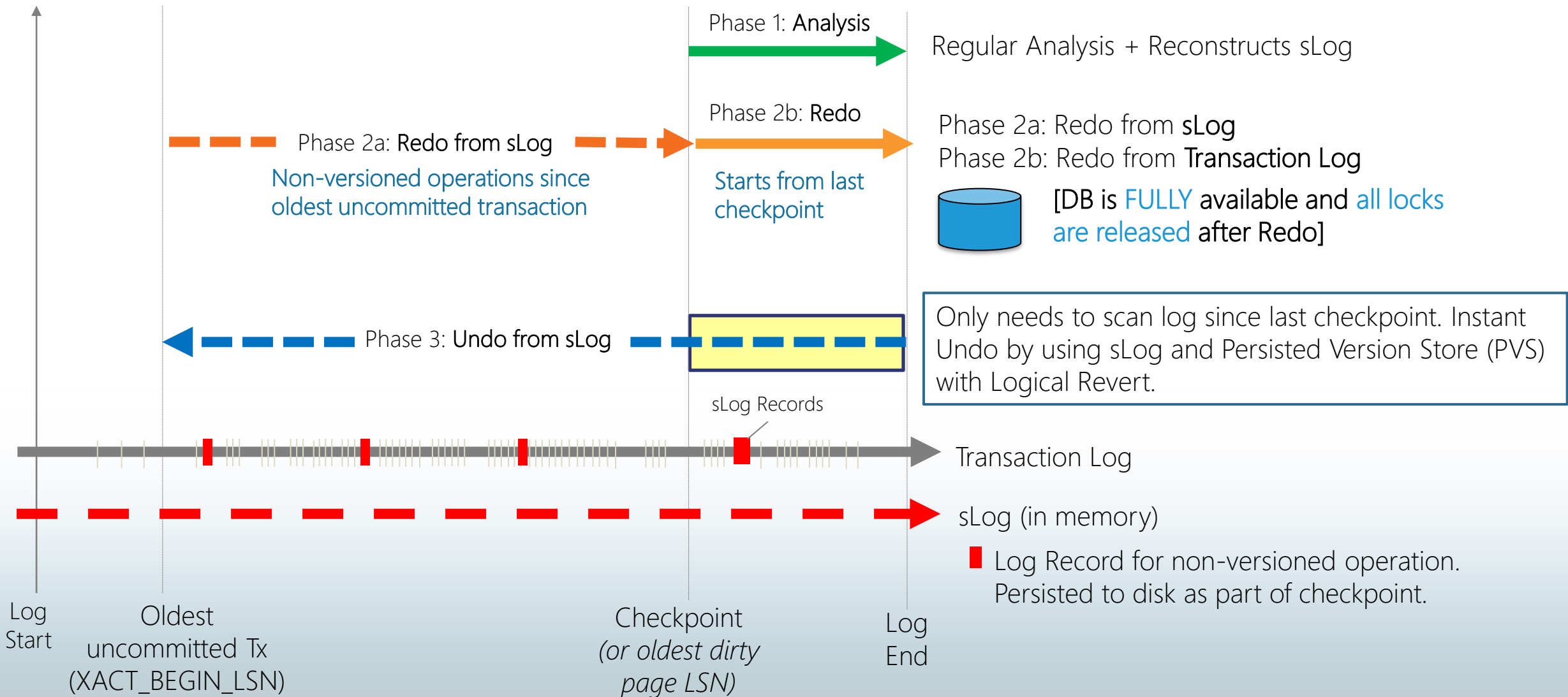
# Accelerated Database Recovery Components

