Database Foundations

6-5
Transaction Control Language (TCL)
Roadmap

- Introduction to Oracle Application Express
- Structured Query Language (SQL)
- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Transaction Control Language (TCL)

- Retrieving Data Using SELECT
- Restricting Data Using WHERE
- Sorting Data Using ORDER BY
- Joining Tables Using JOIN

You are here: DFO 6-5
Objectives

This lesson covers the following objectives:

• Describe the purpose of the Transaction Control Language (TCL)

• Explain the TCL operations that are required to manage a transaction:
  – COMMIT
  – SAVEPOINT
  – ROLLBACK

• Describe the need for read consistency
Database Transactions

A database transaction consists of one of the following statements:

- DML statements that represent one consistent change to the data
- One DDL statement
- One TCL statement
Database Transactions: Start and End

• Begin when the first DML SQL statement is executed.

• End with one of the following events:
  – A **COMMIT** or **ROLLBACK** statement is issued.
  – A DDL or TCL statement executes (automatic commit).
  – The user exits SQL Developer or SQL*Plus.
  – The system crashes.
Advantages of COMMIT and ROLLBACK Statements

With COMMIT and ROLLBACK statements, you can:

• Ensure data consistency
• Preview data changes before making changes permanent
• Group logically related operations
## Transaction Control Statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMIT</td>
<td>Ends the current transaction by making all pending data changes permanent.</td>
</tr>
<tr>
<td>SAVEPOINT name</td>
<td>Marks a savepoint within the current transaction.</td>
</tr>
<tr>
<td>ROLLBACK</td>
<td>Ends the current transaction by discarding all pending data changes.</td>
</tr>
<tr>
<td>ROLLBACK TO SAVEPOINT name</td>
<td>Rolls back the current transaction to the specified savepoint, thereby discarding any changes and/or savepoints that were created after the savepoint to which you are rolling back. If you omit the TO SAVEPOINT clause, the ROLLBACK statement rolls back the entire transaction. Because savepoints are logical, there is no way to list the savepoints that you created.</td>
</tr>
</tbody>
</table>
Explicit Transaction Control Statements

Transaction

TIME

COMMIT

SAVEPOINT A

DELETE FROM copy_emp;

INSERT INTO dept VALUES (80, 'Marketing', 400, 1900);

UPDATE emp
SET dept_id = 110;

SAVEPOINT B

INSERT ...

ROLLBACK to SAVEPOINT B
ROLLBACK to SAVEPOINT A
ROLLBACK
Rolling Back Changes to a Marker

• Create a marker in the current transaction by using the SAVEPOINT statement.

• Roll back to that marker by using the ROLLBACK TO SAVEPOINT statement.

```
UPDATE...
SAVEPOINT update_done;
SAVEPOINT update_done succeeded.
INSERT...
ROLLBACK TO update_done;
ROLLBACK TO succeeded.
```
Implicit Transaction Processing

• An automatic commit occurs under the following circumstances:
  – A DDL statement is issued.
  – A TCL statement is issued.
  – There is a normal exit from SQL Developer or SQL*Plus, without explicitly issuing `COMMIT` or `ROLLBACK` statements

• An automatic rollback occurs when there is an abnormal termination of SQL Developer or SQL*Plus or when there is a system failure.
State of the Data Before COMMIT or ROLLBACK

• The previous state of the data can be recovered.
• The current session can review the results of the DML operations by using the `SELECT` statement.
• Other sessions cannot view the results of the DML statements issued by the current session.
• The affected rows are locked; other sessions cannot change the data in the affected rows.
State of the Data After COMMIT

• Data changes are saved in the database.
• The previous state of the data is overwritten.
• All sessions can view the results.
• Locks on the affected rows are released; those rows are available for other sessions to manipulate.
• All savepoints are erased.
Committing Data

• Make the changes:

DELETE FROM EMPLOYEES
WHERE employee_id=113;
1 rows deleted

INSERT INTO departments
VALUES (290, 'Corporate Tax', NULL, 1700);
1 rows inserted

• Commit the changes:

COMMIT;
committed.
State of the Data After ROLLBACK

Discard all pending changes by using the ROLLBACK statement:

• Data changes are undone.
• Previous state of the data is restored.
• Locks on the affected rows are released.

```
DELETE FROM copy_emp;
ROLLBACK ;
```
State of the Data After ROLLBACK: Example

```sql
DELETE FROM test;
4 rows deleted.

ROLLBACK;
Rollback complete.

DELETE FROM test WHERE  id = 100;
1 row deleted.

SELECT * FROM   test WHERE  id = 100;
No rows selected.

COMMIT;
Commit complete.
```
Statement-Level Rollback

• If a single DML statement fails during execution, only that statement is rolled back.
• The Oracle server implements an implicit savepoint.
• All other changes are retained.
• The user should terminate transactions explicitly by executing a \texttt{COMMIT} or a \texttt{ROLLBACK} statement.
Read Consistency

• Read consistency guarantees a consistent view of the data at all times.

• Changes made by one user do not conflict with the changes made by another user.

• Read consistency ensures that, on the same data:
  – Readers do not wait for writers.
  – Writers do not wait for readers.
  – Writers wait for writers.
Implementing Read Consistency

User A

UPDATE employees
SET salary = 7000
WHERE last_name = 'Grant';

User B

SELECT *
FROM userA.employees;

Data blocks

Undo segments

Changed and unchanged data

Before change ("old" data)

Read-consistent image
Summary

In this lesson, you should have learned how to:

• Describe the purpose of the Transaction Control Language (TCL)

• Explain the TCL operations that are required to manage a transaction:
  – COMMIT
  – SAVEPOINT
  – ROLLBACK

• Describe the need for read consistency