BASES DE DATOS

2º CURSO E.U.I. / F.I.

Práctica 3: El lenguaje SQL

2ª Parte: Definición de datos

10 DE ABRIL DE 2000

1. El lenguaje SQL del sistema ORACLE8

1.1 Definición de datos.

En el sistema ORACLE no existe el concepto de esquema de base de datos tal como aparece en el lenguaje SQL estándar. Asociado a cada usuario definido en el sistema se crea una base de datos en la que se almacenarán todos los objetos (tablas, vistas, procedimientos,...) creados por él.

De todos los elementos que pueden incluirse en un esquema SQL estándar, en el sistema ORACLE8 pueden definirse los siguientes: relaciones básicas, vistas, y privilegios de acceso. Además, pueden definirse algunos detalles relativos a la representación interna de los datos (índices, *tablespace*, *cluster*...); disparadores para modelar comportamiento activo (*triggers*); así como elementos de programación (funciones y procedimientos, paquetes de procedimientos,...).

Definición de relación básica

Definición_relación_básica ::= CREATE TABLE nom_relación
(coma*lista_elemento_relación_básica*)

elemento_relación_básica ::= *definición_atributo*
| *restricción_relación*

definición_atributo ::= nom_atributo *tipo_datos*
[DEFAULT (expresión)]
[*lista_restricción_atributo*]

tipo_datos::= | CHAR (longitud)
| VARCHAR (longitud)
| NUMBER [(precisión[, escala])]
| DATE

restricción_atributo ::= [CONSTRAINT nombre_restricción]
{[NOT] NULL
| UNIQUE
| PRIMARY KEY
| REFERENCES nom_relación* [(nom_atributo*)]
[ON DELETE CASCADE]
| CHECK (*condición*) }
[*cuando_comprobar*]

restricción_relación ::=
[CONSTRAINT nombre_restricción]
{ UNIQUE (comalista_nom_atributo)
| PRIMARY KEY (comalista_nom_atributo)
| FOREIGN KEY (comalista_nom_atributo)
REFERENCES nom_relación* [(comalista_nom_atributo*)]
[ON DELETE CASCADE]
| CHECK (*condición*)}
[*cuando_comprobar*]

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

Los tipos de datos disponibles son:

Númericos:
- NUMBER [(precisión[, escala])] donde precisión es el número total de dígitos y escala el número de cifras decimales
- enteros: NUMBER (precisión)
- reales: NUMBER (precisión, escala)

Alfanuméricos:
- de longitud fija: CHAR (longitud)
- de longitud variable: VARCHAR (longitud)

Fechas:
- DATE

La expresión de la cláusula DEFAULT se construye a partir de constantes de los tipos de datos predefinidos, operadores y funciones del sistema siguiendo la sintaxis adecuada en cada caso. En la expresión no se puede hacer referencia a otros atributos de la relación. El tipo de la expresión debe coincidir con el tipo de datos del atributo en el que se incluye la cláusula.

En ORACLE8 sólo se contempla el tipo de integridad referencial *débil* y la posibilidad de incluir la directriz de restauración de la integridad referencial *borrado en cascada.*

La condición de la cláusula CHECK es una expresión lógica que se define con la misma sintaxis que la condición de la cláusula WHERE de la sentencia SELECT con las siguientes limitaciones: sólo se puede hacer referencia a atributos de la relación sobre la que se define la restricción y no se pueden incluir subconsultas ni funciones agregadas. La relación satisface la restricción de integridad definida con la cláusula CHECK si para todas las tuplas la condición se evalúa a cierto o a indefinido (debido a la presencia de valores nulos).

En la directriz *cuando_comprobar* asociada a una restricción de integridad, la opción DEFERRABLE (resp. NOT DEFERRABLE) permite indicar dinámicamente al sistema (sentencia de SQL *SET CONSTRAINT*) si la comprobación de la restricción se debe hacer al final de la transacción (resp. después de cada sentencia SQL); el valor por

defecto es NOT DEFERRABLE. Si la restricción se ha definido como DEFERRABLE, la opción INITIALLY IMMEDIATE (resp. INITIALLY DEFERRED) indica que inicialmente la restricción se comprueba después de cada operación SQL (resp. después de la transacción); el valor por defecto es INITIALLY IMMEDIATE .

En la definición de una relación básica en ORACLE8 se pueden incluir otras cláusulas que no aparecen en la sintaxis del SQL estándar con las que se definen detalles de representación física de la relación.

