9

Manipulación de Datos

Objetivos

Al finalizar esta lección, debería estar capacitado para lo siguiente:

- Describir cada sentencia de lenguaje de manipulación de datos (DML)
- Insertar filas en una tabla
- Actualizar filas en una tabla
- Suprimir filas de una tabla
- Controlar transacciones

Agenda

- Adición de nuevas filas a una tabla
  - Sentencia `INSERT`
- Cambio de datos en la tabla
  - Sentencia `UPDATE`
- Eliminación de filas de una tabla:
  - Sentencia `DELETE`
  - Sentencia `TRUNCATE`
- Control de transacciones de la base de datos mediante `COMMIT`, `ROLLBACK` y `SAVEPOINT`
- Consistencia de lectura
- Cláusula `FOR UPDATE` en una sentencia `SELECT`

Lenguaje de Manipulación de Datos

- Las sentencias DML se ejecutan al:
  - Agregar nuevas filas a una tabla
  - Modificar filas existentes en una tabla
  - Eliminar filas existentes de una tabla
- Una *transacción* consta de una recopilación de sentencias DML que forman una unidad lógica de trabajo.

Adición de una Nueva Fila a una Tabla

DEPARTMENTS

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700

70	Public Relations	100	1700

Nuevo fila

Insertar una nueva fila en la tabla DEPARTMENTS.

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	70	Public Relations	100	1700
2	10	Administration	200	1700
3	20	Marketing	201	1800
4	50	Shipping	124	1500
5	60	IT	103	1400
6	80	Sales	149	2500
7	90	Executive	100	1700
8	110	Accounting	205	1700
9	190	Contracting	(null)	1700

Sintaxis de las Sentencias INSERT

- Agregar nuevas filas a una tabla mediante la sentencia INSERT:

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

- Con esta sintaxis, sólo se inserta una fila cada vez.

Inserción de Filas

- Insertar una nueva fila que contenga los valores de cada columna.
- Mostrar valores en el orden por defecto de las columnas de la tabla.
- Opcionalmente, mostrar la lista de columnas en la cláusula `INSERT`.

```
INSERT INTO departments(department_id,
       department_name, manager_id, location_id)
VALUES (70, 'Public Relations', 100, 1700);
```

```
1 rows inserted
```

- Encerrar valores de caracteres y de fecha entre comillas simples.

Inserción de Filas con Valores Nulos

- Método implícito: omitir la columna de la lista de columnas.

```
INSERT INTO   departments (department_id,
                           department_name)
VALUES        (30, 'Purchasing');
1 rows inserted
```

- Método explícita: especificar la palabra clave `NULL` en la cláusula `VALUES`.

```
INSERT INTO   departments
VALUES        (100, 'Finance', NULL, NULL);
1 rows inserted
```

Inserción de Valores Especiales

La función `SYSDATE` registra la fecha y hora actuales.

```
INSERT INTO employees (employee_id,
                       first_name, last_name,
                       email, phone_number,
                       hire_date, job_id, salary,
                       commission_pct, manager_id,
                       department_id)
VALUES                 (113,
                       'Louis', 'Popp',
                       'LPOPP', '515.124.4567',
                       SYSDATE, 'AC_ACCOUNT', 6900,
                       NULL, 205, 110);
```

```
1 rows inserted
```

Inserción de Valores de Fecha y Hora Específicos

- Agregar un nuevo empleado.

```
INSERT INTO employees
VALUES      (114,
             'Den', 'Raphealy',
             'DRAPHEAL', '515.127.4561',
             TO_DATE('FEB 3, 1999', 'MON DD, YYYY'),
             'SA_REP', 11000, 0.2, 100, 60);
```

```
1 rows inserted
```

