UNIVERSIDAD TECNOLÓGICA DE PEREIRA

Ingeniería en Sistemas y Computación

Bases de Datos. (IS644 Gr 100)

Taller. 02/05/2014

PERMISOS O PRIVILEGIOS

Privilegios

- **Privilegios del sistemas:** Obtener acceso a la base de datos
- **Privilegios a Objetos:** Manipulación de contenido de objetos de la base de datos
- **Schemas:** Colección de objetos, tales como tablas vistas y secuencias

PRIVILEGIOS DEL SISTEMA

- **Creación de usuarios**
- **Eliminación de usuarios**
- **Remover tablas**
- **Backup's de tablas**

Creación de usuarios

Sintaxis: **Creación de Usuarios**

```
CREATE USER user_name
   IDENTIFIED { BY password
                    | EXTERNALLY [ AS 'certificate_DN' ]
                    | GLOBALLY [ AS '[ directory_DN ]' ]
                    }
   [ DEFAULT TABLESPACE tablespace
   | TEMPORARY TABLESPACE
         { tablespace | tablespace_group }
   | QUOTA integer [ K | M | G | T | P | E ]
           | UNLIMITED }
           ON tablespace
     [ QUOTA integer [ K | M | G | T | P | E ]
           | UNLIMITED }
                ON tablespace
     ]
   | PROFILE profile_name
   | PASSWORD EXPIRE
   | ACCOUNT { LOCK | UNLOCK }
       [ DEFAULT TABLESPACE tablespace
       | TEMPORARY TABLESPACE
            { tablespace | tablespace_group }
       | QUOTA integer [ K | M | G | T | P | E ]
                | UNLIMITED }
                ON tablespace
          [ QUOTA integer [ K | M | G | T | P | E ]
                | UNLIMITED }
                ON tablespace
           ]
```

```
    | PROFILE profile
    | PASSWORD EXPIRE
    | ACCOUNT { LOCK | UNLOCK } ]
   ] ;
```

Ejemplo:

Comando	Explicación
1. CREATE USER sidney 2. IDENTIFIED BY out_standing1 3. DEFAULT TABLESPACE example 4. QUOTA 10M ON example 5. TEMPORARY TABLESPACE temp 6. PROFILE app_user 7. PASSWORD EXPIRE;	1. Crea un usuario llamado sidney 2. Define la clave como out_standing1 3. Define el lugar de almacenamiento example 4. Define el tamaño máximo a utilizar 10M 5. Define el lugar de almacenamiento temporal 6. Define el perfil a utilizar 7. Define que el password tiene vencimiento

```
CREATE USER    Scott IDENTIFIED BY tiger;
```

Privilegios del sistema para usuarios

Sintaxis:

GRANT privilegio [, privilegio...] TO user [, user| role, PUBLIC...];

Privilegios del sistema:

- CREATE SESSION
- CREATE TABLE
- CREATE SEQUENCE
- CREATE VIEW
- CREATE PROCEDURE

Ejemplo:

GRANT create session, create table, create sequence, create view TO pablo;

Cambiando el Password

Sintaxis:

```
ALTER USER user IDENTIFIED BY password;
```

Ejemplo:

```
ALTER USER scott IDENTIFIED BY lion;
```

Privilegios de Objetos

Privilegios de objetos	Table	View	Sequence	Procedure
ALTER	√		√	
DELETE	√	√		
EXECUTE				√
INDEX	√			
INSERT	√	√		
REFERENCES	√	√		
SELECT	√	√	√	

Privilegios de objetos

Sintaxis:

GRANT object_priv [(columns)] ON object TO{user|role|PUBLIC} [WITH GRANT OPTION];

Ejemplos:

```
GRANT  select ON employees TO sue, rich;

GRANT  update (department_name, location_id)
ON    departments TO scott, manager;
```

Clausulas WITH GRANT OPTION y PUBLIC

Dar permiso a un usuario para dar permisos a otro usuario:

GRANT select, insert ON departments TO Scott WITH GRANT OPTION;

Dar permiso a todos los usuarios para consulta de información:

GRANT select ON alice.departments TO PUBLIC;