Para borrar una relación se utiliza la sentencia **DROP TABLE nombre_relación.**

Modificación de relaciones

modificación_relación::= ALTER TABLE nombre_relación
{ADD (*comalista_elemento_relación_básica*)
| MODIFY (*comalista_definición_atributo*)
| {DROP
| [VALIDATE | NOVALIDATE] ENABLE
| DISABLE } (*restricción*) }

restricción::= {PRIMARY [CASCADE]
| UNIQUE (comalista_nombre_atributo) [CASCADE]
| CONSTRAINT nombre_restricción }

La opción ADD permite añadir nuevos atributos o restricciones a una relación.

La opción MODIFY permite modificar la definición de los atributos.

La opción ENABLE (resp. DISABLE) permite activar (resp. desactivar) una restricción de integridad. Con la opción VALIDATE (opción por defecto), el sistema se asegura que al activar una restricción los datos almacenados en la bases de datos en el momento de la activación satisfacen la restricción.

La opción DROP permite borrar una restricción de integridad. La opción CASCADE borra en cascada cualquier restricción de integridad definida en el esquema que dependa de la restricción que se acaba de borrar.

Definición de vistas

definición_vista::= CREATE [OR REPLACE] VIEW *nombre_vista*
[(comalista_nombre_atributo)] AS *sentencia_SELECT*
[WITH CHECK OPTION]

Debido a que una vista puede definirse a través de cualquier sentencia SELECT, la traducción de una operación de actualización sobre la vista en actualizaciones sobre relaciones básicas no es siempre posible ya que pueden presentarse ambigüedades, por

este motivo la actualización de vistas está sometida a ciertas restricciones que evitan esta posible ambigüedad.

Estas restricciones son en ORACLE8:

a) una vista definida por una SELECT que contiene operadores conjuntistas (UNION, INTERSECT,…), el operador DISTINCT, funciones agregadas (SUM, AVG, ..) o la cláusula GROUP BY no es actualizable.

b) si la vista está definida sobre una única relación básica el sistema traducirá la actualización sobre la vista en una operación de actualización sobre la relación básica siempre que no se viole ninguna restricción de integridad definida sobre dicha relación.

c) si la vista está definida sobre una concatenación de relaciones, la actualización está sometida a las siguientes restricciones adicionales:

   - la actualización sólo puede modificar una de las relaciones básicas.
   - la actualización modificará la relación básica que cumpla la propiedad de *conservación de la clave*, es decir aquella relación tal que su clave primaria podría ser también clave de la vista si sus atributos fuesen seleccionadas por la SELECT que define la vista.
   - la actualización no se realizará si viola alguna de las restricciones definidas sobre la relación básica que se va a actualizar.

Definición de privilegios

El propietario del esquema de una base de datos es el propietario de todos los objetos definidos en él. Para que otro usuario pueda realizar operaciones sobre los objetos de la base de datos debe tener la autorización necesaria; estas autorizaciones deben concederlas el propietario del objeto por medio de la sentencia GRANT.

definición_operación_grant::= GRANT *comalista_privilegios_sistema*
TO {PUBLIC | comalista_usuario}
[WITH ADMIN OPTION]

- "usuario" es el identificador de un usuario.
- con la cláusula PUBLIC se transmiten los privilegios a todos los usuarios.
- la cláusula WITH ADMIN OPTION concede permiso para ceder a terceros los privilegios obtenidos.

Los privilegios que se pueden otorgar son privilegios de administrador, es decir crear, borrar o modificar elementos de los esquemas: tablas, índices, vistas, etc.

Definición de reglas de actividad

El SGBD ORACLE8 contempla la definición de reglas de actividad (*triggers* en ORACLE) en el esquema de la base de datos.

Las reglas de actividad permiten especificar un comportamiento activo (independiente de la intervención del usuario) por parte del sistema de gestión de bases de datos, teniendo numerosas aplicaciones: comprobación de restricciones de integridad, control de la seguridad, definición de reglas de funcionamiento interno de la organización, mantenimiento de datos derivados, etc.

Independientemente de la sintaxis particular de cada SGBD, las reglas de actividad tienen la estructura: Evento-Condición-Acción, especificando por medio de estas tres componentes "una *acción* que el sistema debe ejecutar como respuesta a la ocurrencia de un *evento* cuando cierta *condición* se satisface en la base de datos".

