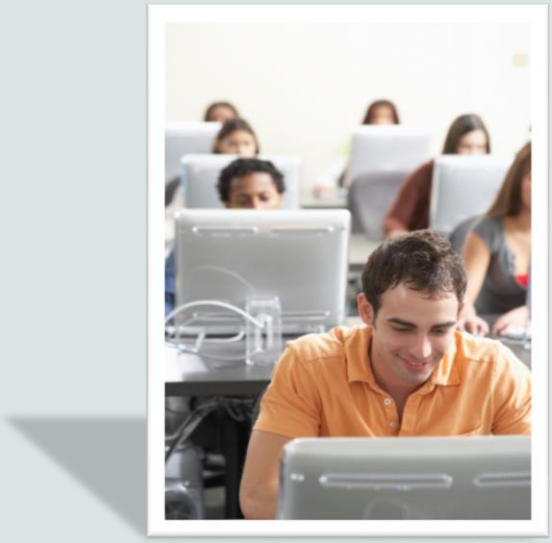

ORACLE
ACADEMY

Programación de bases de datos con SQL

12-1
Sentencias INSERT



ORACLE ACADEMY

Objetivos

En esta lección, aprenderá a:

- Explicar la importancia de poder modificar los datos de una base de datos
- Construir y ejecutar sentencias INSERT que insertan una única fila con una cláusula VALUES
- Construir y ejecutar sentencias INSERT que utilizan valores especiales, valores nulos y valores de fecha
- Construir y ejecutar sentencias INSERT que copian filas de una tabla a otra mediante una subconsulta

Objetivo

- Hasta ahora, ha aprendido cómo acceder a datos de una base de datos.
- Ha llegado el momento de aprender a realizar cambios en los datos de la base de datos.
- En la empresa, las bases de datos son dinámicas.
- Están constantemente en el proceso de inserción, actualización y supresión de datos.

Objetivo

- Piense en cuántas veces cambia la base de datos de alumnos de la escuela de un día a otro y de un año a otro.
- Si no se realizaran cambios, la base de datos perdería rápidamente su utilidad.
- En esta lección, comenzará a utilizar sentencias de lenguaje de manipulación de datos (DML) para realizar cambios en una base de datos.

Copia de Tablas antes de la Inserción

- Será el responsable de la modificación de tablas en el esquema.
- También será responsables de restaurarlos al igual que un administrador de base de datos real asume esta responsabilidad.
- Para mantener las tablas de esquema en su estado original, realizará una copia de cada tabla antes de terminar las actividades prácticas de esta lección y de las posteriores.
- En cada actividad práctica, utilizará la copia de la tabla que cree, pero no la original.
- Si por accidente modificara una copia de la tabla, podrá utilizar la tabla original para restaurar la copia.

Copia de Tablas antes de la Inserción

- Debe asignar un nombre a cada tabla copiada: copy_tablename.
- Las copias de tablas no heredarán las reglas de integridad de clave primaria a clave ajena asociadas (restricciones de relación) de las tablas originales.
- Sin embardo, los tipos de dato de columna se heredan de las tablas copias.

Sintaxis para Crear una Copia de una Tabla

- Cree la sintaxis de tabla:

```
CREATE TABLE copy_tablename
 AS (SELECT * FROM tablename);
```

- Por ejemplo:

```
CREATE TABLE copy_employees
AS (SELECT * FROM employees);
```

```
CREATE TABLE copy_departments
AS (SELECT * FROM departments);
```

Sintaxis para Crear una Copia de una Tabla

- Para verificar y ver la copia de la tabla, utilice las siguientes sentencias DESCRIBE y SELECT:

```
DESCRIBE copy_employees;
```

```
SELECT * FROM copy_employees;
```

```
DESCRIBE copy_departments;
```

```
SELECT * FROM copy_departments;
```


La estructura y el contenido de la tabla también se pueden ver con el Explorador de Objetos en APEX.

