

# Administración

---

La administración de una base de datos es una de las tareas más importante para el buen funcionamiento de la misma. En este capítulo se exponen el uso del lenguaje de control de datos para concesión y eliminación de privilegios del sistema; creación, modificación y borrado de diferentes objetos. También se expone cómo arrancar y parar la base de datos y cómo se debe de llevar el control del espacio de disco. Al final se dedica un apartado a la exposición de dos aplicaciones que se utilizan, principalmente, para la realización de copias de seguridad, migraciones de software y/o reorganizaciones físicas.

## 8.1. Introducción

Una base de datos es un conjunto coherente de información compartida por varios usuarios. El hecho de que haya un número determinado de usuarios exige una gestión rigurosa del acceso a las informaciones. Esta gestión permitirá asegurar dos propiedades esenciales para una base de datos:

- La *confidencialidad* de los datos que consiste en poner a disposición de los usuarios autorizados una determinada información e impedir a los no autorizados su acceso a la misma. Esta tarea se controla mediante la gestión de los permisos de acceso.
- La *integridad* de los datos que consiste en verificar si los datos insertados y modificados en la base son correctos y verifican las reglas que gestionan el sistema de información. Esta tarea se denomina control de integridad de la base de datos.

El control de integridad está asegurado por la aplicación de una serie de reglas llamadas *restricciones de integridad*<sup>1</sup>. Éstas se especifican y aplican en los programas que manipulan los datos de la base. Por tanto, esta tarea es propia de los diseñadores y desarrolladores de las aplicaciones.

Por el contrario, la gestión de los permisos de acceso es una tarea que debe ser compartida entre los diseñadores y el Administrador de la Base de Datos (*DBA*). Los diseñadores, por su conocimiento de las diferentes etapas del sistema de información, seleccionan los diferentes grupos de usuarios con sus permisos de acceso a los objetos de la base. Por otra parte, el administrador de la base de datos tiene la responsabilidad de utilizar o no la elección hecha por los diseñadores<sup>2</sup>.

## 8.2. Privilegios del sistema

En el plano de los privilegios, el principio fundamental es que un usuario no pueda efectuar una operación cualquiera salvo si tiene el privilegio apropiado para esta operación.

La primera opción a definir es el entorno en el que el usuario se coloca cuando se conecta al sistema. Dicho de otra manera, ¿cuáles son los objetos que puede manipular mediante la posesión de un privilegio apropiado? El entorno de un usuario de Oracle está formado por el conjunto de objetos creados por él, los objetos a los que tiene acceso con permiso de otros usuarios y los recursos del sistema que tiene concedidos.

Oracle proporciona unas características de seguridad extensas para conseguir proteger la información, tanto de personas desautorizadas, como de daños intencionados o inadvertidos. Esta seguridad está proporcionada por la concesión y supresión de privilegios sobre las bases de privilegio a privilegio, persona a persona, conjunta e independientemente del sistema de seguridad que ya posea el ordenador. En nuestro caso, cada usuario tiene un nombre y una clave de acceso a Oracle, y es propietario de sus objetos y recursos concedidos por otros usuarios.

El administrador de la base de datos puede conceder privilegios del sistema a los usuarios de dos maneras distintas. Una, concediendo uno a uno los privilegios, o bien utilizando el concepto de función (también llamadas *roles* o papeles).

Una función Oracle es un objeto de la base de datos que posee un nombre y tiene asignado un conjunto de privilegios (o tipo de acceso que cada usuario requiere según su estatus y responsabilidades). Se pueden conceder o asignar privilegios específicos a las funciones, y a continuación asignar las funciones a los usuarios adecuados. Esto facilita la concesión y eliminación de privilegios a los usuarios.

Los privilegios del sistema de la base de datos permiten ejecutar un conjunto concreto

---

<sup>1</sup>Ver capítulo 2.

<sup>2</sup>Hay que resaltar que no siempre el administrador de la base de datos es diferente de los diseñadores de ésta.

de órdenes, mientras que los privilegios de objetos de la base de datos permiten realizar ciertas operaciones sobre varios objetos. Se conceden o se revocan utilizando las órdenes del lenguaje de control de datos (DCL).

