

SQL



Lenguaje de Manipulación de Datos

DML

- una sentencia DML se ejecuta cuando en la base de datos se necesita:
 - Anexar,
 - Actualizar, o
 - Borrar datos.
- Una *transacción* es una unidad lógica de trabajo formada por sentencias DML.

Insertando Renglones

Renglón nuevo

50	DEVELOPMENT	DETROIT
----	-------------	---------

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

“...inserta un renglón en una tabla...”



DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON
50	DEVELOPMENT	DETROIT

Sentencia INSERT

INSERT inserta renglones

```
INSERT INTO  table [(column [, column...])]  
VALUES      (value [, value...]);
```

```
SQL> INSERT INTO      dept (deptno, dname, loc)  
      2  VALUES      (50, 'DEVELOPMENT', 'DETROIT');  
1 row created.
```

Sentencia INSERT

Insertar renglones con valores nulos

Omita la columna

```
SQL> INSERT INTO dept (deptno, dname)
      2 VALUES (60, 'MIS');
1 row created.
```

Escriba la palabra clave NULL.

```
SQL> INSERT INTO dept
      2 VALUES (70, 'FINANCE', NULL);
1 row created.
```

Insertando valores especiales

SYSDATE es la fecha y hora actual

```
SQL> INSERT INTO      emp (empno, ename, job,  
  2                   mgr, hiredate, sal, comm,  
  3                   deptno)  
  4 VALUES           (7196, 'GREEN', 'SALESMAN',  
  5                   7782, SYSDATE, 2000, NULL,  
  6                   10);  
  
1 row created.
```

Sentencia INSERT

Insertando valores de fecha

– Alta de un empleado nuevo

```
SQL> INSERT INTO emp
  2  VALUES      (2296, 'AROMANO', 'SALESMAN', 7782,
  3                TO_DATE('FEB 3, 97', 'MON DD, YY'),
  4                1300, NULL, 10);
1 row created.
```

Empleado nuevo:

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
-----	-----	-----	-----	-----	-----	-----	-----
2296	AROMANO	SALESMAN	7782	03-FEB-97	1300		10

Sentencia INSERT

Variables de entrada

```
SQL> INSERT INTO      dept (deptno, dname, loc)
  2  VALUES          (&department_id,
  3                   '&department_name', '&location');
```

```
Enter value for department_id: 80
Enter value for department_name: EDUCATION
Enter value for location: ATLANTA

1 row created.
```


Inserte renglones

- ACCEPT: almacena el valor en una variable.
- PROMPT: etiqueta para solicitar el valor de la variable

```
ACCEPT      department_id PROMPT 'Please enter the -  
department number:'  
ACCEPT      department_name PROMPT 'Please enter -  
the department name:'  
ACCEPT      location PROMPT 'Please enter the -  
location:'  
INSERT INTO dept (deptno, dname, loc)  
VALUES      (&department_id, '&department_name',  
            '&location');
```

Copiando renglones de otra tabla

- Una subconsulta nos permite obtener los datos a insertar.

```
SQL> INSERT INTO managers(id, name, salary, hiredate)
2          SELECT empno, ename, sal, hiredate
3          FROM    emp
4          WHERE   job = 'MANAGER';
3 rows created.
```

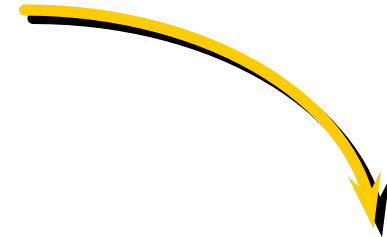
- La clausula VALUES no se utiliza
- Las columna de la clausula INSERT deben coincidir con las columnas de la clausula SELECT

Actualizando Datos

EMP

EMPNO	ENAME	JOB	...	DEPTNO
7839	KING	PRESIDENT		10
7698	BLAKE	MANAGER		30
7782	CLARK	MANAGER		10
7566	JONES	MANAGER		20
...				