Vistas el sistema para confirmar privilegios

Data Dictionary View	Description
ROLE_SYS_PRIVS	System privileges granted to roles
ROLE_TAB_PRIVS	Table privileges granted to roles
USER_ROLE_PRIVS	Roles accessible by the user
USER_TAB_PRIVS_MADE	Object privileges granted on the user's objects
USER_TAB_PRIVS_RECD	Object privileges granted to the user
USER_COL_PRIVS_MADE	Object privileges granted on the columns of the user's objects
USER_COL_PRIVS_RECD	Object privileges granted to the user on specific columns
USER_SYS_PRIVS	Lists system privileges granted to the user

Retirando privilegios a objetos

Sintaxis:

REVOKE {privilege [, privilege...]|ALL} ON object FROM {user[, user...]|role|PUBLIC} [CASCADE CONSTRAINTS];

Ejemplo:

REVOKE select, insert ON departments FROM scott;

COMANDOS BÁSICOS PL/SQL

Operadores PL/SQL

Tipo de operador	Operadores
Operador de asignación	**:= (dos puntos + igual)**
Operadores aritméticos	**+ (suma)** **- (resta)** *** (multiplicación)** **/ (división)** **** (exponente)**
Operadores relacionales o de comparación	**= (igual a)** **<> (distinto de)** **< (menor que)** **> (mayor que)** **>= (mayor o igual a)** **<= (menor o igual a)**
Operadores lógicos	**AND (y lógico)** **NOT (negacion)** **OR (o lógico)**
Operador de concatenación	**||**

Condicionales: IF ELSE

```
IF (expresión) THEN
        -- Instrucciones
ELSIF (expresión) THEN
        -- Instrucciones
ELSE -- Instrucciones
END IF;
```

Ciclos: LOOP

```
LOOP -- Instrucciones
        IF (expresion) THEN
                -- Instrucciones
                EXIT;
        END IF;
END LOOP;
```

Ciclos: WHILE

```
WHILE (expresion) LOOP
```

```
        -- Instrucciones
END LOOP;
```

Ciclo: For

```
FOR contador IN [REVERSE] inicio..final LOOP
        -- Instrucciones
END LOOP;
```

Estructura de un bloque

```
[ DECLARE | IS | AS ]
        /*Parte declarativa*/
BEGIN
        /*Parte de ejecución*/
        [ EXCEPTION ]
        /*Parte de excepciones*/
END;
```

Atrapando una Excepciones

Adición de la cláusula EXCEPTION entre las palabras reservadas BEGIN y END

Ejemplo

```
DECLARE
 -- Declaramos una excepcion identificada por VALOR_NEGATIVO
VALOR_NEGATIVO EXCEPTION;
valor NUMBER;
BEGIN
    -- Ejecucion
     valor := -1;
     IF valor < 0 THEN
           RAISE VALOR_NEGATIVO;
     END IF;

     EXCEPTION
          -- Excepcion
     WHEN VALOR_NEGATIVO THEN
            dbms_output.put_line('El valor no puede ser negativo');
END;
```

Códigos de excepción

Excepcion	Se ejecuta ...	SQLCODE
ACCESS_INTO_NULL	El programa intentó asignar valores a los atributos de un objeto no inicializado	-6530
COLLECTION_IS_NULL	El programa intentó asignar valores a una tabla anidada aún no inicializada	-6531
CURSOR_ALREADY_OPEN	El programa intentó abrir un cursor que ya se encontraba abierto. Recuerde que un cursor de ciclo FOR automáticamente lo abre y ello no se debe especificar con la sentencia OPEN	-6511
DUP_VAL_ON_INDEX	El programa intentó almacenar valores duplicados en una columna que se mantiene con restricción de integridad de un índice único (unique index)	-1
INVALID_CURSOR	El programa intentó efectuar una operación no válida sobre un cursor	-1001
INVALID_NUMBER	En una sentencia SQL, la conversión de una cadena de caracteres hacia un número falla cuando esa cadena no representa un número válido	-1722
LOGIN_DENIED	El programa intentó conectarse a Oracle con un nombre de usuario o password inválido	-1017
NO_DATA_FOUND	Una sentencia SELECT INTO no devolvió valores o el programa referenció un elemento no inicializado en una tabla indexada	100