### 8.2.1. Lo que puede conceder el administrador

Oracle ofrece tres *roles*, funciones o papeles que se crean en el momento de la creación de la base de datos. Estas tres funciones son compatibles con los privilegios que existían en las anteriores versiones. Las funciones son: `connect`, `resource` y `dba`<sup>3</sup>. Cada uno de éstas confieren derechos y poderes al usuario. Algunos de los derechos que se reservan para el administrador no se dan nunca o no los necesitan los usuarios normales. Puede ocurrir que algunos usuarios necesiten más privilegios que los demás, por supuesto serán casos muy concretos. Es entonces cuando el administrador debe estudiar estos casos y conceder los privilegios solicitados o no concederlos.

### 8.2.2. La función `connect`

Los usuarios, cuyo trabajo con la base de datos sea solamente de consulta a través de niveles externos, son a los que se concede el `role connect`. Esta función es simplemente el privilegio de usar Oracle. Este derecho llega a ser significativo con la añadidura de acceder a tablas específicas que pertenecen a otros usuarios y el privilegio de utilizar las órdenes del lenguaje de manipulación de datos (DML).

La tabla 8.1 recogen algunos de los privilegios del sistema asociado a la función `connect`.

### 8.2.3. La función `resource`

Esta función se suele conceder a usuarios que necesiten realizar un trabajo más complejo con la base de datos que incluya programación y manejo de estructuras más complejas.

Esta función no tiene restricciones en términos de cantidad de espacio de disco que puede ser demandado por el usuario para crear cualquier objeto.

Un usuario que tenga asignada la función `resource` puede crear sus objetos:

- Sin restricción de `tablespace`.
- Sin restricción de cuota de espacio en disco.

---

<sup>3</sup>Oracle Corporation recomienda que el DBA diseñe sus propias funciones ya que la seguridad de la base de datos depende de ellas. Estas funciones pueden que no se creen automáticamente en versiones futuras de Oracle.

Privilegio	Acción
<code>alter session</code>	Permite modificar los parámetros que configuran nuestra sesión de trabajo.
<code>create cluster</code>	Permiso para la creación de agrupamientos en nuestro esquema.
<code>create database link</code>	Permite crear enlaces a bases de datos privadas en nuestro esquema.
<code>create sequence</code>	Permiso para crear secuencias en nuestro esquema.
<code>create session</code>	Permiso para conectarse a la base de datos.
<code>create synonym</code>	Permiso para crear sinónimos en nuestro esquema.
<code>create table</code>	Permite crear tablas en nuestro esquema, si se nos ha dado cuota en el <code>tablespace</code> que va a contener a la tabla.
<code>create view</code>	Nos permite crear vistas en nuestro esquema.

Tabla 8.1: Privilegios asociados al `role connect`

Puede ocurrir que el administrador ponga una restricción sobre algún `tablespace`, en este caso sólo se pueden crear objetos en los `tablespace` especificados por el administrador. Lo normal es asignar un `tablespace` de trabajo por defecto<sup>4</sup> y que sólo se puedan crear objetos en él.

En la tabla 8.2 se muestran algunos de los privilegios del sistema asociados al `role resource`.

Privilegio	Acción
<code>create cluster</code>	Permiso para la creación de agrupamientos en nuestro esquema.
<code>create procedure</code>	Permiso para crear procedimientos almacenados, funciones y paquetes en nuestro esquema.
<code>create sequence</code>	Permiso para crear secuencias en nuestro esquema.
<code>create table</code>	Permite crear tablas en nuestro esquema.
<code>create trigger</code>	Nos permite crear un disparador en nuestro esquema.

Tabla 8.2: Privilegios asociados al `role resource`

#### 8.2.4. La función `dba`

Un usuario con el `role dba` tiene todos los privilegios del sistema y acceso a todos los objetos de todos los usuarios, puede dar o quitar los privilegios del sistema y cambiar las contraseñas de los usuarios. Además, el usuario que tiene este `role dba` puede realizar

<sup>4</sup>Normalmente al crear un usuarios se le asigna un `tablespace` por defecto. Ver pregunta 8.5.1.

los mandatos propios de un administrador. En la tabla 8.3 se especifica algunos de los privilegios que tiene.

Privilegio	Acción
<code>grant any privilege</code>	Tiene todos los privilegios del sistema y los puede conceder o no a los usuarios y/o funciones y puede hacer que los usuarios y/o las funciones los puedan trapasar a otros con la cláusula <code>with admin option</code> .
<code>grant any role</code>	Tiene permiso para conceder cualquier función a cualquier usuario y/o función.
<code>exp_full_database</code>	Esta función permite exportar a un fichero del <i>SO</i> toda los objetos de una base de datos incluyendo los privilegios.
<code>imp_full_database</code>	Este <code>role</code> permite importar toda una base de datos contenida en un fichero del <i>SO</i> a una base de datos, restaurando todos los objetos y privilegios de dicha base de datos.