INSERT

- La sentencia INSERT se utiliza para agregar una nueva fila a una tabla. La sentencia necesita tres valores:
  - el nombre de la tabla
  - los nombres de las columnas de la tabla que se va a rellenar
  - los valores correspondientes para cada columna
- ¿Cómo podemos INSERTAR los datos siguientes para crear un nuevo departamento en la tabla copy_departments?

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
200	Human Resources	205	1500

INSERT

- La siguiente sintaxis utiliza INSERT para agregar un nuevo departamento a la tabla copy_departments.
- Esta sentencia muestra explícitamente cada columna tal y como aparece en la tabla.
- Los valores para cada columna se muestran en el mismo orden.
  - Tenga en cuenta que los valores de número no están entre comillas simples.

```
INSERT INTO copy_departments
 (department_id, department_name, manager_id, location_id)
VALUES
 (200,'Human Resources', 205, 1500);
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
200	Human Resources	205	1500

Los nombres de columna se pueden mostrar en cualquier orden (aunque no supone ninguna ventaja hacerlo así) mientras que los valores de columna se muestran en el mismo orden que los nombres de columna.

INSERT

- Otra forma de insertar valores en una tabla es agregarlos implícitamente omitiendo los nombres de columna.
- Una precaución: los valores de cada columna deben coincidir exactamente con el orden por defecto en el que aparecen en la tabla (como se muestra en una sentencia DESCRIBE) y se debe proporcionar un valor para cada columna.

Nota: en el momento de la escritura, la función DESCRIBE no funciona como se esperaba en APEX. Mientras sea así, utilice el Explorador de Objetos para ver la información detallada de la tabla.

INSERT

- La sentencia INSERT en este ejemplo se ha escrito sin nombrar explícitamente las columnas.
- Sin embargo, para mayor claridad, es mejor utilizar los nombres de columna en una cláusula INSERT.

```
INSERT INTO copy_departments
VALUES
  (210,'Estate Management', 102, 1700);
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
210	Estate Management	102	1700

Comprobación de la Tabla en Primer Lugar

- Antes de insertar datos en una tabla, debe comprobar varios detalles de la tabla.
- La sentencia de nombre de tabla DESCRIBE devolverá una descripción de la estructura de la tabla y el gráfico de resumen de la tabla.
- RESUMEN DE LA TABLA COPY_DEPARTMENTS:

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
DEPARTMENT_ID	NUMBER	-	4	0	-	✓
DEPARTMENT_NAME	VARCHAR2	30	-	-	-	-
MANAGER_ID	NUMBER	-	6	0	-	✓
LOCATION_ID	NUMBER	-	4	0	-	✓

Nota: como esta tabla se creó como copia de la tabla de departamentos, la restricción de clave primaria no se ha copiado de la original.

Resumen de la Tabla

- Como se muestra en el ejemplo, el resumen de la tabla proporciona información sobre cada columna de la tabla, como:
  - permiso de valores duplicados
  - tipo de dato permitido
  - cantidad de datos permitida
  - permiso de valores NULL

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
EMPLOYEE_ID	NUMBER	-	6	0	1	-
FIRST_NAME	VARCHAR2	20	-	-	-	✓
LAST_NAME	VARCHAR2	25	-	-	-	-
EMAIL	VARCHAR2	25	-	-	-	-
PHONE_NUMBER	VARCHAR2	20	-	-	-	✓
HIRE_DATE	DATE	7	-	-	-	-
JOB_ID	VARCHAR2	10	-	-	-	-
SALARY	NUMBER	-	8	2	-	✓
COMMISSION_PCT	NUMBER	-	2	2	-	✓
MANAGER_ID	NUMBER	-	6	0	-	✓
DEPARTMENT_ID	NUMBER	-	4	0	-	✓
BONUS	VARCHAR2	5	-	-	-	✓

Resumen de la Tabla

- Observe que la columna Tipo de Dato para los tipos de dato de caracteres especifica entre paréntesis el número máximo de caracteres permitidos.