En el sistema ORACLE8, el *evento* es cualquier operación de actualización sobre la base de datos; la *condición* es una expresión lógica escrita con la sintaxis del SQL y la *acción* es un procedimiento escrito en el lenguaje de programación PL/SQL en el que se pueden incluir operaciones de manipulación de la base de datos, de gestión de errores o de comunicación con el usuario.

definición_regla :=
{CREATE | REPLACE} TRIGGER nombre_regla
{BEFORE | AFTER | INSTEAD OF} ***evento*** [*disyunción_eventos*]
ON {nombre_relación | nombre_vista}
[[REFERENCING OLD AS nombre_referencia [NEW AS nombre_referencia]]
[FOR EACH ROW} [WHEN (***condición)***]]
bloque PL/SQL

Combinando las opciones {BEFORE | AFTER } *evento* y FOR EACH {ROW | STATEMENT} se pueden definir cuatro tipos de reglas con una semántica distinta:

	FOR EACH STATEMENT	**FOR EACH ROW**
AFTER	la regla se ejecuta una vez después de la ejecución de la operación de actualización	la regla se ejecuta una vez después de la actualización de cada tupla afectada por de la operación de actualización
BEFORE	la regla se ejecuta una vez antes de la ejecución de la operación de actualización	la regla se ejecuta una vez antes de la actualización de cada tupla afectada por de la operación de actualización

La ejecución del evento que activa una regla se intercala con la ejecución de las reglas activadas por el evento según la semántica de ejecución definida para cada tipo

de regla. Las acciones de las reglas ejecutadas por la ocurrencia de un evento extienden la transacción de la que forma parte el evento que activó las reglas.

Eventos:

{BEFORE | AFTER | INSTEAD OF} *evento* [*disyunción_eventos*]
ON {nombre_relación | nombre_vista}

disyunción_eventos := OR *evento* [*disyunción_eventos*]

evento := INSERT | DELETE | UPDATE [OF comalista_nombre_atributo]

Tipos de evento: operaciones de actualización de la base de datos

Composición de eventos: disyunción de eventos

Parametrización de eventos:

- sólo los eventos de las reglas del tipo FOR EACH ROW (orientadas a la tupla) están parametrizados
- la parametrización de los eventos es implícita: 2*n parámetros si el evento es UPDATE y n parámetros si el evento es INSERT o DELETE (siendo *n* el grado de la relación *nombre_relación* o de la vista *nombre_vista)*
- el nombre de estos parámetros es el nombre de un atributo de la relación precedido de la palabra reservada OLD si el evento es DELETE o UPDATE y de la palabra reservada NEW si el evento es INSERT o UPDATE (estas palabras reservadas se pueden sustituir por los correspondientes *nombre_referencia* asociados en la cláusula REFERENCING)
- estos parámetros pueden pasarse a la *condición de la regla*, utilizándolos directamente en ella
- estos parámetros pueden pasarse a la *acción de la regla* (*bloque PL/SQL*) utilizándolos directamente en ella. (En el bloque PL/SQL los palabras reservadas OLD y NEW deben ir precedidas de dos puntos).

Condiciones:

WHEN (condición)

Sólo las reglas del tipo FOR EACH ROW y de los tipos BEFORE y AFTER pueden incluir *condición*

Tipo de condición: la *condición de la regla* es una expresión lógica que sigue la sintaxis de la condición de la cláusula WHERE de la sentencia SELECT del SQL con las siguientes limitaciones:

- no puede contener subconsultas ni funciones agregadas
- sólo se puede hacer referencia a los parámetros del evento

(estas limitaciones reducen las posibilidades de la *condición de la regla* a expresar exclusivamente condiciones sobre la tupla actualizada y por ello muchas condiciones se expresan en el *bloque PL/SQL* que representa la *acción de la regla*)

Instanciación de la condición: si en la condición se hace referencia a parámetros del evento, éstos se instancian cuando la regla es activada

Acciones:

bloque PL/SQL

El bloque PL/SQL es un procedimiento escrito en un lenguaje de programación propio de ORACLE, cuya sintaxis se incluye en el anexo.

Tipos de acciones:

- sentencias de manipulación de la BD: INSERT, DELETE, UPDATE, SELECT...INTO....
- sentencias de programa : asignación, selección, iteración.
- sentencias de gestión de errores y de entrada-salida.