Tabla 8.3: Privilegios asociados al `role dba`

Un usuario con el `role dba` posee espacio ilimitado en los `tablespace`, y la capacidad de asignar o revocar cuota en cualquier `tablespace` a cualquier usuario.

Una base de datos Oracle puede tener más de un usuario que tenga asignada la función `dba`, aunque es aconsejable que este número sea bajo ya que dispone de todos los privilegios del sistema. Hace falta, por tanto, limitar este privilegio a un número muy reducido de personas por motivos de seguridad.

Sólo el administrador puede crear a otros usuarios. Cuando crea un nuevo usuario, debe asignarle un perfil<sup>5</sup> compuesto por los recursos del sistema que se le han asignado, un nombre de usuario y una palabra de paso, opcionalmente los `tablespace` y las cuotas, e informar a dicho usuario de la asignación de todas estas restricciones.

### 8.3. Concesión de privilegios del sistema

Los privilegios del sistema se conceden con la orden del lenguaje de control de datos `grant`. Esta orden asigna uno o más privilegios del sistema a usuarios y/o funciones. Un privilegio del sistema es una autorización para utilizar alguna de las distintas órdenes de definición o control de datos.

A los usuarios se les asignan privilegios del sistema para permitirles entrar en la base de datos y realizar operaciones. Un usuario que no tenga concedido ningún privilegio del sistema no puede realizar absolutamente nada en Oracle.

<sup>5</sup>Ver preguntas 8.5.1 y 8.5.2.

Como ya hemos visto, una función es un objeto de la base de datos que tiene asociada un conjunto de privilegios que se obtienen mediante una o varias órdenes `grant`.

Sólo el administrador o cualquier otra persona con semejante autoridad, o sea que tenga concedida la función `dba`, puede conceder o anular los privilegios del sistema a otros usuarios.

### Formato:

```
GRANT {priv_sistema|función} [, {priv_sistema|función}] ...  
TO {esquema|función|public} [, {esquema|función|public}] ...  
[WITH ADMIN OPTION];
```

*priv\_sistema* es uno de los privilegios del sistema que se va a otorgar.

*función* es el nombre de una función o role.

`to` indica a qué esquema, función o `public` se le van a conceder los privilegios del sistema.

`public` concede privilegios del sistema o funciones a todos los usuarios.

`with admin option` permite que el usuario o función que recibe el o los privilegios del sistema y/o la o las funciones puedan concederlos a otros usuarios o funciones. Quien recibe este privilegio puede incluso modificar o suprimir la función asignada con `with admin option`.

Los privilegios del sistema se pueden conceder a un usuario o esquema, a una función o a `public`. Además, una función se puede asignar a una esquema, a otra función o a `public`. Esto nos permite conceder paquetes de privilegios en vez de hacerlo uno a uno tanto a esquemas como a otras funciones.

### 8.3.1. Eliminación de privilegios del sistema

La orden `revoke` retira privilegios o funciones a usuarios o a funciones. Se puede eliminar cualquier privilegio del sistema.

Si se le revoca un privilegio a un usuario, ya no podrá ejecutar las operaciones permitidas por el privilegio. Si se le retira un privilegio a una función, ningún usuario al que se le haya asignado dicha función podrá ejecutar las operaciones, a menos que le sean permitidas por otra función. Si retiramos un privilegio `public`, ningún usuario que haya recibido el privilegio a través de `public` podrá ejecutar las operaciones permitidas por el privilegio.

Si retiramos una función a un usuario, éste ya no podrá activarla, pero puede continuar ejerciendo el privilegio dentro de la sesión activa. Si retiramos una función a una función,

los usuarios que la tenían asignada ya no podrán activarla, pero podrán continuar usando el privilegio durante la sesión activa. Si se retira una función a `public`, los usuarios que hayan recibido la función a través de `public` ya no podrán activarla.

**Formato:**

```
REVOKE {priv_sistema|función} [, {priv_sistema|función}]
FROM {esquema|función|public} [, {esquema|función|public}];
```

`from` indica a qué esquema, función o `public` se le van a revocar los privilegios del sistema.