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
EMPLOYEE_ID	NUMBER	-	6	0	1	-
FIRST_NAME	VARCHAR2	20	-	-	-	✓
LAST_NAME	VARCHAR2	25	-	-	-	-
EMAIL	VARCHAR2	25	-	-	-	-
PHONE_NUMBER	VARCHAR2	20	-	-	-	✓
HIRE_DATE	DATE	7	-	-	-	-
JOB_ID	VARCHAR2	10	-	-	-	-
SALARY	NUMBER	-	8	2	-	✓
COMMISSION_PCT	NUMBER	-	2	2	-	✓
MANAGER_ID	NUMBER	-	6	0	-	✓
DEPARTMENT_ID	NUMBER	-	4	0	-	✓
BONUS	VARCHAR2	5	-	-	-	✓

Resumen de la Tabla

- First_name tiene un tipo de dato VARCHAR2(20), esto significa que se pueden introducir un máximo de 20 caracteres para esta columna.

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
EMPLOYEE_ID	NUMBER	-	6	0	1	-
FIRST_NAME	VARCHAR2	20	-	-	-	✓
LAST_NAME	VARCHAR2	25	-	-	-	-
EMAIL	VARCHAR2	25	-	-	-	-
PHONE_NUMBER	VARCHAR2	20	-	-	-	✓
HIRE_DATE	DATE	7	-	-	-	-
JOB_ID	VARCHAR2	10	-	-	-	-
SALARY	NUMBER	-	8	2	-	✓
COMMISSION_PCT	NUMBER	-	2	2	-	✓
MANAGER_ID	NUMBER	-	6	0	-	✓
DEPARTMENT_ID	NUMBER	-	4	0	-	✓
BONUS	VARCHAR2	5	-	-	-	✓

Resumen de la Tabla

- Para los tipos de dato Number, los paréntesis especifican la precisión y la escala.
- La precisión es el número total de dígitos y la escala es el número de dígitos a la derecha de la posición decimal.

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
EMPLOYEE_ID	NUMBER	-	6	0	1	-
FIRST_NAME	VARCHAR2	20	-	-	-	✓
LAST_NAME	VARCHAR2	25	-	-	-	-
EMAIL	VARCHAR2	25	-	-	-	-
PHONE_NUMBER	VARCHAR2	20	-	-	-	✓
HIRE_DATE	DATE	7	-	-	-	-
JOB_ID	VARCHAR2	10	-	-	-	-
SALARY	NUMBER	-	8	2	-	✓
COMMISSION_PCT	NUMBER	-	2	2	-	✓
MANAGER_ID	NUMBER	-	6	0	-	✓
DEPARTMENT_ID	NUMBER	-	4	0	-	✓
BONUS	VARCHAR2	5	-	-	-	✓

Los tipos de dato se tratan más detalladamente más adelante en el curso.

Resumen de la Tabla

- La columna SALARY permite números con una precisión de 8 y una escala de 2.
- El valor máximo permitido en esta columna es 999999,99.

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
EMPLOYEE_ID	NUMBER	-	6	0	1	-
FIRST_NAME	VARCHAR2	20	-	-	-	✓
LAST_NAME	VARCHAR2	25	-	-	-	-
EMAIL	VARCHAR2	25	-	-	-	-
PHONE_NUMBER	VARCHAR2	20	-	-	-	✓
HIRE_DATE	DATE	7	-	-	-	-
JOB_ID	VARCHAR2	10	-	-	-	-
SALARY	NUMBER	-	8	2	-	✓
COMMISSION_PCT	NUMBER	-	2	2	-	✓
MANAGER_ID	NUMBER	-	6	0	-	✓
DEPARTMENT_ID	NUMBER	-	4	0	-	✓
BONUS	VARCHAR2	5	-	-	-	✓

Inserción de Filas con Valores Nulos

- La sentencia INSERT no necesita especificar todas las columnas: se pueden excluir las columnas con valores nulos.
- Si a todas las columnas que necesitan un valor se les asigna un valor, la inserción funciona.

