



# Multi-Version Concurrency Control (MVCC)

February 2010

**INGRES**<sup>®</sup>

# MVCC – Why Use It?

Time

## Update Transaction

- Updates row A in table X
- Updates row B in table Y
- Commits transaction

## Read Transaction with Traditional Locking

- Attempts to read row A in table X

*BLOCKED because read cannot acquire lock held by update!*

- Read from table X finally succeeds after update has committed

## Read Transaction with MVCC Locking

- Successfully reads row A from table X
- Successfully reads row B from table Y

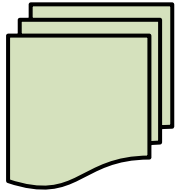
*Data read is pre-update, so no locking required*

*No application hangs → Better performance!*

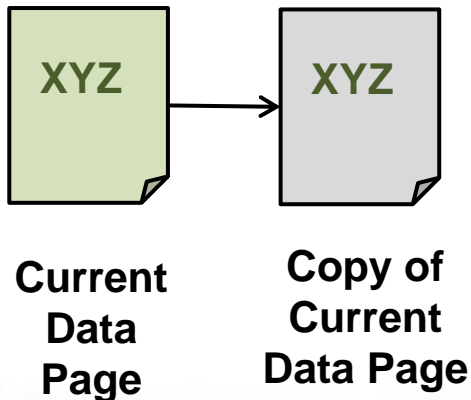
- With MVCC, readers are no longer blocked by writers
- Likewise, writers are no longer blocked by readers

# MVCC – How it Works

- ~~Approach 1 – Store multiple copies of data pages and/or records~~
- Approach 2 – Reconstruct older versions of data dynamically, using log records



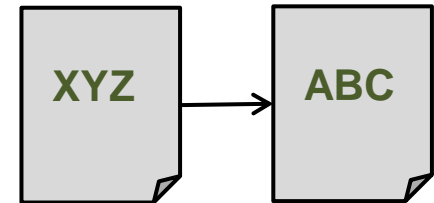
Step 1: Read desired data page and make a copy of the page



Step 2: Locate log records corresponding to uncommitted modifications made to the desired page

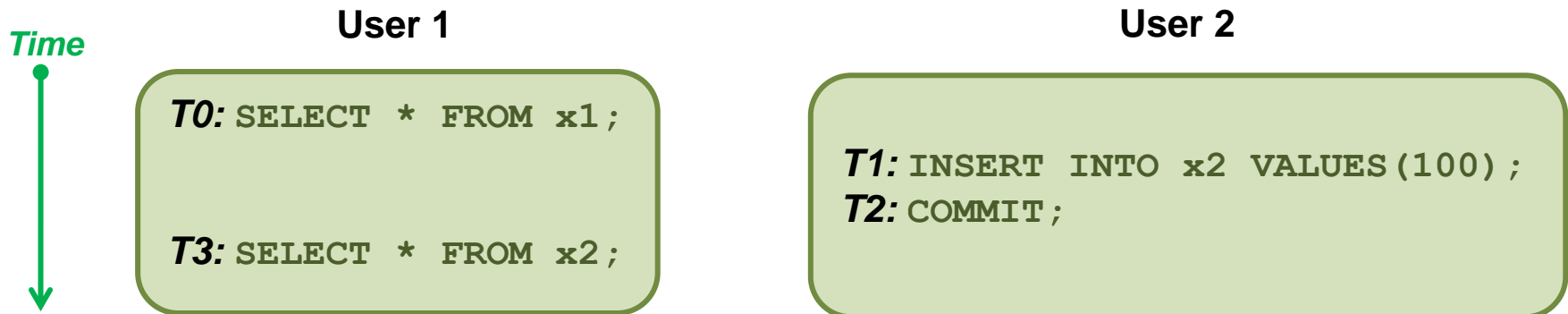


Step 3: Use log records to “undo” uncommitted updates made to the data page. I.e., roll back to some consistent snapshot in the past.



# MVCC – What Data Does the User See?

- It depends on their isolation level setting



Isolation Level	Read Consistency Level	Snapshot Corresponds To	SELECT from x2 By User 1
Serializable	Transaction	Start of transaction (T0)	<b>Does not</b> see new value
Read Committed	Statement	Start of statement (T2)	<b>Does</b> see new value

- With traditional locking, User 1 will see new value

# MVCC – Beware of Application Consistency Issues

## Before Updates:

Table X	Key	Col	Table Y	Key	Col
	1	\$1000		1	\$5000

## After Updates:

Table X	Key	Col	Table Y	Key	Col
	1	\$900		1	\$5100

Time



```
T1: UPDATE X SET Col=Col-100 WHERE Key=1;
T3: UPDATE Y SET Col=Col+100 WHERE Key=1;
T4: COMMIT;
```

```
T0: { start of transaction }
T2: SELECT Col FROM X
WHERE Key=1;
T5: SELECT Col FROM Y
WHERE Key=1;
T6: COMMIT;
```

Isolation Level	Read Consistency Level	SELECT from X Returns	SELECT from Y Returns	X.Col + Y.Col
Serializable	Transaction	\$1000	\$5000	\$6000
Read Committed	Statement	\$1000	\$5100	\$6100



**Inconsistent!**

**NOTE: With traditional locking, selects will return \$900 and \$5100**