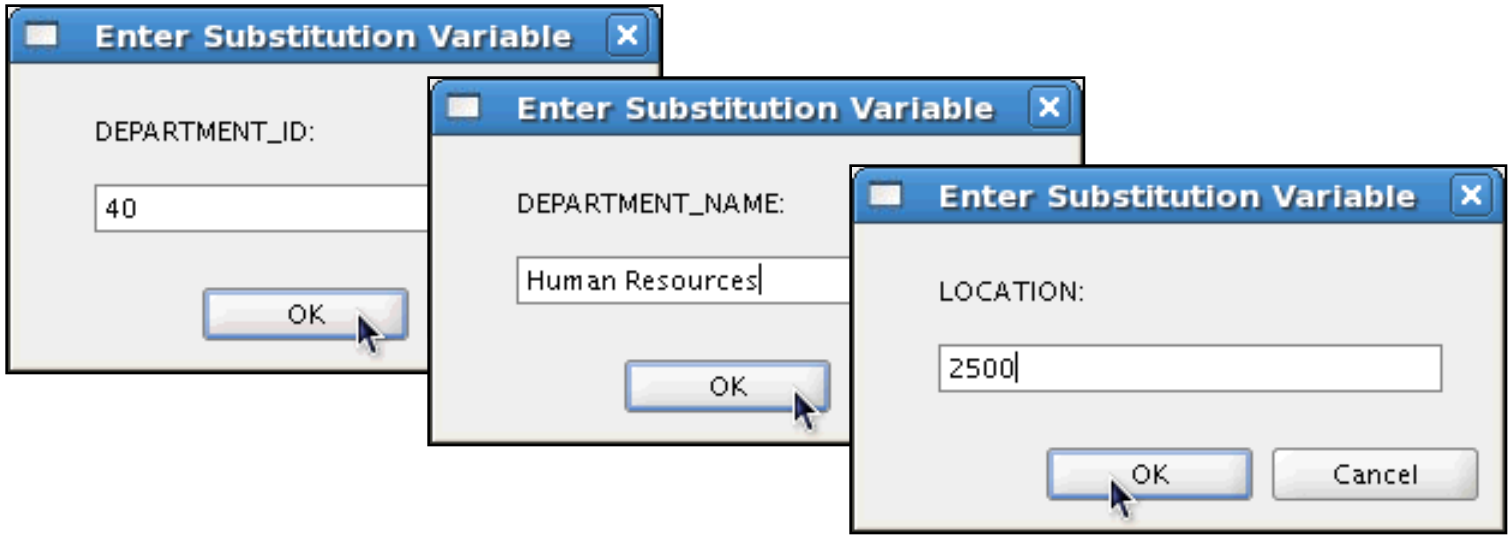
- Verificar la adición.

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT
1	114	Den	Raphealy	DRAPHEAL	515.127.4561	03-FEB-99	SA_REP	11000	0.2

Creación de un Script

- Utilizar el carácter de sustitución & en una sentencia SQL para solicitar valores.
- & es un marcador de posición del valor de la variable.

```
INSERT INTO departments
            (department_id, department_name, location_id)
VALUES      (&department_id, '&department_name',&location);
```



Copia de Filas de Otra Tabla

- Escribir la sentencia `INSERT` con una subconsulta:

```
INSERT INTO sales_reps(id, name, salary, commission_pct)
  SELECT employee_id, last_name, salary, commission_pct
  FROM   employees
  WHERE  job_id LIKE '%REP%';
4 rows inserted
```

- No utilizar la cláusula `VALUES`.
- Hacer coincidir el número de columnas de la cláusula `INSERT` con el de la subconsulta.
- Inserta todas las filas devueltas por la subconsulta en la tabla, `sales_reps`.

Agenda

- Adición de nuevas filas a una tabla
  - Sentencia `INSERT`
- Cambio de datos en la tabla
  - Sentencia `UPDATE`
- Eliminación de filas de una tabla:
  - Sentencia `DELETE`
  - Sentencia `TRUNCATE`
- Control de transacciones de la base de datos mediante `COMMIT`, `ROLLBACK` y `SAVEPOINT`
- Consistencia de lectura
- Cláusula `FOR UPDATE` en una sentencia `SELECT`

Cambio de Datos en la Tabla

EMPLOYEES

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	MANAGER_ID	COMMISSION_PCT	DEPARTMENT_ID
100	Steven	King	24000	(null)	(null)	90
101	Neena	Kochhar	17000	100	(null)	90
102	Lex	De Haan	17000	100	(null)	90
103	Alexander	Hunold	9000	102	(null)	60
104	Bruce	Ernst	6000	103	(null)	60
107	Diana	Lorentz	4200	103	(null)	60
124	Kevin	Mourgos	5800	100	(null)	50

Actualizar filas en la tabla EMPLOYEES:

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	MANAGER_ID	COMMISSION_PCT	DEPARTMENT_ID
100	Steven	King	24000	(null)	(null)	90
101	Neena	Kochhar	17000	100	(null)	90
102	Lex	De Haan	17000	100	(null)	90
103	Alexander	Hunold	9000	102	(null)	80
104	Bruce	Ernst	6000	103	(null)	80
107	Diana	Lorentz	4200	103	(null)	80
124	Kevin	Mourgos	5800	100	(null)	50

Sintaxis de Sentencias UPDATE

- Modificar los valores existentes en una tabla con la sentencia UPDATE:

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

- Actualizar más de una fila cada vez (si es necesario).