Estas sentencias pueden estructurarse utilizando las estructuras de control del lenguaje PL/SQL (selección, iteración, etc.). (ver ANEXO).

En el bloque PL/SQL no pueden incluirse sentencias de definición de datos ni de control de transacciones (COMMIT, ROLLBACK, SAVEPOINT)

Composición de acciones: la acción de la regla puede consistir en una secuencia de acciones definida por la estructura del bloque PL/SQL

Instanciación de la acción: los parámetros del evento se pueden pasar a la acción de la regla (bloque PL/SQL) haciendo referencia a ellos directamente, estos parámetros se instancian cuando se activa la regla.

Notas:

- Si la regla es del tipo BEFORE *evento* FOR EACH ROW, entonces la acción de la regla (bloque PL/SQL) puede hacer una asignación sobre NEW.nombre_atributo
- Si la regla puede ser activada por más de un tipo de evento, en el bloque PL/SQL se pueden usar los predicados lógicos INSERTING, DELETING y UPDATING ['nombre_atributo'] para programar distintas acciones en función del tipo de evento que ha activado la regla.
- En ORACLE8 en la acción de una regla (bloque PL/SQL) no puede consultarse ni actualizarse la relación actualizada por el evento de la regla.

- Las reglas de tipo INSTEAD OF sólo pueden utilizarse sobre vistas y se utilizan principalmente para programar las operaciones de actualización intentando superar las limitaciones en la actualización de vistas de los sistemas relacionales.

Lenguaje de reglas:

Creación: CREATE TRIGGER nombre_regla ……..

Borrado: DROP TRIGGER nombre_regla.

Modificación: REPLACE TRIGGER nombre_regla .....

Recompilación: ALTER TRIGGER nombre_regla COMPILE.

Consulta: consultas a las tablas del diccionario del sistema: USER_TRIGGERS, ALL_TRIGGERS, DBA_TRIGGERS

Prioridad entre reglas: no existe

Habilitar y deshabilitar reglas:

ALTER TRIGGER nombre_regla [ENABLE | DISABLE]

ALTER TABLE nombre_relación [{ENABLE | DISABLE} ALL TRIGGERS]

Modularización: no existe

Ejemplos:

a) La siguiente regla define en el esquema de la base de datos el siguiente comportamiento activo: "después de cada inserción en la relación R hacer una copia de la tupla insertada en la relación R_copia".

(El esquema de R es R(A:dom_A, B:dom_B))

```
CREATE TRIGGER T1
AFTER INSERT  ON  R
FOR EACH ROW
BEGIN
INSERT INTO R_copia VALUES(:NEW.A, :NEW.B);
END;
```

b) La siguiente regla define en el esquema de la base de datos el siguiente comportamiento activo: " después de cada operación de actualización de la relación R registrar la información sobre el usuario y la fecha de la actualización en la relación R_control"


```
CREATE TRIGGER T2
AFTER INSERT OR UPDATE OR DELETE  ON  R
FOR EACH STATMENT
BEGIN
INSERT INTO R_control VALUES(user, sysdate);
END;
```

(*user* y *sysdate* son funciones del sistema que devuelven respectivmente el usuario de la sesión y la fecha del sistema)

1.2 Manipulación de datos

Las sentencias de manipulación de datos del lenguaje SQL del sistema ORACLE8 son las mismas que las del lenguaje SQL estándar que ya han sido presentadas, con las siguientes excepciones relativas a la combinación de tablas con operadores conjuntistas y a la concatenación de tablas.

Concatenación de tablas:

Los únicos operadores de concatenación del SQL estándar que existen en ORACLE8 son el operador LEFT OUTER JOIN y el operador RIGHT OUTER JOIN con una sintaxis distinta.

Sintaxis del operador de SQL **LEFT [OUTER] JOIN** en ORACLE8:

La sentencia en SQL:

```
SELECT [ALL | DISTINCT] comalista_item_seleccionado
    FROM t1  LEFT OUTER JOIN  t2  ON  t1.A  op  t2.B AND .....
```

es equivalente en ORACLE8 a:

```
SELECT [ALL | DISTINCT] comalista_item_seleccionado
    FROM t1, t2
    WHERE  t1.A op t2.B (+) AND .....
```

Sintaxis del operador de SQL **RIGHT [OUTER] JOIN** en ORACLE8:

La sentencia en SQL:

```
SELECT [ALL | DISTINCT] comalista_item_seleccionado
  FROM t1  RIGHT OUTER JOIN  t2  ON  t1.A  op  t2.B  AND .....
```

es equivalente en ORACLE8 a:

```
SELECT [ALL | DISTINCT] comalista_item_seleccionado
        FROM t1, t2
        WHERE  t1.A (+) op t2.B AND .....
```

Si la condición de combinación del JOIN externo se define sobre varias columnas, el signo (+) deberá aparecer en cada columna que participa en la combinación en el mismo sentido: si es RIGHT en la columna de la tabla t1 y si es LEFT en la columna de la tabla t2.

