Transacciones

BASES DE DATOS

Profesor: Héctor Gómez Gauchía

Materiales: Héctor Gómez Gauchía, Mercedes García Merayo

Transacciones

- Una transacción es un conjunto de una o varias instrucciones SQL cuya ejecución constituye una unidad lógica e indivisible.

- Específicamente, una transacción es:
  - Una o más instrucciones DML que producen un cambio en el estado de la base de datos.
  - Una instrucción DDL.

- Las sentencias de una transacción se realizan todas o ninguna. No existen operaciones parciales.

Control de Transacciones

- El control de transacciones permite gestionar los cambios realizados por instrucciones DML y su agrupación en transacciones.

- Permite a los diseñadores de las aplicaciones la creación de unidades lógicas de instrucciones para mantener la consistencia de los datos.

- Para ello existe un conjunto de instrucciones SQL que permiten implementar el control de las transacciones.

Transacciones

- Las transacciones pueden ser **confirmadas** o **canceladas.**
- Cuando una transacción se confirma, *los cambios provocados por ella se hacen permanentes*.
- Todas las sentencias de una transacción pueden ejecutarse correctamente, pero sus efectos podrán ser cancelados mientras la transacción no sea confirmada.
- La cancelación de una transacción significa que *todos los cambios hechos por las instrucciones de una transacción son cancelados*.

Control de Transacciones

- La sentencia **COMMIT** termina una transacción y hace permanentes y visibles los cambios al resto de los usuarios.

- La sentencia **ROLLBACK** descarta todos los cambios realizados durante la transacción actual, esto es, desde el último **COMMIT o ROLLBACK**.

- La instrucción **SAVEPOINT <nombre>** establece un punto en una transacción hasta el cual se podrá cancelar la transacción mediante una sentencia **ROLLBACK TO SAVEPOINT <nombre>.**

Transacciones

- Una transacción comienza con la primera instrucción SQL ejecutable (instrucciones DML y DDL o la instrucción **SET TRANSACTION**).

- Una transacción termina en las siguientes situaciones:

  - La ejecución explícita de las sentencias **COMMIT** o **ROLLBACK** sin la claúsula **SAVEPOINT**.

    Mediante la instrucción **COMMIT** el usuario solicita que los cambios realizados por una transacción sean permanentes y visibles al resto de los usuarios.

Transacciones

- Una transacción termina en las siguientes situaciones
  - Se ejecuta una instrucción DDL. El gestor realiza de forma implícita un **`COMMIT`** antes y depués de cada sentencia DDL.
  - La ejecución implícita de un **`COMMIT`** cuando el usuario termina la sesión. Este comportamiento es configurable.
  - La ejecución implícita de un **`ROLLBACK`** debida a un error durante la ejecución de un proceso.
- Tras la finalización de una transacción, la siguiente instrucción SQL ejecutable inicia automáticamente la siguiente transacción.

Ejemplo

LA TRANSACCION COMIENZA AQUI

- UPDATE cuentas
  SET saldo= saldo- 500
  WHERE cuenta= 3209;
- SELECT count(*) INTO existe from cuentas
  WHERE cuenta=3208;
- IF existe >0 then

  UPDATE cuentas
  SET saldo = saldo + 500
  WHERE cuenta=3208;

  INSERT INTO movimientos
  VALUES (mov_seq.NEXTVAL,
  'TR' 3209, 3208, 500);

  COMMIT;

ELSE ROLLBACK;

SENTENCIAS DE LA TRANSACCIÓN

LA TRANSACCION TERMINA

UNA NUEVA TRANSACCION COMIENZA AQUI

Puntos de control (Savepoints)

- Es posible declarar puntos intermedios en una transacción que permiten guardar el trabajo realizado hasta el mismo.
- Un punto de control se establece mediante la instrucción **`SAVEPOINT <nombre>.`**
- Para cancelar una transacción hasta un punto de control se utiliza la sentencia **`ROLLBACK TO SAVEPOINT <nombre>:`**
  - Solo deshace las sentencias realizadas entre el punto de control y el **`ROLLBACK`**
  - Preserva el **`SAVEPOINT`** correspondiente y los **`SAVEPOINTS`** anteriores a este.
  - Libera los bloqueos obtenidos tras el **`SAVEPOINT`** y preserva los obtenidos antes del mismo.
- Una transacción que se retrocede a un punto de control permanece activa.

Control de Transacciones

- La sentencia `SET TRANSACTION` inicia una transacción explicitamente

```
SET TRANSACTION [read only |read write] NAME <nombre>
```

- La opción `read only` hace que todas las consultas que se incluyen en la transacción accedan a los datos en el estado en el que se encontraban antes de que se iniciara la misma. Las consultas no ven los cambios realizados por otros usuarios. No permite instrucciones DML.

- La opción `read write` es la opción por defecto. Los datos son consistentes durnte la ejecución de cada sentencia en la transacción.

Control de Transacciones

```
CREATE TABLE EMPLEADO
 (NIF VARCHAR2(9) NOT NULL,
 NOMBRE VARCHAR2(20),
 SALARIO NUMBER(6,2),
 APELLIDOS VARCHAR2(40),
 CONSTRAINT EMP_PK PRIMARY KEY (NIF));

INSERT INTO empleado
VALUES('10000000A','Jorge',3000.11,'Perez Sala');

ROLLBACK;
```

Control de Transacciones

```
INSERT INTO empleado
VALUES('30000000C','Javier',2000.22,'Sala
Rodriguez');

INSERT INTO empleado
VALUES('30000000C','Soledad',2000.33,'Lopez J.');

INSERT INTO empleado
VALUES('40000000D','Sonia',1800.44,'Moldes R.');

INSERT INTO empleado
VALUES('50000000E','Antonio',1800.44,'Lopez A.');

COMMIT;
```

Control de Transacciones

- table EMPLEADO creado.
- 1 filas insertadas.
- rollback terminado.
- 1 filas insertadas.
- Error que empieza en la línea 18 del comando:
  INSERT INTO empleado
  VALUES('30000000C','Soledad',2000.33,'Lopez J.')
  Informe de error:
  Error SQL: ORA-00001: unique constraint (MGM.EMP_PK) violated
- 1 filas insertadas.
- 1 filas insertadas.
- confirmado.