“...cambiar el departamento de un empleado”



EMP

EMPNO	ENAME	JOB	...	DEPTNO
7839	KING	PRESIDENT		10
7698	BLAKE	MANAGER		30
7782	CLARK	MANAGER		20
7566	JONES	MANAGER		20
...				

Sentencia UPDATE

- Actualiza uno o más registros

```
UPDATE      table  
SET         column = value [, column = value]  
[WHERE      condition];
```

Sentencia UPDATE

- En la clausula WHERE se indican los renglones a ser modificados.

```
SQL> UPDATE emp
2 SET deptno = 20
3 WHERE empno = 7782;
1 row updated.
```

- Si se omite, se actualizan todos los renglones.

```
SQL> UPDATE employee
2 SET deptno = 20;
14 rows updated.
```

Actualizando Datos con Subconsultas

```
SQL> UPDATE emp
2 SET (job, deptno) =
3      (SELECT job, deptno
4      FROM emp
5      WHERE empno = 7499)
6 WHERE empno = 7698;
1 row updated.
```

Actualizando Datos de otras Tablas

```
SQL> UPDATE    employee
      2  SET      deptno = (SELECT    deptno
      3                                FROM      emp
      4                                WHERE     empno = 7788)
      5  WHERE    job      = (SELECT    job
      6                                FROM      emp
      7                                WHERE     empno = 7788);
```

2 rows updated.

Error de integridad

```
SQL> UPDATE    emp
      2  SET      deptno = 55
      3  WHERE    deptno = 10;
```

El departamento no existe

```
UPDATE emp
      *
ERROR at line 1:
ORA-02291: integrity constraint (USR.EMP_DEPTNO_FK)
violated - parent key not found
```



Borrando Renglones

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON
50	DEVELOPMENT	DETROIT
60	MIS	
...		

**“...borrar el
departamento 50...”**

DEPT



DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON
60	MIS	
...		

Sentencia DELETE

```
DELETE [FROM]    table  
[WHERE           condition];
```

- Indique en la clausula WHERE el renglón a borrar

```
SQL> DELETE FROM    department  
      2 WHERE        dname = 'DEVELOPMENT';  
1 row deleted.
```

- Si se omite se borran todos los renglones

```
SQL> DELETE FROM    department;  
4 rows deleted.
```

Borrando Renglones basándose en otra tabla

```
SQL> DELETE FROM      employee
2  WHERE               deptno =
3                      (SELECT  deptno
4                          FROM    dept
5                          WHERE    dname = 'SALES' );
```

6 rows deleted.

Error de integridad

```
SQL> DELETE FROM dept
      2 WHERE deptno = 10;
```



```
DELETE FROM dept
      *
ERROR at line 1:
ORA-02292: integrity constraint (USR.EMP_DEPTNO_FK)
violated - child record found
```

No es posible borrar renglones cuya llave primaria este referenciada por una llave secundaria en otra tabla