Actualización de Filas en una Tabla

- Si se especifica la cláusula `WHERE`, se modifican los valores de una fila o varias filas específicas:

```
UPDATE employees
SET    department_id = 50
WHERE employee_id = 113;
```

```
1 rows updated
```

- Si se omite la cláusula `WHERE`, se modifican los valores de todas las filas de la tabla:

```
UPDATE   copy_emp
SET      department_id = 110;
```

```
22 rows updated
```

- Especificar `SET` *`column_name`*`= NULL` para actualizar un valor de columna a `NULL`.

Actualización de Dos Columnas con una Subconsulta

Actualizar el cargo y el salario del empleado 113 para que coincida con los del empleado 205.

```
UPDATE employees
SET       job_id   = (SELECT  job_id
                     FROM    employees
                     WHERE   employee_id = 205),
          salary   = (SELECT  salary
                     FROM    employees
                     WHERE   employee_id = 205)
WHERE employee_id = 113;
1 rows updated
```

Actualización de Filas Basada en Otra Tabla

Utilizar subconsultas en las sentencias `UPDATE` para actualizar los valores de fila de una tabla según los valores de otra tabla:

```
UPDATE   copy_emp
SET      department_id  =  (SELECT department_id
                           FROM employees
                           WHERE   employee_id = 100)
WHERE    job_id         =  (SELECT job_id
                           FROM employees
                           WHERE  employee_id = 200);

1 rows updated
```

Agenda

- Adición de nuevas filas a una tabla
  - Sentencia `INSERT`
- Cambio de datos en la tabla
  - Sentencia `UPDATE`
- Eliminación de filas de una tabla:
  - Sentencia `DELETE`
  - Sentencia `TRUNCATE`
- Control de transacciones de la base de datos mediante `COMMIT`, `ROLLBACK` y `SAVEPOINT`
- Consistencia de lectura
- Cláusula `FOR UPDATE` en una sentencia `SELECT`

Eliminación de Filas de Tablas

DEPARTMENTS

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700

Suprimir una fila de la tabla DEPARTMENTS:

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700

Sentencia DELETE

Puede eliminar filas existentes de una tabla mediante la sentencia DELETE:

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

Supresión de Filas de Tablas

- Se suprimen filas concretas si se especifica la cláusula WHERE:

```
DELETE FROM departments
WHERE  department_name = 'Finance';
1 rows deleted
```

- Se suprimen todas las filas de la tabla si omite la cláusula WHERE:

```
DELETE FROM  copy_emp;
22 rows deleted
```

Supresión de Filas Basada en Otra Tabla

Utilizar subconsultas en las sentencias `DELETE` para eliminar filas de una tabla según los valores de otra tabla:

```
DELETE FROM employees
WHERE  department_id =
                (SELECT department_id
                 FROM   departments
                 WHERE  department_name
                        LIKE '%Public%');
```

```
1 rows deleted
```

Sentencia TRUNCATE

- Elimina todas las filas de una tabla, dejando la tabla vacía y la estructura de la misma intacta.
- Es una sentencia de lenguaje de definición de datos (DDL) en lugar de una sentencia DML; no se puede deshacer fácilmente.
- Sintaxis:

```
TRUNCATE TABLE table_name;
```

- Ejemplo:

```
TRUNCATE TABLE copy_emp;
```

Agenda

- Adición de nuevas filas a una tabla
  - Sentencia `INSERT`
- Cambio de datos en la tabla
  - Sentencia `UPDATE`
- Eliminación de filas de una tabla:
  - Sentencia `DELETE`
  - Sentencia `TRUNCATE`
- Control de transacciones de base de datos mediante `COMMIT`, `ROLLBACK` y `SAVEPOINT`
- Consistencia de lectura
- Cláusula `FOR UPDATE` en una sentencia `SELECT`

Transacciones de Base de Datos

Una transacción de base de datos consiste en una de las siguientes opciones:

- Sentencias DML que constituyen un cambio consistente de los datos
- Una sentencia DDL
- Una sentencia de lenguaje de control de datos (DCL)