Se pueden eliminar todos los privilegios a un usuario o a una función pero esto no quiere decir que se elimine al usuario o a la función de Oracle, ni que se destruyan los objetos que el usuario ha creado; simplemente quiere decir que se prohíbe al usuario que acceda a la base de datos. Otros usuarios a los que se les ha proporcionado la capacidad de acceder a dichos objetos, continuarán exactamente con la misma capacidad de acceso.

## 8.4. La orden `create`

La orden `create` es una orden de definición de datos, como ya hemos visto en el capítulo 2, que se utiliza para crear y definir las estructuras de los datos. En una base de datos existen otros objetos, además de los ya estudiados, que se necesitan para el funcionamiento de una base de datos Oracle. En este apartado vamos a describir cómo se crean a los usuarios, los perfiles y las funciones.

## 8.5. Creación de la base de datos

Una vez instalado el producto, pueden crearse otras bases de datos. Para ello, se puede utilizar el procedimiento estándar de Oracle o hacerlo manualmente con la orden `create database` bajo una cuenta privilegiada, o sea, que tenga la función `dba` o bien la función `SYSDBA`.

```
CREATE DATABASE [nom_base_datos]
[DATAFILE especific_fichero [, especific_fichero,...]]...;
```

*nom\_base\_datos* es el nombre de la base de datos. Es opcional y si no le damos nombre Oracle tomará el valor que tenga la variable de inicialización `DB_NAME`.

*especific\_fichero* es el nombre completo de un fichero. Este nombre dependerá del sistema operativo que estemos usando.

En cualquier caso, es preciso para la nueva base:

- Un identificador externo de la base, llamado `SID`<sup>6</sup>. Este identificador, que contiene una cadena de caracteres, permite identificar los procesos de la *instance* cuando está activa.
- Un nombre para la base de datos<sup>7</sup>.
- Un nombre para el fichero que contenga los parámetros de inicialización, generalmente *init.ora*<sup>8</sup>.
- Un nombre para los ficheros de control.
- Un nombre para los ficheros de recuperación y
- Un nombre y un tamaño para cada fichero base.

### 8.5.1. Creación de un usuario

Oracle trata a cada usuario de la base de datos como un objeto al igual que hace con las tablas, vistas, etc. La información de los usuarios queda registrada en una tabla del diccionario de datos y se puede consultar utilizando las vistas que hemos visto en el capítulo 4.

En el momento de la creación, a cada usuario se le asigna un nombre de usuario, una palabra de acceso, una ubicación de sus objetos, por defecto, una cuota de disco, y todos los recursos necesarios para la utilización de Oracle lo mejor posible. Estos recursos se otorgan asignándole al usuario un perfil.

#### Formato:

```
CREATE USER usuario IDENTIFIED
{BY clave | EXTERNALLY | ...}
[ {DEFAULT TABLESPACE nom_tablespace
| TEMPORARY TABLESPACE nom_tablespace
| [QUOTA {entero [K | M] | UNLIMITED} ON nom_tablespace
| QUOTA {entero [K | M] | UNLIMITED}
ON nom_tablespace} ...
| PROFILE nom_perfil} ...];
```

*usuario* es el nombre del usuario o esquema que se va a crear<sup>9</sup>.

<sup>6</sup>En nuestra base de datos `$ORACLE_SID=orcl`.

<sup>7</sup>Suponemos que nuestra base de datos tiene por nombre `orcl`.

<sup>8</sup>También puede llamarse *initORACLE\_SID.ora*. Nosotros tenemos ficheros llamados *init.ora* e *initorcl.ora*.

<sup>9</sup>Ver longitud o tamaño de los nombres de los objetos en el capítulo 2.

*clave* es el conjunto de caracteres que forman la palabra de acceso o contraseña<sup>10</sup>.

*externally* es un acceso directo a la base de datos sin necesidad de introducir contraseña, por tanto Oracle comprobará que los datos corresponden con los del *SO*<sup>11</sup>.

*default tablespace* es el *tablespace* donde va a crear el usuario sus objetos, si es que no especifica otro espacio.

*temporary tablespace* es el *tablespace* para la creación de los objetos temporales.

*entero* especifica el tamaño máximo que se le designa en el *tablespace* especificado en *on* para la creación de objetos. Vendrá expresado en *kilobytes* o en *megabytes*.