Transacciones

Consta de

- Una o más sentencias DML

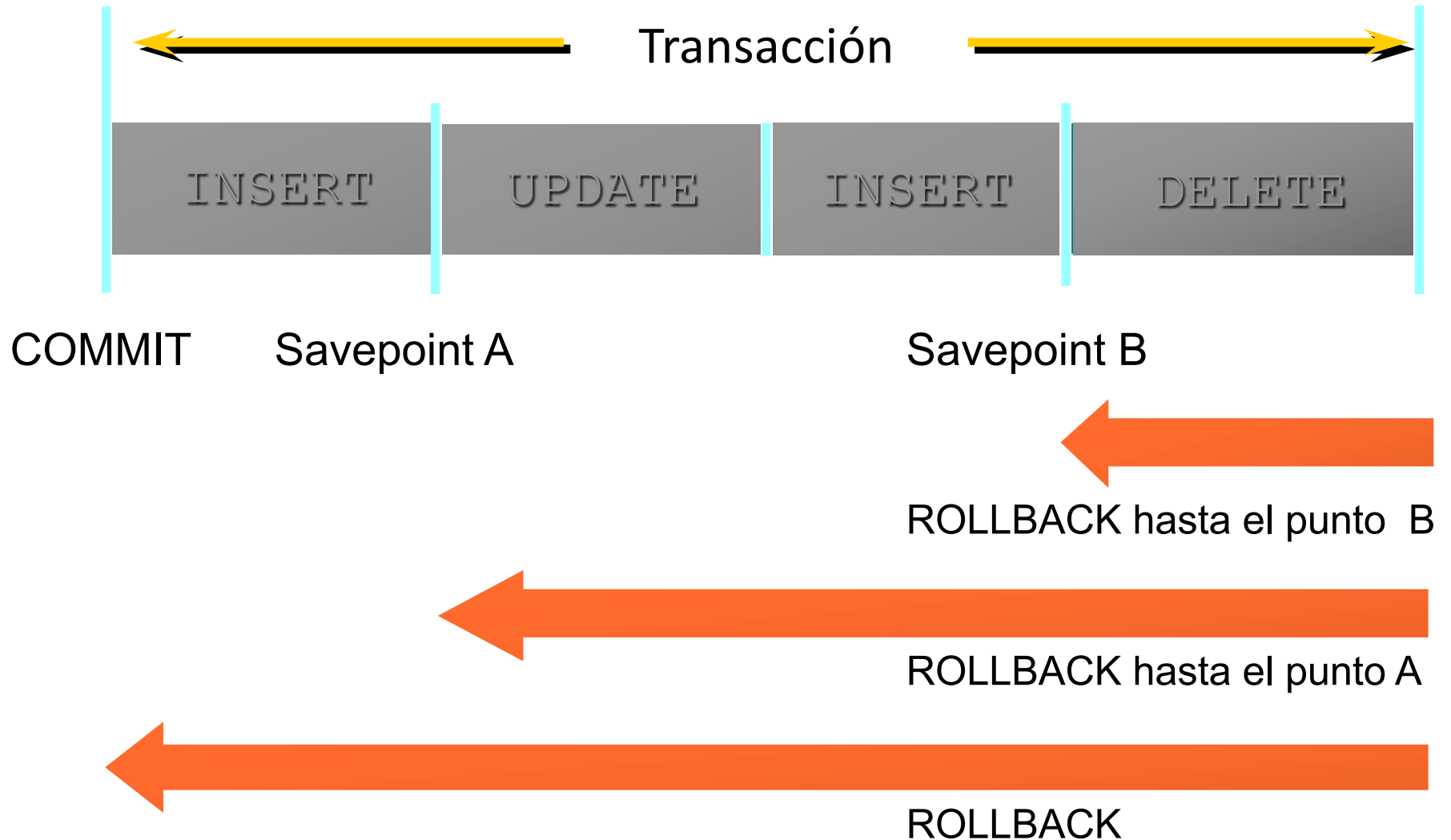
Puede contener

- Una sentencia DDL
- Una sentencia DCL

Transacciones

- Inicia cuando la primer sentencia del SQL es ejecutada
- Finaliza cuando:
 - Encuentra un COMMIT ó ROLLBACK
 - Ejecuta una sentencia DDL or DCL (commit automático)
 - El usuario abandona
 - Hay un estado de caída de sistema

Control de Transacciones



Transacciones automáticas

Una confirmación automática se produce cuando:

1. Se emite una declaración DDL o DCL
2. Es una salida normal del SQL * Plus (COMMIT o ROLLBACK implícito).

Un retroceso automático se genera cuando termina SQL * Plus de manera anormal o hay un fallo del sistema

Datos antes del COMMIT ó ROLLBACK

Sólo el usuario que lo realiza puede revisar los resultados de las operaciones DML mediante la instrucción SELECT.

