

# **Databases**

**(Bases de Datos, BDA)**

**Escuela Técnica Superior de Informática Aplicada**  
**Facultad de Informática**

**Lab exercise book n°4: Study of the Oracle DBMS**  
**Transaction management.**

1 Concept of transaction.....	2
2 Managing transactions in SQL.....	2
3 Managing transactions in ORACLE.....	2
4 Exercises.....	3

*Departamento de Sistemas Informáticos and Computación*  
*2007/2008*

## 1 Concept of transaction

A transaction is a sequence of manipulation instructions on the database which conforms a logical execution unit. A transaction must satisfy the properties of atomicity, consistency, isolation and persistence (ACID). The maintenance of each of these properties the responsibility of a module of the DBMS: atomicity and persistence is the responsibility of the recovery module, isolation is the responsibility of the concurrence model and consistency is the responsibility of the integrity checking module and the recovery module.

## 2 Managing transactions in SQL

SQL offers the following instructions for handling transactions:

- Starting a transaction: there is no specific instruction to start a transaction.
- Ending a transaction (with confirmation): COMMIT [WORK] (confirmed transaction).
- Ending a transaction (with cancellation): ROLLBACK [WORK] (cancelled transaction).

It is important to remind that the end of a transaction does not imply its definitive confirmation, since the final confirmation is conditioned by the subsequent checking of the integrity constraint. The DBMS has the capability of confirming or cancelling a transaction definitively which has just confirmed by a user. Moreover, the DBMS can interrupt and cancel a transaction which has not been finalised by the user. The DBMS must ensure that the changes which have been performed by a confirmed transaction (in a definitive way) must be permanently recorded in the database and the changes which have been performed by a cancelled or interrupted transaction must be undone.

One of the important properties of transactions is consistency, i.e., a transaction must lead the database to a consistent state, in which all the integrity constraints are met. Consequently, it is important to know the strategy adopted by the system to make this checking.

The existence of the concept of transaction and its nature of logical execution unit sets out two possible strategies for the integrity checking. A constraint can be checked after the execution of each SQL instruction which is relevant for the constraint (this is the IMMEDIATE mode) or after the end of each transaction which includes an SQL instruction which is relevant for the constraint (this is the DEFERRED mode). The mode of a constraint can be changed locally inside a transaction if the constraint has been defined as DEFERRABLE); a constraint is DEFERRABLE if the property check can be DEFERRED to the end of the transaction. The mode of constraint check is defined in the database schema with the clause [*when\_to\_check*]:

```
when_to_check ::= [[NOT] DEFERRABLE] [INITIALLY {IMMEDIATE | DEFERRED}]
```

And the instruction in SQL which allows us to change the mode of a defined constraint as deferred inside a transaction, is:

```
SET CONSTRAINT { constraint1, constraint2, ... | ALL} {IMMEDIATE | DEFERRED}
```

## 3 Managing transactions in ORACLE

When working interactively with ORACLE a transaction is started with the first SQL<sup>1</sup> instruction which is executed by the user after starting the system (start session) or after the last transaction. A transaction ends when the user finalises it explicitly with the COMMIT [WORK]

---

<sup>1</sup> We assume that only the DML instructions are used.

sentence (partially confirmed transaction) or it is cancelled explicitly with the ROLLBACK [WORK] sentence (cancelled transaction) or either when the system finalises the transaction implicitly because of the end of the session (partially confirmed transaction) or the system implicitly cancels the transaction because of an error (cancelled transaction). In other words, in ORACLE, the operation which starts a transaction is implicit and the operation which ends a transaction can be explicit or implicit.

Regarding the constraint checking strategy in ORACLE, it is the same as the standard SQL with the following behaviour wrt. the violation of a constraint: "if, during the execution of a transaction a constraint which has been defined with immediate mode is violated, the system only undoes the effect of the SQL operation which has caused the violation of the constraint (*statement rollback*) and the transaction can continue; if, when the transaction is finished, a constraint which has been defined with deferred mode is violated, the system cancels the transaction and undoes its global effect (*transaction rollback*)".

#### 4 Exercises

Consider the database which you designed in the previous lab session and, checking it with the solutions provided by the teacher on the web, perform the following exercises.

- a) In ORACLE, there is no "update in cascade" directive for the restoration of the referential integrity. How can you modify the code of a user who has had loans, without violating the referential integrity of the loan table? (need of the concept of transaction).
- b) Design a transaction which performs tuple insertions over the table *Users* according the following transaction schema:

```

COMMIT;
INSERT INTO user VALUES ('s1', ..., ..., ..., 0);
INSERT INTO user VALUES ('s2', ..., ..., ..., 0);
INSERT INTO user VALUES ('s1', ..., ..., ..., 0);
INSERT INTO user VALUES ('s3', ..., ..., ..., 0);
COMMIT;
    
```

where s1, s2 and s3 are user codes which haven't been used previously and the dots (...) represent any possible values for *name*, *address* and *telephone*.

This transaction violates the constraint of primary key for the table *User*.

Execute two instances of the previous transaction schema, one with a primary key constraint for the table *user* in IMMEDIATE mode and the other with DEFERRED MODE. Which differences do you see? (atomicity and consistency).

- c) Start in your PC two different sessions over the databases. In each session perform the following transactions ( $t_i$  indicates the order in which the operations must be performed):

Session 1	Session 2
$t_0$ "query the total number of users"	
	$t_1$ "query the number total of users"
$t_2$ "insert a new user"	

t<sub>3</sub> “query the total number of users”

t<sub>4</sub> “query the total number of users”

t<sub>5</sub> “end the transaction”

t<sub>6</sub> “query the total number of users”

t<sub>7</sub> “insert a new user”

t<sub>8</sub> “query the total number of users”

t<sub>9</sub> “cancel the transaction”

t<sub>10</sub> “query the total number of users”

t<sub>11</sub> “query the total number of users”

Once both transactions have been finished, query the total number of users in both sessions. How do you interpret the results of the query instructions t<sub>3</sub>, t<sub>4</sub>, t<sub>6</sub> and t<sub>8</sub>? and the results of t<sub>10</sub> and t<sub>11</sub>? (isolation and persistence).

According to the previous exercises, can you affirm that ORACLE maintains the persistence property?