Inserción de Filas con Valores Nulos

- En nuestro ejemplo, la columna EMAIL se define como una columna NOT NULL.
- Un intento implícito de agregar valores a la tabla como el que se muestra generará un error.

```
INSERT INTO copy_employees
  (employee_id, first_name, last_name, phone_number, hire_date,
 job_id, salary)
VALUES
  (302,'Grigorz','Polanski', '8586667641', '15-Jun-2015',
 'IT_PROG',4200);
```

```
ORA-01400: cannot insert NULL into
("US_A009EMEA815_PLSQL_T01"."COPY_EMPLOYEES"."EMAIL")
```


Se ha omitido la columna EMAIL en la inserción, por lo que la inserción falla ya que no se permiten valores NULL para la columna EMAIL.

Inserción de Filas con Valores Nulos

- Una inserción implícita insertará automáticamente un valor nulo en las columnas que permiten valores nulos.
- Para agregar explícitamente un valor nulo a una columna que permite valores nulos, utilice la palabra clave NULL en la lista VALUES.

Inserción de Filas con Valores Nulos

- Para especificar cadenas vacías y/o fechas que faltan, utilice comillas simples vacía (sin espacios entre ellas de esta forma") para los datos que faltan.

```
INSERT INTO copy_employees
  (employee_id, first_name, last_name, email, phone_number,
 hire_date, job_id, salary)
VALUES
  (302,'Grigorz','Polanski', 'gpolanski', '', '15-Jun-2015',
 'IT_PROG',4200);
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
302	Grigorz	Polanski	gpolanski	-	15/Jun/2015	IT_PROG	4200

COMM_PCT	MGR_ID	DEPT_ID	BONUS
-	-	-	-

.....

NOTA: los nombres de columna en la salida se han abreviado para que quepan en la diapositiva.

Inserción de Valores Especiales

- Los valores especiales como SYSDATE y USER se pueden introducir en la lista VALUES de una sentencia INSERT.
- SYSDATE colocará la fecha y hora actuales en una columna.
- USER insertará el nombre de usuario de la sesión actual, que es OAE_PUBLIC_USER en Oracle Application Express.

Inserción de Valores Especiales

- En este ejemplo se agrega USER como apellido y SYSDATE para la fecha de contratación.

```
INSERT INTO copy_employees
  (employee_id, first_name, last_name, email, phone_number, hire_date,
 job_id, salary)
VALUES
  (304,'Test',USER, 't_user', 4159982010, SYSDATE, 'ST_CLERK',2500);
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
304	Test	APEX_PUBLIC_USER	t_user	4159982010	15-Jun-2015	ST_CLERK	2500

COMM_PCT	MGR_ID	DEPT_ID	BONUS
-	-	-	-

.....

Algunas columnas se han omitido de la salida por motivos de visualización.

Inserción de Valores de Fecha Específicos

- El modelo de formato por defecto para tipos de dato de fecha es DD-Mes-AAAA.
- Con este formato de fecha, la hora por defecto de medianoche (00:00:00) también se incluye.
- En la sección anterior, hemos aprendido cómo utilizar la función TO_CHAR para convertir una fecha en una cadena de caracteres cuando queremos recuperar y mostrar un valor de fecha con un formato que no es el formato por defecto.
- Este es un recordatorio de TO_CHAR:

```
SELECT first_name, TO_CHAR(hire_date,'Month, fmdd, yyyy')
FROM employees
WHERE employee_id = 101;
```

FIRST_NAME	TO_CHAR(HIRE_DATE,'MONTH,FMDD,YYYY')
Neena	21 de septiembre de 1989

Inserción de Valores de Fecha Específicos

- Del mismo modo, si deseamos INSERTAR una fila con un formato que no sea el formato por defecto para una columna de fecha, debemos utilizar la función TO_DATE para convertir el valor de fecha (cadena de caracteres) en una fecha.

```
INSERT INTO copy_employees
  (employee_id, first_name, last_name, email, phone_number, hire_date,
 job_id, salary)
VALUES
 (301,'Katie','Hernandez', 'khernandez','8586667641',
  TO_DATE('July 8, 2015', 'Month fmdd, yyyy'), 'MK_REP',4200);
```

