Recovery

During Normal Processing
State of Transaction

- **Active**
  - Begin
  - Read, Write
  - Ask commit
  - abort

- **Partially Committed**
  - Commit
  - Grant commit
  - abort Dirty read

- **Committed**
- **Failed**
- **Done**

Own choice to quit, dies because of deadlock policy or serialization policy.
Deferred Update Abort

• Deferred Update Policy:
  – Data changed by transaction not available to others until commit
  – Strict 2PL (hold locks till end of transaction)
  – Data versioning (other transactions read earlier data)

• No UNDO needed
  – Changed data never available.
Immediate Update Abort

• Immediate Update Policy
  – Data changed by transaction available to others before commit
  – Ordinary 2PL
  – Optimistic protocols

• UNDO needed to restore original data

• Dirty reads may abort other transactions:
  Cascading Rollback
**Immediate Update Example**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

```
start, \textcolor{red}{T3}
read, \textcolor{red}{T3}, C
write, \textcolor{red}{T3}, B, 15, 12 12
start, \textcolor{red}{T2}
read, \textcolor{red}{T2}, B
write, \textcolor{red}{T2}, B, 12, 18 18
start, \textcolor{red}{T1}
read, \textcolor{red}{T1}, A
read, \textcolor{red}{T1}, D
write, \textcolor{red}{T1}, D, 20, 25 25

read, \textcolor{red}{T2}, D
write, \textcolor{red}{T2}, D, 25, 26 26

read, \textcolor{red}{T3}, A
write, \textcolor{red}{T3}, A, 30, 35 35
```
# Immediate Update Example

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>T3</td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

start, \( T_3 \)
read, \( T_3, C \)
write, \( T_3, B, 15, 12 \)

start, \( T_2 \)
read, \( T_2, B \)
write, \( T_2, B, 12, 18 \)

start, \( T_1 \)
read, \( T_1, A \)
read, \( T_1, D \)
write, \( T_1, D, 20, 25 \)

read, \( T_2, D \)
write, \( T_2, D, 25, 26 \)

read, \( T_3, A \)
write, \( T_3, A, 30, 35 \)

Available before commit
Immediate Update Example

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{start}, & \quad T3 \\
\text{read}, & \quad T3, C \\
\text{write}, & \quad T3, B, 15, 12 \\
\text{start}, & \quad T2 \\
\text{read}, & \quad T2, B \quad \text{Dirty Read} \\
\text{write}, & \quad T2, B, 12, 18 \\
\text{start}, & \quad T1 \\
\text{read}, & \quad T1, A \\
\text{read}, & \quad T1, D \\
\text{write}, & \quad T1, D, 20, 25 \\
\text{read}, & \quad T2, D \\
\text{write}, & \quad T2, D, 25, 26 \\
\text{read}, & \quad T3, A \\
\text{write}, & \quad T3, A, 30, 35, 35 \quad \text{Available before commit}
\end{align*}
\]
Immediate Update Example

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

start, T3
read, T3, C
write, T3, B, 15, 12
start, T2
read, T2, B Dirty Read
write, T2, B, 12, 18
start, T1
read, T1, A
read, T1, D
write, T1, D, 20, 25

T1 asks and gets commit here--no one else wrote A or D
read, T2, D
write, T2, D, 25, 26

read, T3, A
write, T3, A, 30, 35

Available before commit
## Immediate Update Example

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

### Transactions

- **T1** asks and gets commit here—no one else wrote A or D
- **T2** asks but must wait because of dirty read

### Operations

- **start**, **T3**
- **read**, **T3**, **C**
- **write**, **T3**, **B**, **15**, **12**
- **start**, **T2**
- **read**, **T2**, **B** [Dirty Read]
- **write**, **T2**, **B**, **12**, **18**
- **start**, **T1**
- **read**, **T1**, **A**
- **read**, **T1**, **D**
- **write**, **T1**, **D**, **20**, **25**

### Notes

- Available before commit
- T1 asks and gets commit here—no one else wrote A or D
- T2 asks but must wait because of dirty read
- T3 asks and gets commit here—no one else wrote A or D
## Immediate Update Example

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

```
start, T3
read,  T3, C
write, T3, B, 15, 12
start, T2
read,  T2, B  Dirty Read
write, T2, B, 12, 18
start, T1
read,  T1, A
read,  T1, D
write, T1, D, 20, 25
```

T1 asks and gets commit here--no one else wrote A or D
```
read,  T2, D
write, T2, D, 25, 26
```

T2 asks but must wait because of dirty read
```
read,  T3, A
write, T3, A, 30, 35
```

If T3 commits, dirty read is ok & T2 can then commit
```
write, T3, A, 30, 35
```
## Cascading Rollback

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

- **T3** starts reading, then writes to **B** at 15, 12
- **T2** starts reading, then writes to **B** at 12, 18
- **T1** starts reading, then reads from **A** and **D**, writes to **D** at 20, 25
  - **T1 asks and gets commit here**
  - **T2** must wait due to dirty read
- **T2** reads from **D** at 25, 26
- **T3** reads from **A** and writes to **A** at 30, 35

The sequence illustrates how cascading rollback occurs when dependencies are not properly managed, leading to conflicts and the need for rollbacks at higher transactions.
Cascading Rollback

