Database Transactions

A database transaction is a group of SQL statements that perform a logical unit of work. You can think of a transaction as an inseparable set of SQL statements whose results should be made permanent in the database as a whole (or undone as a whole).

An example of a database transaction might be a transfer of money from one bank account to another. You would need one UPDATE statement to subtract from the total amount of money from one account, and another UPDATE would add money to the other account. Both the subtraction and the addition must be permanently recorded in the database; otherwise, money will be lost. If there is a problem with the money transfer, then the subtraction and addition must both be undone.

This simple example uses only two UPDATE statements, but a transaction may consist of many INSERT, UPDATE, and DELETE statements.

Committing and Rolling Back a Transaction

To permanently record the results made by SQL statements in a transaction, you perform a commit, using the SQL COMMIT statement. If you need to undo the results, you perform a rollback, using the SQL ROLLBACK statement, which resets all the rows back to what they were originally.

The following example adds a row to the customers table and then makes the change permanent by performing a COMMIT:

```
INSERT INTO customers VALUES (6, 'Fred', 'Green', '01-JAN-1970', '800-555-1215');

1 row created.

COMMIT;

Commit complete.
```

The following example updates customer #1 and then undoes the change by performing a ROLLBACK:

```
UPDATE customers SET first_name = 'Edward' WHERE customer_id = 1;

1 row updated.

ROLLBACK;

Rollback complete.
```

Starting and Ending a Transaction

A transaction is a logical unit of work that enables you to split up your SQL statements. A transaction has a beginning and an end.

A transaction begins when one of the following events occurs:

- You connect to the database and perform a DML statement (an INSERT, UPDATE, or DELETE).
- A previous transaction ends (see below) and you enter another DML statement.

A transaction ends when one of the following events occurs:

- You perform a COMMIT or a ROLLBACK.
- You perform a DDL statement, such as a CREATE TABLE, ALTER, TRUNCATE, etc. statement, in which case a COMMIT is automatically performed.
- You perform a DCL statement, such as a GRANT statement, in which case a COMMIT is automatically performed.
- You disconnect from the database. If you exit SQL*Plus normally, by entering the EXIT command, a COMMIT is automatically performed for you. If SQL*Plus terminates abnormally—for example, if the computer on which SQL*Plus was running were to crash—a ROLLBACK is automatically performed. This applies to any program that accesses a database. For example, if you wrote a Java program that accessed a database and your program crashed, a ROLLBACK would be automatically performed.
- You perform a DML statement that fails, in which case a ROLLBACK is automatically performed for that individual DML statement.

Note: It is poor practice not to explicitly commit or roll back your transactions, so perform a COMMIT or ROLLBACK at the end of your transactions.

Note: You can see any changes you have made during the transaction by querying the modified tables, but other users cannot see the changes. After you commit the transaction, the changes are visible to other users' statements that execute after the commit.

Example:

```
1. CREATE TABLE stud(ID INT,name VARCHAR2(20));
2. INSERT INTO stud(ID, name) VALUES(1,'Bob');
3. INSERT INTO stud(ID,name) VALUES(2,'Sara');
4. COMMIT;
5. INSERT INTO stud(ID,name) VALUES(3,'Roberto');
6. ROLLBACK;
7. INSERT INTO stud(ID,name) VALUES(4,'Li');
8. ALTER TABLE stud ADD CONSTRAINT stud_pk PRIMARY KEY(ID);
9. INSERT INTO stud(ID,name) VALUES(5,'Maria');
10. ROLLBACK;
11. COMMIT;
```

Note:

- The CREATE statement would have applied a COMMIT to any previous transactions.
- The INSERT statement starts a transaction
- The COMMIT statement hardens all transactions to the table(s)
- The ALTER statement applies an implicit COMMIT (DDL statements do this)
- The ROLLBACK statement ends the transaction

Result:

```
SELECT ID, name FROM stud;
```

ID	NAME
1	Bob
2	Sara
4	Li

Rollback to a Specific Point Using SAVEPOINT

```
1. CREATE TABLE stud(ID INT,name VARCHAR2(20));
2. INSERT INTO stud(ID, name) VALUES(1,'Bob');
3. SAVEPOINT after_Bob
4. INSERT INTO stud(ID,name) VALUES(2,'Sara');
5. ROLLBACK TO SAVEPOINT after_Bob
6. COMMIT;
7. INSERT INTO stud(ID,name) VALUES(3,'Roberto');
8. ROLLBACK;
9. INSERT INTO stud(ID,name) VALUES(4,'Li');
10. ALTER TABLE stud ADD CONSTRAINT stud_pk PRIMARY KEY(ID);
11. INSERT INTO stud(ID,name) VALUES(5,'Maria');
12. ROLLBACK;
13. COMMIT;
```

Note: The ROLLBACK TO SAVEPOINT does NOT end the transaction.

Result:

SELECT ID, name FROM stud;

ID	NAME
1	Bob
4	Li