## Cleaner

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

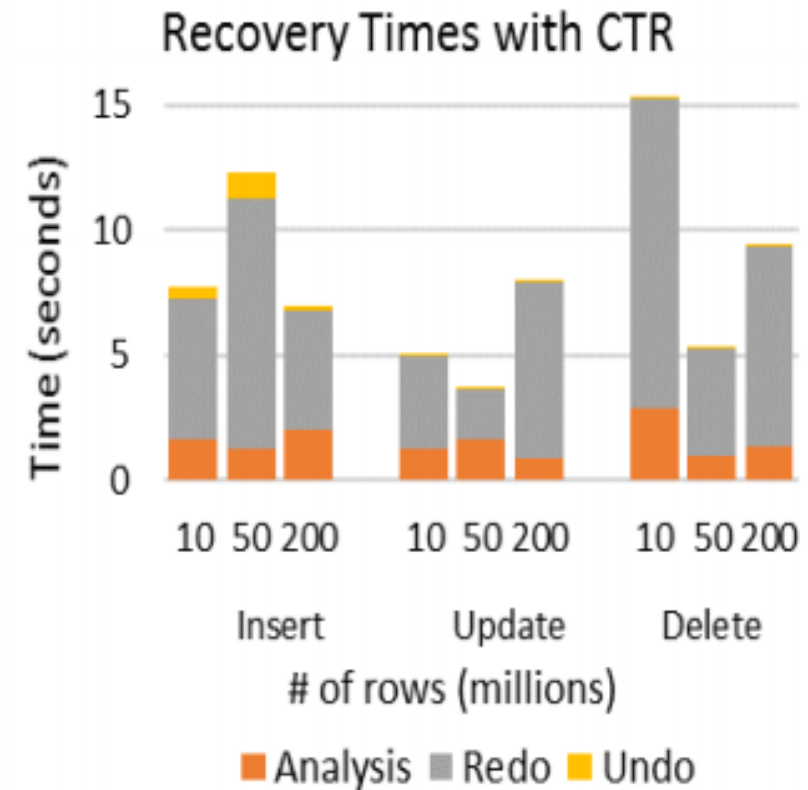
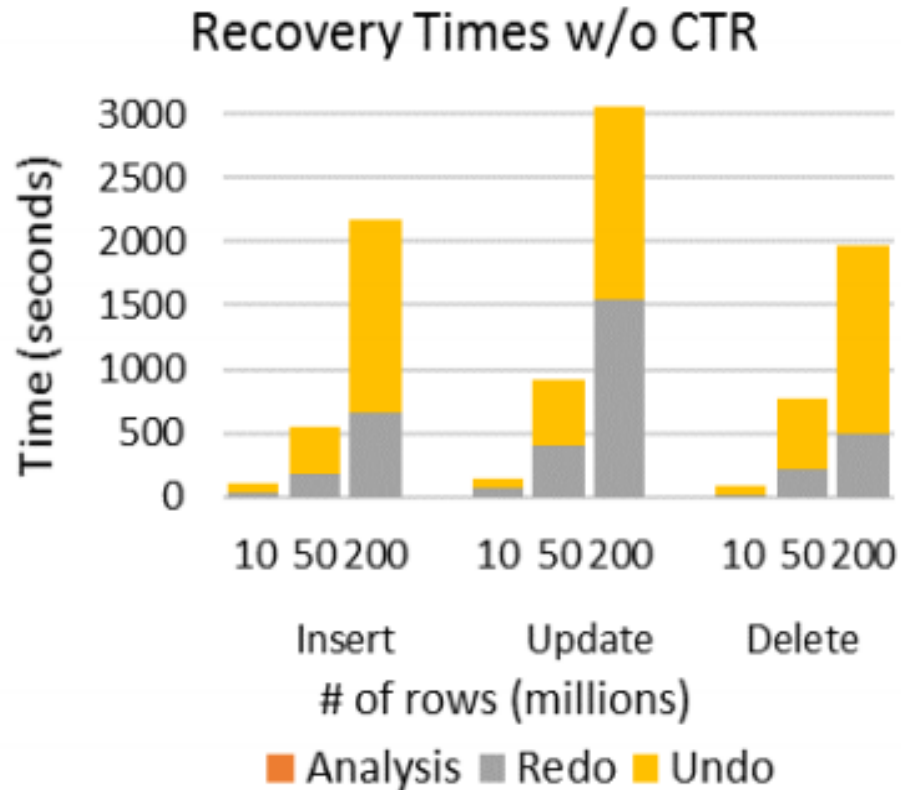


# Accelerated Database Recovery process



# Recovery Time Comparison

## Constant Time Recovery in SQL Server



# Accelerated Database 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?