Operadores de combinación conjuntista de tablas:

Sintaxis de los operadores de SQL **UNION, UNION ALL, INTERSECT y EXCEPT** en ORACLE8:

- Los operadores de SQL **UNION, UNION ALL y INTERSECT** existen con la misma sintaxis en ORACLE8.

- El operador de SQL **EXCEPT** se denomina en ORACLE 8 **MINUS** (con la misma sintaxis)

- Salvo en **UNION ALL**, todos los operadores eliminan duplicados.

2. Ejercicio de prácticas:

Se desea diseñar una base de datos para la gestión de una pequeña biblioteca de un departamento. Después de realizar el análisis del sistema, se han identificado los requerimientos que van a realizarse con más frecuencia; éstos son:

- consultar los datos de un libro: código del libro, título, autor(es), temática y en caso de estar prestado, el socio que lo tiene actualmente en préstamo.
- consultar la información sobre un socio: código del socio, nombre, dirección, teléfono y libros que actualmente tiene en préstamo así como la fecha de préstamo.
- consultar los préstamos históricos de un socio: código del libro, fecha del préstamo y fecha de la devolución.
- dar de alta, dar de baja y modificar los datos de un socio.
- gestionar los préstamos: prestar un libro a un socio y registrar la devolución de un libro.

Algunas restricciones de integridad que se han detectado son:

- el código del libro identifica unívocamente al libro.
- el código del socio identifica unívocamente al socio.

- el conjunto de temas utilizados para clasificar un libro son: física, electricidad, mecánica y óptica.
- la fecha de devolución de un libro debe ser posterior a la fecha de préstamo.
- el número total de libros que tiene prestados un socio es un *dato derivado* que será mantenido automáticamente por el sistema.

Realizar las siguientes tareas:

a) definir el esquema relacional de la base datos anterior (usando los conceptos del modelo relacional).

b) definir la base de datos en el sistema ORACLE8.

c) realizar actualizaciones y consultas sobre la base de datos creada.

Anexo: Lenguaje PL/SQL

Estructura de un bloque PL/SQL:

DECLARE

Sección de declaración de variables

BEGIN

Sentencias del bloque

END

Sección de declaración de variables:

- Variables locales al bloque:

nombre_variable tipo_dato

tipo_dato::= {**NUMBER** | **CHAR**( ) | **DATE** }

(las declaraciones de esta sección deben ir separadas por punto y coma)

Sentencias del cuerpo del bloque PL/SQL:

secuencia_de_sentencias::= sentencia; [sentencia;] ...

- Sentencia de selección:

IF condición **THEN** secuencia_de_sentencias

[**ELSE** secuencia_de_sentencias] **END IF;**

- Sentencias de repetición:

WHILE condición **LOOP**

secuencia_de sentencias ;

END LOOP;

FOR *contador* **IN** *mínimo .. máximo* **LOOP**

secuencia_de_sentencias ;

END LOOP;

- Asignación :

nombre_variable **:=** *expresión*

- Sentencias SQL: **INSERT, DELETE, UPDATE, SELECT... INTO**.....

- Sentencias de entrada-salida: dbms_output.put_line ('*mensaje*'). Para usar esta función el paquete dbms_output debe estar activado, esto se hace con la sentencia SQL **SET SERVEROUTPUT ON**

Manejo de errores:

Si durante la ejecución de una regla (trigger) se produce un error predefinido en el sistema o definido por el usuario, entonces se anulan todas las actualizaciones realizadas por la acción de la regla así como el evento que la activó.

La sentencia **RAISE_APPLICATION_ERROR** (*nro_error*, ' *mensaje'*) provoca la ocurrencia del error de número interno *nro_error* y envía al usuario el mensaje '*mensaje*'. *(nro_error* debe ser un número negativo entre -20000 y -20999).