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

\[\begin{align*}
\text{start, } & T3 \\
\text{read, } & T3, C \\
\text{write, } & T3, B, 15, 12 \\
\text{start, } & T2 \\
\text{read, } & T2, B \\
\text{write, } & T2, B, 12, 18 \\
\text{start, } & T1 \\
\text{read, } & T1, A \\
\text{read, } & T1, D \\
\text{write, } & T1, D, 20, 25 \\
\text{T1 asks and gets commit here} \\
\text{read, } & T2, D \\
\text{write, } & T2, D, 25, 26 \\
\text{T2 asks but must wait because of dirty read} \\
\text{read, } & T3, A \\
\text{write, } & T3, A, 30, 35 \\
\text{T3 aborts!}
\end{align*}\]
Cascading Rollback

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1:</td>
<td>start,</td>
<td>T1,</td>
<td>A,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T1,</td>
<td>A,</td>
<td>20,</td>
</tr>
<tr>
<td>T2:</td>
<td>start,</td>
<td>T2,</td>
<td>B,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T2,</td>
<td>B,</td>
<td>12,</td>
</tr>
<tr>
<td>T3:</td>
<td>start,</td>
<td>T3,</td>
<td>B,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T3,</td>
<td>B,</td>
<td>15,</td>
</tr>
<tr>
<td>T2:</td>
<td>start,</td>
<td>T2,</td>
<td>B,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T2,</td>
<td>B,</td>
<td>12,</td>
</tr>
<tr>
<td>T1:</td>
<td>start,</td>
<td>T1,</td>
<td>A,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T1,</td>
<td>A,</td>
<td>12,</td>
</tr>
<tr>
<td>T1:</td>
<td>start,</td>
<td>T1,</td>
<td>D,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T1,</td>
<td>D,</td>
<td>20,</td>
</tr>
<tr>
<td>T2:</td>
<td>start,</td>
<td>T2,</td>
<td>D,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T2,</td>
<td>D,</td>
<td>25,</td>
</tr>
<tr>
<td>T3:</td>
<td>start,</td>
<td>T3,</td>
<td>A,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T3,</td>
<td>A,</td>
<td>30,</td>
</tr>
<tr>
<td>T3:</td>
<td>start,</td>
<td>T3,</td>
<td>A,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T3,</td>
<td>A,</td>
<td>30,</td>
</tr>
<tr>
<td>T3:</td>
<td>start,</td>
<td>T3,</td>
<td>A,</td>
<td>write,</td>
</tr>
<tr>
<td></td>
<td>read,</td>
<td>T3,</td>
<td>A,</td>
<td>30,</td>
</tr>
</tbody>
</table>

T1 asks and gets commit here
T2 asks but must wait because of dirty read
T3 ABORTS!
Cascading Rollback

\[
\begin{array}{cccc}
A & B & C & D \\
30 & 15 & 40 & 20 \\
\end{array}
\]

\begin{align*}
\text{start}, & \quad T3 \\
\text{read}, & \quad T3, C \\
\text{write}, & \quad T3, B, 15, 12 \\
\text{start}, & \quad T2 \\
\text{read}, & \quad T2, B \\
\text{write}, & \quad T2, B, 12, 18 \\
\text{start}, & \quad T1 \\
\text{read}, & \quad T1, A \\
\text{read}, & \quad T1, D \\
\text{write}, & \quad T1, D, 20, 25
\end{align*}

T1 asks and gets commit here

\begin{align*}
\text{read}, & \quad T2, D \\
\text{write}, & \quad T2, D, 25, 26
\end{align*}

T2 asks but must wait because of dirty read

\begin{align*}
\text{read}, & \quad T3, A \\
\text{write}, & \quad T3, A, 30, 35
\end{align*}

T3-aborts!
Cascading Rollback

```
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>
```

start, $T_3$
read, $T_3$, $C$
write, $T_3$, $B$, 15, 12

start, $T_2$
read, $T_2$, $B$
write, $T_2$, $B$, 12, 18

start, $T_1$
read, $T_1$, $A$
read, $T_1$, $D$
write, $T_1$, $D$, 20, 25

$T_1$ asks and gets commit here
read, $T_2$, $D$
write, $T_2$, $D$, 25, 26

$T_2$ asks but must wait because of dirty read
read, $T_3$, $A$
write, $T_3$, $A$, 30, 35

$T_3$ ABORTS!

$T_2$ ABORTS -- dirty read
Cascading Rollback

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{start, } & T3 \\
\text{read, } & T3, C \\
\text{write, } & T3, B, 15, 12 \\
\text{start, } & T2 \\
\text{read, } & T2, B \\
\text{write, } & T2, B, 12, 18 \\
\text{start, } & T1 \\
\text{read, } & T1, A \\
\text{read, } & T1, D \\
\text{write, } & T1, D, 20, 25 \\
\text{T1 asks and gets commit here} & \\
\text{read, } & T2, D \\
\text{write, } & T2, D, 25, 26 \\
\text{T2 asks but must wait because of dirty read} & \\
\text{read, } & T3, A \\
\text{write, } & T3, A, 30, 35 \\
\text{T3 ABORTS!} & \\
\text{T2 ABORTS -- dirty read} &
\end{align*}
\]
Cascading Rollback

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>30</td>
<td>15</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

start, T3
read,  T3, C
write, T3, B, 15, 12

start, T2
read,  T2, B
write, T2, B, 12, 18

start, T1
read,  T1, A
read,  T1, D
write, T1, D, 20, 25

T1 asks and gets commit here
read,  T2, D
write, T2, D, 25, 26

T2 asks but must wait because of dirty read
read,  T3, A
write, T3, A, 30, 35

T3 ABORTS! T2 ABORTS -- dirty read
Summary

• Rollback needed in case of abort with immediate updates
• Cascading rollback needed if dirty read happened
  – Depends on update policy of the database.
• Log used for UNDO only
• No REDO for normal processing