Transacciones de la Base de Datos: Inicio y Fin

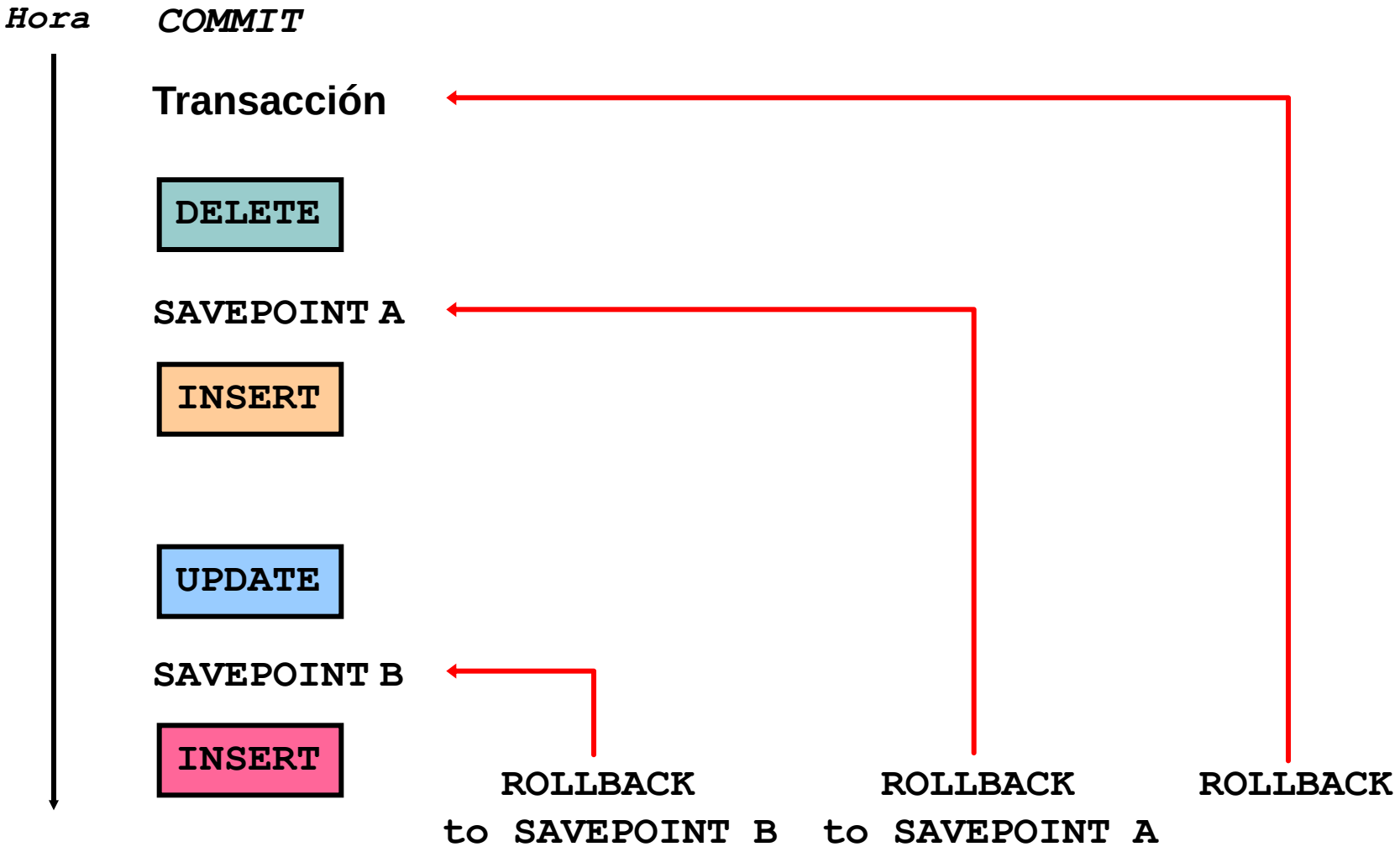
- Empezar cuando se ejecute la primera sentencia SQL de DML.
- Terminar con uno de los siguientes eventos:
  - Se emite una sentencia `COMMIT` o `ROLLBACK`.
  - Se ejecuta una sentencia DDL o DCL (confirmación automática).
  - El usuario sale de SQL Developer o SQL*Plus.
  - El sistema falla.