Inserción de Valores de Fecha Específicos

- Un segundo ejemplo de TO_DATE permite la inserción de una hora del día concreta, sustituyendo el la hora de medianoche por defecto.

```
INSERT INTO copy_employees
  (employee_id, first_name, last_name, email, phone_number, hire_date,
 job_id, salary)
VALUES
  (303,'Angelina','Wright', 'awright','4159982010',
 TO_DATE('July 10, 2015 17:20', 'Month fmdd, yyyy HH24:MI'),
 'MK_REP', 3600);
```

```
SELECT first_name, last_name,
 TO_CHAR(hire_date, 'dd-Mon-YYYY HH24:MI') As "Date and Time"
FROM copy_employees
WHERE employee_id = 303;
```

FIRST_NAME	LAST_NAME	Fecha y Hora
Angelina	Wright	10-Jul-2015 17:20

Uso de una Subconsulta para Copiar Filas

- Cada sentencia INSERT que hemos visto hasta el momento solo agrega una fila a la tabla.
- Sin embargo, suponga que deseamos copiar 100 filas de una tabla a otra.
- No queremos tener que escribir y ejecutar 100 sentencias INSERT independientes, una tras otra.
- Eso llevaría mucho tiempo.
- Afortunadamente, SQL nos permite utilizar una subconsulta dentro de una sentencia INSERT.

NOTA: en la lección 3 de esta sección, veremos la sentencia INSERT ALL. Esta permite agregar varias filas en una sentencia SQL.

Uso de una Subconsulta para Copiar Filas

- Todos los resultados de la subconsulta se insertan en la tabla.
- Por lo tanto podamos copiar 100 filas (o 1000 filas) con una subconsulta de varias filas en INSERT.
- Como era de esperar, no necesita una cláusula VALUES al utilizar una subconsulta para copiar filas porque los valores insertados serán exactamente los valores devueltos por la subconsulta.

Uso de una Subconsulta para Copiar Filas

- En el ejemplo que se muestra, una nueva tabla denominada SALES_REPS se está rellenando con copias de algunas de las filas y columnas de la tabla EMPLEADOS.
- La cláusula WHERE selecciona aquellos empleados que tengan identificadores de trabajo como "%REP%".

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


NOTA: para ejecutar la sentencia INSERT anterior, se tendrá que crear primero la tabla sales_reps.

Uso de una Subconsulta para Copiar Filas

- El número de columnas y sus tipos de dato de la lista de columnas de la cláusula INSERT deben coincidir con el número de columnas y sus tipos de dato de la subconsulta.
- La subconsulta no está incluida entre paréntesis como se hace con las subconsultas en la cláusula WHERE de una sentencia SELECT.

Uso de una Subconsulta para Copiar Filas

- Si deseamos copiar todos los datos: todas las filas y todas las columnas: la sintaxis es aún más sencilla.
- Para seleccionar todas las filas de la tabla EMPLEADOS e insertarlos en la tabla SALES_REPS, la sentencia se escribe de la siguiente forma:

```
INSERT INTO sales_reps
 SELECT *
 FROM employees;
```

- De nuevo, esto solo funcionará si ambas tablas tienen el mismo número de columnas que coincida con los tipos de dato y que estén en el mismo orden.

Terminología

Entre los términos clave utilizados en esta lección se incluyen:

- INSERT INTO
- USER
- Transaction
- Explícito

Resumen

En esta lección, debe haber aprendido lo siguiente:

- Explicar la importancia de poder modificar los datos de una base de datos
- Construir y ejecutar sentencias INSERT que insertan una única fila con una cláusula VALUES
- Construir y ejecutar sentencias INSERT que utilizan valores especiales, valores nulos y valores de fecha
- Construir y ejecutar sentencias INSERT que copian filas de una tabla a otra mediante una subconsulta


ORACLE
ACADEMY