Los renglones afectados están bloqueados y no pueden ser modificados.

Se pueden deshacer cambios.

Datos después del COMMIT

- Los cambios en los datos son permanentes
- Se pierden los valores anteriores de los datos
- Los usuarios ya pueden ver los resultados
- Los renglones afectados se liberan
- Los puntos de retorno se borran.

Manipulación de datos

```
SQL> UPDATE    emp
      2  SET      deptno = 10
      3  WHERE    empno = 7782;
```

1 row updated.

```
SQL> COMMIT;
```

Commit complete.

ROLLBACK

Datos después del ROLLBACK

La instrucción ROLLBACK deshace los cambios realizados después de una instrucción COMMIT

El valor de los datos se restaura

Los renglones bloqueados se liberan

```
SQL> DELETE FROM      employee;
```

```
14 rows deleted.
```

```
SQL> ROLLBACK;
```

```
Rollback complete.
```

Puntos de retorno

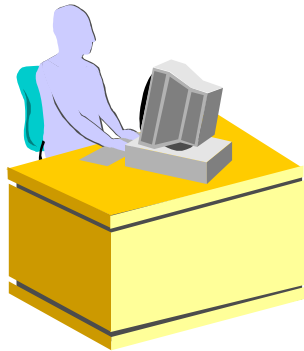
- Crear un punto de retorno con la sentencia SAVEPOINT.
- Retrocedamos a ese punto mediante la sentencia ROLLBACK TO SAVEPOINT

```
SQL> UPDATE...  
SQL> SAVEPOINT update_done;  
Savepoint created.  
SQL> INSERT...  
SQL> ROLLBACK TO update_done;  
Rollback complete.
```

Consistencia de Lectura

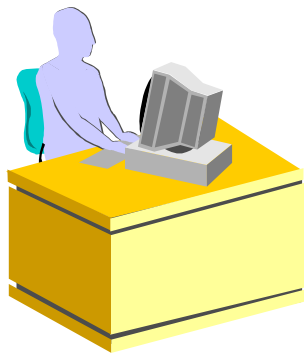
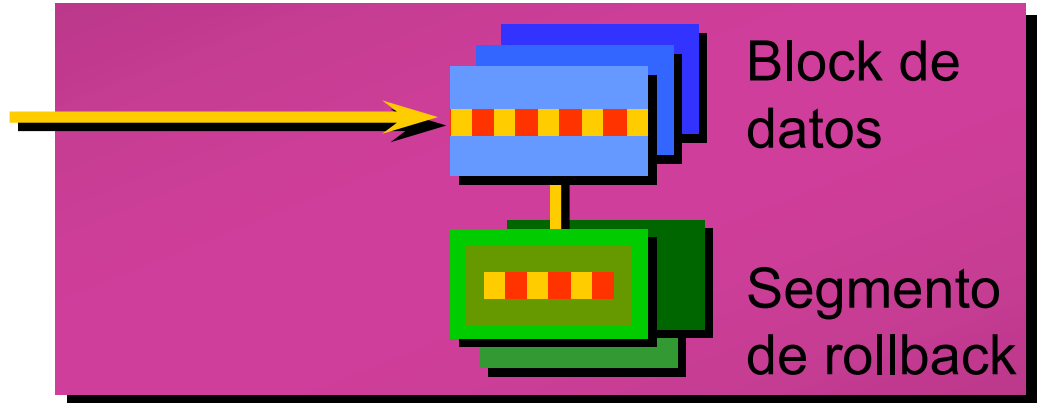
- Garantiza la consistencia de los datos en todo momento y que los cambios realizados por los usuarios no entren en conflicto.
- Asegura que:
 - Los lectores no esperen a los escritores
 - Los escritores no esperen a los lectores

Consistencia de Lectura



Usuario 1

```
UPDATE emp  
SET sal = 2000  
WHERE ename =  
'SCOTT';
```



Usuario 2

```
SELECT *  
FROM emp;
```