Excepcion	Se ejecuta ...	SQLCODE
NOT_LOGGED_ON	El programa efectuó una llamada a Oracle sin estar conectado	-1012
PROGRAM_ERROR	PL/SQL tiene un problema interno	-6501
ROWTYPE_MISMATCH	Los elementos de una asignación (el valor a asignar y la variable que lo contendrá) tienen tipos incompatibles. También se presenta este error cuando un parámetro pasado a un subprograma no es del tipo esperado	-6504
SELF_IS_NULL	El parámetro SELF (el primero que es pasado a un método MEMBER) es nulo	-30625
STORAGE_ERROR	La memoria se terminó o está corrupta	-6500
SUBSCRIPT_BEYOND_COUNT	El programa está tratando de referenciar un elemento de un arreglo indexado que se encuentra en una posición más grande que el número real de elementos de la colección	-6533
SUBSCRIPT_OUTSIDE_LIMIT	El programa está referenciando un elemento de un arreglo utilizando un número fuera del rango permitido (por ejemplo, el elemento “-1”)	-6532

Excepción	Se ejecuta ...	SQLCODE
NOT_LOGGED_ON	El programa efectuó una llamada a Oracle sin estar conectado	-1012
PROGRAM_ERROR	PL/SQL tiene un problema interno	-6501
ROWTYPE_MISMATCH	Los elementos de una asignación (el valor a asignar y la variable que lo contendrá) tienen tipos incompatibles. También se presenta este error cuando un parámetro pasado a un subprograma no es del tipo esperado	-6504
SELF_IS_NULL	El parámetro SELF (el primero que es pasado a un método MEMBER) es nulo	-30625

STORAGE_ERROR	La memoria se terminó o está corrupta	-6500
SUBSCRIPT_BEYOND_COUNT	El programa está tratando de referenciar un elemento de un arreglo indexado que se encuentra en una posición más grande que el número real de elementos de la colección	-6533
SUBSCRIPT_OUTSIDE_LIMIT	El programa está referenciando un elemento de un arreglo utilizando un número fuera del rango permitido (por ejemplo, el elemento “-1”)	-6532

CREACIÓN DE DISPARADORES O (TRIGGERS)

Creación de triggers o disparadores

Sintaxis:

```
CREATE [OR REPLACE] TRIGGER <nombre_trigger>
{BEFORE|AFTER}
{DELETE|INSERT|UPDATE [OF col1, col2, ..., colN]
[OR {DELETE|INSERT|UPDATE [OF col1, col2, ..., colN]...]}
ON <nombre_tabla>
[FOR EACH ROW [WHEN (<condicion>)]]
DECLARE
  -- variables locales
BEGIN
  -- Sentencias
[EXCEPTION]
  -- Sentencias control de excepcion
END <nombre_trigger>;
```

Ejemplo:

```
CREATE OR REPLACE TRIGGER author_bir
        BEFORE INSERT ON author
        FOR EACH ROW
BEGIN
        If upper(:new.name) = 'JONATHAN GENNICK' then
                raise_application_error(20000, 'Sorry, that genius is not allowed.');
        end if;
 END;
```

Disparadores: variables :OLD (información anterior) y :NEW (nueva información)

Disparadores: predicados (inserting, updating y deleting)

Procedimientos Anónimos

```
-- Procedimiento anónimo ya que no tiene el nombre del procedimiento
DECLARE
/*
        zona de declaración cursores locales, variables y métodos,
        pero no es obligatorio
*/
BEGIN
-- Zona de instrucciones

NULL; -- Si olvido alguna sentencia!
EXCEPTION
     when NO_DATA_FOUND then
     raise_application_error(-20000, 14 ‘sección de manipulación de errores');
END;
/
-- slash dice ejecute este procedimiento
```