Ventajas de las Sentencias COMMIT y ROLLBACK

Con las sentencias COMMIT y ROLLBACK, puede:

- Garantizar la consistencia
- Visualizar una presentación preliminar de los cambios de los datos antes de hacerlos permanentes
- Agrupar componentes relacionados de forma lógica

Sentencias de Control de Transacciones Explícitas



Rollback de los Cambios de un Marcador

- Crear un marcador en una transacción actual mediante la sentencia `SAVEPOINT`.
- Realizar rollback en dicho marcador mediante la sentencia `ROLLBACK TO SAVEPOINT`.

```
UPDATE...
SAVEPOINT update_done;
SAVEPOINT update_done succeeded.
INSERT...
ROLLBACK TO update_done;
ROLLBACK TO succeeded.
```

Procesamiento de Transacciones Implícitas

- Una transacción automática se produce en las siguientes circunstancias:
  - Se emite una sentencia DDL.
  - Se emite una sentencia DCL.
  - Salida normal de SQL Developer o SQL*Plus, sin emitir explícitamente las sentencias `COMMIT` o `ROLLBACK`.
- Un rollback automático se produce con la terminación anormal de SQL Developer o SQL*Plus o por un fallo del sistema.

Estado de los Datos antes de `COMMIT` o `ROLLBACK`

- Se puede recuperar el estado anterior de los datos.
- El usuario actual puede revisar los resultados de las operaciones DML mediante la sentencia `SELECT`.
- Los demás usuarios *no pueden* ver los resultados de las sentencias DML del usuario actual.
- Las filas afectadas están *bloqueadas*; otros usuarios no pueden cambiar los datos de las filas afectadas.

Estado de los Datos después de COMMIT

- Los cambios de datos se guardan en la base de datos.
- Se sobrescribe el estado anterior de los datos.
- Todos los usuarios pueden ver los resultados.
- Los bloqueos de las filas afectadas se liberan y dichas filas quedan disponibles para que las manipulen otros usuarios.
- Se borran todos los puntos de grabación.

Confirmación de Datos

- Realice estos cambios:

```
DELETE FROM employees
WHERE employee_id = 99999;

1 rows deleted

INSERT INTO departments
VALUES (290, 'Corporate Tax', NULL, 1700);

1 rows inserted
```

- Confirme los cambios:

```
COMMIT;

COMMIT succeeded.
```

Estado de los Datos después de ROLLBACK

Desechar todos los cambios pendientes mediante la sentencia ROLLBACK:

- Se deshacen los cambios de datos.
- Se restaura el estado anterior de los datos.
- Se liberan los bloqueos de las filas afectadas.

```
DELETE FROM copy_emp;
ROLLBACK ;
```

Estado de los Datos después de ROLLBACK: Ejemplo

```
DELETE FROM test;
25,000 rows deleted.

ROLLBACK ;
Rollback complete.

DELETE FROM test WHERE  id = 100;
1 row deleted.

SELECT * FROM   test WHERE  id = 100;
No rows selected.

COMMIT;
Commit complete.
```

Rollback a Nivel de Sentencias

- Si falla una sentencia DML durante la ejecución, sólo se realiza un rollback de dicha sentencia.
- El servidor de Oracle implanta un punto de grabación implícito.
- Los demás cambios se retienen.
- El usuario debe terminar las transacciones explícitamente con la ejecución de una sentencia `COMMIT` o `ROLLBACK`.

Agenda

- Adición de nuevas filas a una tabla
  - Sentencia `INSERT`
- Cambio de datos en la tabla
  - Sentencia `UPDATE`
- Eliminación de filas de una tabla:
  - Sentencia `DELETE`
  - Sentencia `TRUNCATE`
- Control de transacciones de base de datos mediante `COMMIT`, `ROLLBACK` y `SAVEPOINT`
- Consistencia de lectura
- Cláusula `FOR UPDATE` en una sentencia `SELECT`

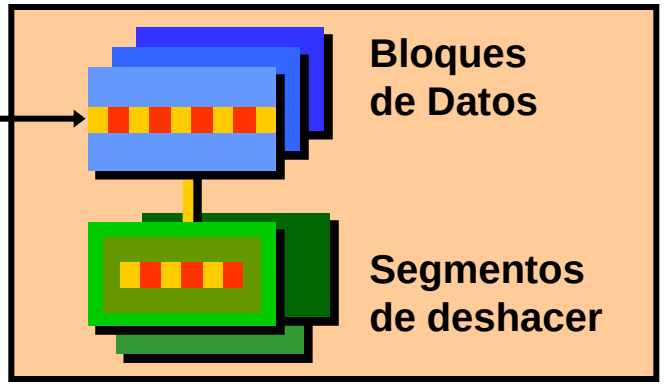
Consistencia de Lectura

- La consistencia de lectura garantiza una vista consistente de los datos en todo momento.
- Los cambios realizados por un usuario no entran en conflicto con los cambios realizados por otro usuario.
- La consistencia de lectura garantiza que en los mismos datos:
  - Los lectores no esperen los escritores.
  - Los escritores no esperen a los lectores.
  - Los escritores esperen a los escritores.