*profile* asigna el perfil denominado *nom\_perfil* al usuario para limitar el uso de recursos del sistema. Si no especificamos un perfil, Oracle asignará el perfil *default*<sup>12</sup> al usuario.

Esta orden solamente crea un esquema o una cuenta de usuario en la base de datos Oracle. Este usuario no tiene asignados privilegios por lo que no podrá acceder al producto Oracle. Para ello, será necesario asignar funciones y privilegios al usuario con la orden *grant*. El privilegio mínimo que se le debe asignar a un usuario para que pueda acceder a Oracle es *create session* (privilegio que está incluido en el *role connect*).

### 8.5.2. Creación de un perfil

Los perfiles son objetos de Oracle al que se le asigna recursos del sistema. Los usuarios poseen recursos a través de los perfiles que tengan concedidos. Es una manera de llevar un control de qué y quienes pueden utilizar todos los medios que tiene el sistema.

El administrador es la persona que puede crear perfiles y otorgarlos a los usuarios. Oracle dispone de un perfil por defecto que se asigna al usuario si no se especifica ningún en el momento de la creación de dicho usuario.

#### Formato:

```
CREATE PROFILE nom_perfil LIMIT
  {{SESSIONS_PER_USER | CPU_PER_SESSION
  | CONNECT_TIME | ...}}
  {entero | UNLIMIT | DEFAULT} |
  {PRIVATE_SGA {entero [K|M] | UNLIMIT | DEFAULT}} ...};
```

*nom\_perfil* es el nombre que le vamos a dar al perfil.

---

<sup>10</sup>En inglés *password*.

<sup>11</sup>Ver capítulo 1.

<sup>12</sup>Este perfil ya viene definido por Oracle y se establece en el momento de la instalación del producto.

`session_per_user` limita al usuario a un número de sesiones en curso.

`cpu_per_session` limita el tiempo de CPU por sesión.

`connect_time` limita el tiempo total transcurrido de una sesión.

`unlimit` indica que el usuario asignado a este perfil puede usar una cantidad ilimitada de estos recursos.

`default` omite el límite para ese recurso en ese perfil.

`private_sga` limita la cantidad de espacio privado que una sesión puede asignar en el espacio compartido del *System Global Area* a un número de *bytes*.

### 8.5.3. Creación de una función

Una función es un objeto de la base de datos que contiene un conjunto de privilegios del sistema y privilegios de acceso a los datos sobre objetos de la base de datos. En vez de asignar laboriosamente los privilegios específicos a cada usuario, Oracle permite definir funciones que recogen los distintos privilegios que necesita una clase de usuarios, y entonces asignar la función (en vez de privilegios individuales) a los usuarios. También se pueden combinar funciones dentro de otras funciones. Esto facilita la gestión de la seguridad en la base de datos.

Como ya hemos visto anteriormente, Oracle ofrece tres funciones, papeles o `role` compatibles con las versiones anteriores, que son: `connect`, `resource` y `dba`.

#### Formato:

```
CREATE ROLE nom_función
[NOT IDENTIFIED |
IDENTIFIED {BY clave | EXTERNALLY [...] }];
```

*nom\_función* es el nombre que le vamos a dar a la función.

`not identified` indica que el usuario al que se le va a asignar no necesita contraseña para habilitar la función.

`identified by` indica que el usuario debe especificar una contraseña para habilitar la función.

`externally` se tomarán los datos del *SO*.

Con esta orden podemos crear una función y asignarle un nombre y conjunto de privilegios. Cuando se le asigna la función a un usuario, se le conceden todos los privilegios de la función.

Primero se crea la función con `create role`, y luego se le asignan privilegios con la orden `grant`. Si queremos concederle a un usuario un función, se la tenemos que dar también con la orden `grant`.

### 8.5.3.1. Activación o desactivación de funciones

Los usuarios pueden tener activada o no una función durante una sesión de trabajo. Esto se utiliza como medida de precaución o seguridad frente al mal uso de los privilegios del sistema o de los privilegios sobre objetos.

La habilitación o deshabilitación de una función se realiza con la orden `set`.

#### Formato:

```
SET ROLE {nom_función [IDENTIFIED BY clave]
          [,nom_función [IDENTIFIED BY clave]] ...|
        ALL [EXCEPT nom_función [,nom_función] ...] |
        NONE};
```

`identified by` pide que se introduzca la contraseña de la función para estar habilitada.

`all` indica todas las funciones excepto las indicadas en la cláusula `except`.