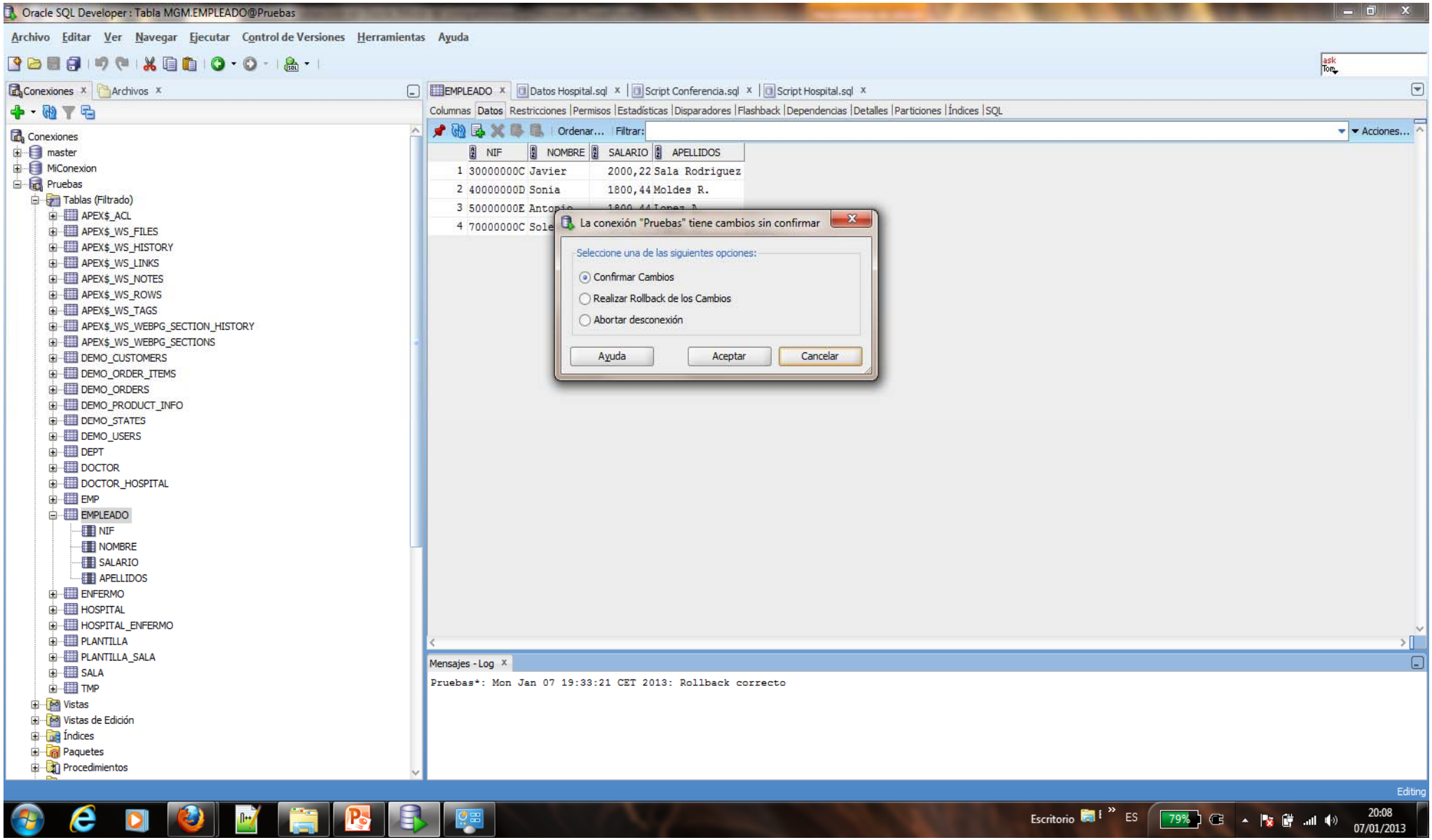
Control de Transacciones

30000000C	Javier	2000,22	Sala Rodriguez
40000000D	Sonia	1800,44	Moldes R.
50000000E	Antonio	1800,44	Lopez A.

```
INSERT INTO empleado
VALUES('70000000C','Soledad',2000.33,'Lopez J.');
```

- Este registro solo será permanente en la base de datos y visible al resto de los usuarios si se confirma la transacción.

Control de Transacciones



Control de Transacciones

```
SET TRANSACTION NAME 'sal_update';
UPDATE empleado SET salario = 7000
WHERE NIF= '30000000C';

SAVEPOINT after_salario;
UPDATE empleado SET salario = 12000
WHERE NIF= '40000000D';

ROLLBACK TO SAVEPOINT after_salario;

UPDATE empleado SET salario = 11000
WHERE NIF= '40000000D';
COMMIT;
```

Control de Transacciones

30000000C	Javier	7000	Sala Rodriguez
40000000D	Sonia	11000	Moldes R.
50000000E	Antonio	1800,44	Lopez A.