# [Image13m](http://images.google.ca/imgres?imgurl=http://www.bmcclure.net/mypictures/Image13m.jpg&imgrefurl=http://www.bmcclure.net/Links2/delphi.htm&h=337&w=556&sz=14&tbnid=ht7lFNDgXjcJ:&tbnh=79&tbnw=130&start=7&prev=/images?q%3Dgreek%2Bmythology%2Boracle%26hl%3Den%26lr%3D%26sa%3DG)Lab Exercises

# DATABASE TRANSACTION

Materials:

Database Systems textbook: Chapter 7, pp. 213-214. Oracle 11g textbook: Chapter 5; Using Transaction control statements pp. 157-162.

**Introduction to database transactions**

A transaction is a logical, atomic unit of work that contains one or more DML (Update/Insert/Delete) or one DDL SQL statements.

A transaction assures that all SQL statements are either **all committed**, which means they are applied to the database, or **all rolled back**, which means they are undone from the database. In the Oracle DBMS every transaction gets a unique identifier called a **transaction ID.**

All Oracle transactions have the basic **ACID** properties. ACID is an acronym for the following:

* **Atomicity**

All tasks of a transaction are performed or none of them are performed (no partial transactions). If a transaction has 1000 inserts, all of them have to be performed or none.

* **Consistency**

The transaction takes the database from one consistent state to another consistent state. For example, in banking, a transfer between two accounts has to keep the consistent amounts.

* **Isolation**

The effect of a transaction is not visible to other transactions **until the transaction is committed**. For example, one user updating the salary in a table does not see the uncommitted changes to the table made concurrently by another user. Thus, it appears to users as if transactions are executing serially.

* **Durability**

Changes made by **committed transactions are permanent**. After a transaction completes, the database ensures through its recovery mechanisms that changes from the transaction are not lost.

The use of transactions is one of the most important ways that a **DBMS differs from a file system**.

**TRANSACTIONS AND AUTOCOMMIT**

SQLDeveloper (the client) uses preferences to set the default for the AUTOCOMMIT parameter. When it is set to ON, each DML statement will be automatically committed. You can view/change the setting by going to Tools | Preferences | Database | Advanced and see if the checkbox for Autocommit is checked. If is checked it means that the AUTOCOMMIT is on.

**Check** the setup of the AUTOCOMMIT for the SQLDeveloper. It should be off (checkbox empty)

Also, you can check the AUTCOMMIT status using SQL\*Plus command:

SHOW AUTOCOMMIT;

Each transaction has a transaction ID and status (there are more attributes but this material is covered by a DBA course):

You can display transaction ID and status using a view of active transactions: V$TRANSACTION

--To see the columns:

DESCRIBE V$TRANSACTION;

**DO** the following exercise for AUTOCOMMIT/ROLLBACK/COMMIT

CREATE TABLE test

(col1 NUMBER (5), col2 VARCHAR2(10) );

**ROLLBACK;**

SELECT Table\_Name FROM User\_Tables; -- do you see the table

**Table test is listed among the User\_tables. This is because each DDL statement is automatically committed to the database and is created physically in the database**

SELECT \* FROM Test; -- what is in the table:

**Table is empty and no rows are returned**

INSERT INTO Test VALUES (1, 'ABC' );

INSERT INTO Test VALUES (2, 'HJKKKKK' );

SELECT \* FROM Test; -- what do you see:

**The rows inserted are returned from the buffer:**

SELECT XID AS "transaction id", STATUS AS "txn status"

FROM V$TRANSACTION;

DESC V$TRANSACTION;

**ROLLBACK;**

SELECT \* FROM Test; -- what do you see:

**No rows are returned. The DML statements are rolled back by the rollback statement and changes made in the buffer are discarded**

INSERT INTO Test VALUES (1, 'ABC' );

INSERT INTO Test VALUES (2, 'HJKKKKK' );

SELECT \* FROM Test; -- what do you see:

**The rows inserted are returned from the buffer:**

**COMMIT;**

SELECT \* FROM Test; -- what do you see:

**The rows inserted are returned from the buffer:**

**ROLLBACK;**

SELECT \* FROM Test; -- what do you see

**The rows inserted are returned from the buffer. The commit statement made the physical changes in the database and the tables are now permanently changed**