Procedimientos Almacenados

```
    CREATE [OR REPLACE] PROCEDURE <procedure_name> [(
        <parameter_name_1> [IN] [OUT] <parameter_data_type_1>,
        <parameter_name_2> [IN] [OUT] <parameter_data_type_2>,...
        <parameter_name_N> [IN] [OUT] <parameter_data_type_N> )] IS

        --zona de declaracion
    BEGIN
        -- zona de comando o ejecucion
        return <return_data_type>;
        EXCEPTION
        -- zona de manejo de excepciones
    END;
```

Funciones

```
  CREATE [OR REPLACE] FUNCTION <function_name> [(
    <parameter_name_1> [IN] [OUT] <parameter_data_type_1>,
    <parameter_name_2> [IN] [OUT] <parameter_data_type_2>,...
    <parameter_name_N> [IN] [OUT] <parameter_data_type_N> )]

    RETURN <return_data_type> IS
    --zona de declaracion

  BEGIN
    -- zona de comando o ejecución
    return <return_data_type>;
    EXCEPTION
    -- zona de manejo de excepciones
  END;
```

Taller Privilegios-Procedimientos-Indices

1. Crear una conexión a la base de datos con el usuario SYS.
2. Crear un usuario llamado usuarioPrueba con clave usuarioPrueba.
3. Creer una conexión a la base de datos con el usuario usuarioPrueba. Funciono? Por qué no funcionó de ser el caso? Corregir este problema si lo tuvo.
4. Cree un tabla llamada Secuenciador con dos campos (id: Numeric, dato varchar2(30)). Funciono? Por qué no funcionó de ser el caso? Corregir este problema si lo tuvo.
5. Insertar 3 registros en la tabla Secuenciador
6. Que debe hacer para consultar la tabla de staff? Realizar las tareas necesarias para que se pueda realizar esta labor.
7. Insertar 3 registros sobre la tabla de staff. Funciono? Por qué no funcionó de ser el caso? Corregir este problema si lo tuvo.
8. Modificar los registros insertados con nueva información. Funciono? Por qué no funcionó de ser el caso? Corregir este problema si lo tuvo.
9. Eliminar los registros actualizados. Funciono? Por qué no funcionó de ser el caso? Corregir este problema si lo tuvo.
10. Permitir que el usuario dueño de la tabla staff consultar la información de la tabla de Secuenciador. Adicione otros 3 registros al secuenciador, desde la conexión del usuario dueño de la tabla staff.
11. Conectarse con el usuario usuarioPrueba.
12. Eliminar todos los registros de la tabla secuenciador
13. Aplicar los cambios.
14. Llenar la tabla secuenciador con el siguiente procedimiento anónimo (analizar este código):

```
DECLARE
        i NUMBER;
BEGIN
        i := 0;
        FOR i IN 1..1000000 LOOP
          INSERT INTO secuenciador (id, texto) VALUES (i, DBMS_RANDOM.STRING ('p', 30));
        END LOOP;
        COMMIT;
END;

/*  parametros de la function DBMS_RANDOM.STRING
opt - The kind of string generated. It's a single character value:
u - uppercase alphabetic characters; no numbers, e.g. 'ABCZ'
l - lowercase alphabetic characters; no numbers, e.g. 'abcz'
a - mixed case alphabetic characters; not numbers, e.g. 'AbCz';
x - uppercase alpha-numeric characters, e.g. 'A1B4X'
p - any printable characters, e.g. 'svO}>,&=RZ'
*/
```

15. Copie en un archivo la información del campo texto de los registros nro 1, 1000, 50.000, 100.000, 500.000, 750.000 y 1´000.000. Registro el tiempo de toma en consultar cada uno de esos datos.
16. Ahora busque los registros con base al dato (texto) que almacenó en el punto anterior), registre los tiempos y compárelos con los punto anterior. Cual es más rápido y por qué?
17. Busque los elementos repetidos del campo de tipo texto y elimínelos.
18. Aplique los cambios.
19. Cree dos índices uno llamado inx_id que será llave primaria y otro índice que será único cuyo nombre será inx_texto y que tendrá el campo de texto.
20. Repita los pasos 15 y 16 y compare y explique los resultados.

21. Enviar al correo del profesor la solución del taller.

HACERLO INDIVIDUAL PARA QUE APRENDAN.