`none` deshabilita todas las funciones en la sesión en curso.

## 8.6. La orden alter

Esta orden del lenguaje de definición de datos se usa, como ya hemos visto, para cambiar o alterar las definiciones de los objetos de la base de datos.

### 8.6.1. Modificación de las características de un usuario

Podemos cambiar o modificar las características de nosotros mismos o de otro usuario, si tenemos privilegios para ello.

#### Formato:

```
ALTER USER nom_usuario
[IDENTIFIED {BY contraseña|EXTERNALLY}]
[DEFAULT TABLESPACE nom_tablespace]
[TEMPORARY TABLESPACE nom_tablespace]
[QUOTA {entero [K | M] | UNLIMITED}
```

```

ON nom_tablespace
[,QUOTA {entero [K | M] | UNLIMITED}
ON nom_tablespace]]
[PROFILE nom_perfil]
[DEFAULT ROLE {nom_función [,nom_función] |
  ALL [EXCEPT nom_función [,nom_función]] |
  none}]];

```

`identified by` se utiliza para cambiar una contraseña de usuario.

`default tablespace` cambia el `tablespace` por defecto del usuario.

`temporary tablespace` nos permite cambiar el espacio lógico temporal del usuario.

`quota` cambia la cuota en un `tablespace`, siendo el espacio de disco máximo que puede usar para almacenar sus objetos.

`unlimited` no establece límite de cuota en el `tablespace` indicado.

`profile` cambia el perfil asignado por el nuevo perfil de recursos.

`default role` establece las funciones por defecto a los usuarios. La cláusula `all` indica todas excepto las indicadas en la cláusula `except` y la cláusula `none` indica que no tendrá ninguna de las funciones por defecto asignadas a los usuarios.

---

#### Ejemplo 8.1

Cambio de clave

```

SQL> alter user gadeschi
  2 identified by esgadi;

```

Usuario modificado.

---

El ejemplo 8.1 nos muestra cómo podemos cambiar nuestra clave actual por otra. Tenemos que tener precaución al dar esta orden ya que la nueva clave es visible por pantalla.

### 8.6.2. Modificación de parámetros

Los usuarios pueden modificar algunos de los parámetros que afectan a su sesión en uso. Estos parámetros hacen referencia al formato por defecto de las fechas y al idioma que Oracle va a utilizar para devolver los mensajes de error.

Para poder utilizar esta orden, necesitamos tener el privilegio del sistema `alter session`.

**Formato:**

```
ALTER SESSION {SET [NLS_LANGUAGE = idioma |
  NLS_TERRITORY = territorio |
  NLS_DATE_FORMAT = formato |
  NLS_DATE_LANGUAGE = idioma [... ]} ...;
```

`nls_language` nos permite indicar el idioma para los errores y otros mensaje de Oracle. También controla el idioma para los nombres de los meses y de los días de la semana.

`nls_territory` permite fijar implícitamente el formato de fecha, carácter decimal, separador de grupos, símbolos monetarios locales, símbolos actuales de *ISO*, etc.

`nls_date_format` fija el formato de fecha por omisión. Toma como formato cualquiera de los formatos de fecha.

`nls_date_language` nos permite cambiar los nombres de los días y los meses y sus abreviaturas.

---

**Ejemplo 8.2** Cambio de idioma en formatos de fecha y mensajes de error

---

```
SQL> alter session set nls_territory = spain;
```

Sesion modificada.

```
SQL> alter session set nls_language = spanish;
```

Sesion modificada.

```
SQL> alter session set nls_date_language = spanish;
```

Sesion modificada.

---

Como podemos observar en el ejemplo 8.2, hemos cambiado el idioma y durante la sesión en curso, los mensajes aparecerán en español. Para hacer permanente estos cambio debemos incluir estas órdenes en nuestro fichero de arranque `login.sql`<sup>13</sup>.

---

<sup>13</sup>Ver capítulo 1.

**Ejemplo 8.3**

Cambio en el formato de las fechas

---

```
SQL> select sysdate
      2 from dual;
```

```
SYSDATE
```

```
-----
07/07/05
```

```
1 fila seleccionada.
```

```
SQL> alter session set nls_date_format = 'dd-mm-yy';
```

```
Sesion modificada.
```

```
SQL> select sysdate
      2 from dual;
```

```
SYSDATE
```

```
-----
07-07-05
```

```
1 fila seleccionada.
```

---

En ejemplo 8.3 el formato por defecto utiliza una barra inclinada (/) para separar los dígitos, nosotros lo hemos cambiado por el guión (-)<sup>14</sup>. Este formato permanecerá mientras no cerremos la sesión en curso. Para hacerlo permanente debemos incluirlo en nuestro fichero de arranque `login.sql`<sup>15</sup>.