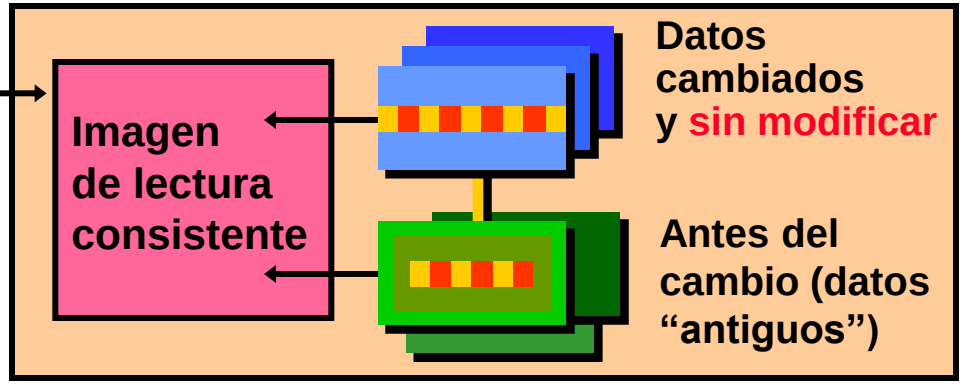
Implementación de Consistencia de Lectura

Usuario A

```
UPDATE employees
SET    salary = 7000
WHERE  last_name = 'Grant';
```



```
SELECT  *
FROM userA.employees;
```



Usuario B

Agenda

- Adición de nuevas filas a una tabla
  - Sentencia `INSERT`
- Cambio de datos en la tabla
  - Sentencia `UPDATE`
- Eliminación de filas de una tabla:
  - Sentencia `DELETE`
  - Sentencia `TRUNCATE`
- Control de transacciones de la base de datos mediante `COMMIT`, `ROLLBACK` y `SAVEPOINT`
- Consistencia de lectura
- Cláusula `FOR UPDATE` en una sentencia `SELECT`

Cláusula FOR UPDATE en una Sentencia SELECT

- Bloquea las filas de la tabla EMPLOYEES en las que job_id sea SA_REP.

```
SELECT employee_id, salary, commission_pct, job_id
FROM employees
WHERE job_id = 'SA_REP'
FOR UPDATE
ORDER BY employee_id;
```

- El bloqueo sólo se libera si emite ROLLBACK o COMMIT.
- Si la sentencia SELECT intenta bloquear una fila bloqueada por otro usuario, la base de datos espera a que la fila esté disponible y, a continuación, devuelve los resultados de la sentencia SELECT.

Cláusula FOR UPDATE: Ejemplos

- Puede utilizar la cláusula `FOR UPDATE` en una sentencia `SELECT` en varias tablas.

```
SELECT e.employee_id, e.salary, e.commission_pct
FROM employees e JOIN departments d
USING (department_id)
WHERE job_id = 'ST_CLERK'
AND location_id = 1500
FOR UPDATE
ORDER BY e.employee_id;
```

- Las filas de las tablas `EMPLOYEES` y `DEPARTMENTS` están bloqueadas.
- Utilizar `FOR UPDATE OF` *`column_name`* para cualificar la columna que desea cambiar, de modo que sólo se bloqueen las filas de esa tabla específica.

Prueba

Las siguientes sentencias producen los mismos resultados:

```
DELETE FROM copy_emp;
```

```
TRUNCATE TABLE copy_emp;
```

1. Verdadero
2. Falso

Resumen

En esta lección debe haber aprendido a utilizar las siguientes sentencias:

Función	Descripción
`INSERT`	Agrega una nueva fila a la tabla.
`UPDATE`	Modifica filas existentes en la tabla.
`DELETE`	Elimina filas existentes de la tabla.
`TRUNCATE`	Elimina todas las filas de una tabla.
`COMMIT`	Convierte todos los cambios pendientes en
`SAVEPOINT`	Se utiliza para realizar un rollback en el marcador de punto de grabación.
`ROLLBACK`	Descarta todos los cambios de datos pendientes.
Cláusula `FOR UPDATE` en `SELECT`	Bloquea las filas identificadas por la consulta `SELECT`.

Práctica 9: Visión General

En esta práctica se abordan los siguientes temas:

- Inserción de filas en tablas
- Actualización y supresión de filas de la tabla
- Control de transacciones