### 8.6.3. Modificación de perfiles

Los perfiles pueden ser modificados usando la orden `alter`.

#### Formato:

```
ALTER PROFILE nom_perfil
LIMIT {[SESSION_PER_USER|CPU_PER_SESSION|CONNECT_TIME]
[entero | UNLIMIT | DEFAULT] |
[PRIVATE_SGA [entero [K|M] | UNLIMIT | DEFAULT] ]};
```

El significado de los argumentos son los mismos que vimos en la orden `create profile`.

---

<sup>14</sup>Ver formatos de fechas.

<sup>15</sup>Ver capítulo 1.

### 8.6.4. Modificación de funciones

Se puede modificar una función para cambiar la autorización necesaria para habilitarla.

**Formato:**

```
ALTER ROLE nom_función
  {NOT IDENTIFIED | IDENTIFIED {BY clave
  | EXTERNALLY | ...}};
```

## 8.7. La orden drop

Esta orden del lenguaje de definición de datos se usa, como ya hemos visto, para borrar los objetos de la base de datos.

### 8.7.1. Eliminación de un usuario

Un usuario puede eliminarse de la base de datos al igual que cualquier otro objeto.

Cuando eliminamos un usuario de la base de datos, éste no puede acceder a la misma ya que se eliminan todos los privilegios del sistema y los recursos asignados. Oracle, antes de eliminar al usuarios, previamente borra todos sus objetos.

**Formato:**

```
DROP USER usuario [CASCADE];
```

`cascade` elimina todos los objetos pertenecientes al usuario, incluyendo las restricciones de integridad referencial. Si no se utiliza esta opción y el usuario aún posee objetos, Oracle no eliminará al usuario.

### 8.7.2. Eliminación de perfiles

Los perfiles pueden eliminarse como cualquier otro objeto de la base de datos.

**Formato:**

```
DROP PERFIL nom_perfil [cascade];
```

`cascade` se especifica esta cláusula para desasignar el perfil *nom\_perfil* a cualquier usuario que lo tenga asignado. Oracle automáticamente asigna el perfil por defecto (`default`)

a cada uno de ellos. Es obligatoria cuando queremos borrar perfiles que están actualmente asignados a algún usuario.

### 8.7.3. Eliminación de funciones

Las funciones también se pueden eliminar. Cuando se elimina una función, Oracle se la revoca a todos los usuarios y funciones que la tienen concedida. Los usuarios que estén en el sistema cuando se elimina una función, si la tienen concedida, la mantendrá mientras dure dicha sesión.

**Formato:**

```
DROP ROLE nom_función;
```

## 8.8. Arranque y parada de una base

Existen distintas maneras de arrancar y parar una base. Aquí vamos a ver las distintas opciones para hacerlo.

### 8.8.1. Estado de una base

La base de datos puede estar activa o no. Cuando no está activa, constituye un conjunto de ficheros sobre el disco no accesibles a los usuarios. Se dice que la base está parada.

Cuando la base está activa puede encontrarse bajo diferentes formas y es posible, por tanto, que tenga diferentes estados.

Una base activa está asociada a una *instance*, encontrándose ésta materializada por unos procesos y una zona de memoria llamada *SGA*, tal como se vió en el capítulo anterior.

La base parada, se activa a continuación de una orden de arranque `startup`. Este arranque puede ser lanzado explícitamente por el administrador o implícitamente a continuación de la orden de creación de la base de datos.

Una base se desactiva cuando el administrador da la orden `shutdown`.

### 8.8.2. Arranque de una base

Una base activa puede estar en varios estados. La sintaxis es la siguiente:

**Formato:**

```
STARTUP [RESTRICT] [NOMOUNT | MOUNT ... | OPEN ...]
```

`restrict` sólo permite la entrada al sistema a usuarios que tengan el privilegio del sistema `restricted session`.

`nomount` no está montada la base de datos pero se está ejecutando la *instance*. La base de datos se arranca en este estado cuando se va a crear una nueva base de datos usando la orden `create database`.

`mount` está montada la base de datos pero no abierta. Este estado es necesario para efectuar las operaciones de mantenimiento de la base, como la restauración, la adición y supresión de ficheros de recuperación, el renombrado de ficheros, etc. Este estado no permite entrar en las tablas del diccionario de datos ni en las de los usuarios.

`open` está montada la base de datos y abierta. Este es el estado de la base cuando está en producción. Los usuarios se conectan a una base abierta con un nombre de usuario y una contraseña.

En el momento del arranque, es necesario precisar el nombre del fichero conteniendo los parámetros de inicialización. Por defecto este es el fichero *init.ora* y se encuentra en un directorio de la cuenta del sistema operativo `oracle`.

### 8.8.3. Parada de una base

Para cerrar una base de datos se utiliza la orden `shutdown`. El administrador tiene distintas maneras de parar una base de datos.

La sintaxis es la siguiente:

**Formato:**

```
SHUTDOWN [ABORT | IMMEDIATE | NORMAL]
```

`abort` para bruscamente la base de datos. Oracle desconecta inmediatamente a los usuarios sin deshacer las transacciones en curso y para la base en un estado probablemente incoherente. La coherencia será restablecida en el próximo arranque de la base. Este es el modo más rápido de «echar abajo» la base de datos.

`immediate` para inmediatamente la base de datos. Oracle deshace las transacciones en curso, desconecta a los usuarios y para la base de datos en un estado coherente.

`normal` para la base de datos de forma normal. Oracle espera a que todos los usuarios se desconecten para parar los procesos de la *instance* y liberar la *SGA*. Durante la espera ningún otro usuario podrá conectarse.

Las órdenes `startup` y `shutdown` sólo puede darse desde una cuenta especial del sistema operativo que tenga privilegios de *DBA*. Dicho de otra manera, un usuario que tenga la función *dba* no puede parar y arrancar una base si no lo hace a partir de una cuenta del sistema operativo que tiene este privilegio.

#### 8.8.4. Parámetros de inicialización

Oracle es un producto muy parametrizable. Esta propiedad permite adaptar el entorno operativo de una base de datos según el *hardware*, el tipo de aplicación, el nivel deseado de seguridad y de rendimiento, el número de usuarios, etc. ...

Esta adaptación se hace seleccionando los valores de unos parámetros contenidos en un fichero editable, que es el fichero de inicialización y que generalmente se denomina *init.ora*. Dicho fichero se carga durante el arranque de la base de datos. Una misma base puede, por tanto, ser explotada de modo diferente según los valores de los parámetros de arranque.

Estos valores caracterizan una *instance* de una base de datos y no pueden ser modificados en el curso operativo de la *instance*. Para modificarlos, hace falta parar la base de datos, hacer la modificación en el fichero *init.ora* y volverla a arrancar. Oracle calcula automáticamente los valores de ciertos parámetros en función de otros.

Los parámetros modificables tiene una influencia más o menos significativa sobre el funcionamiento de la base. Todos tienen un valor por defecto y pueden clasificarse por categorías.

El contenido del fichero *initORACLE\_SID.ora* de la base de datos *SALES* (*initsales.ora*<sup>16</sup> se muestra en los ejemplos 8.4 y 8.5).

### 8.9. Gestión del espacio en disco

Desde su creación, una base de datos Oracle posee un *tablespace* llamado *system* que contiene inicialmente el diccionario de datos y el segmento de *Rollback system*. Este *tablespace* tiene asignado al menos un fichero base.

Para garantizar el buen funcionamiento de la base es necesario crear como mínimo otro *tablespace*, orientar los datos de los usuarios hacia estos *tablespace* y reservar el *tablespace system* para el diccionario de datos y el segmento de *Rollback system*.

---

<sup>16</sup>Fichero de arranque de Oracle7.

**Ejemplo 8.4**

Fichero initsales.ora

---

```
# include database configuration parameters
# fichero de configuración
ifile = /usr/ora/dbs/configSALES.ora

# nombre de los segmentos de rollback excepto el que corresponde
# al system
rollback_segments = (r01,r02,r03,r04)

# Número de ficheros abiertos
db_files = 20

# El máximo número de bloques leídos en una operación de E/S
# durante una búsqueda secuencial
db_file_multiblock_read_count = 8

# El número máximo de bloques de datos en el buffer
db_block_buffers = 200

# Tamaño en bytes del shared pool
shared_pool_size = 3500000

# Número de bloques del fichero redo log
# necesarios para disparar un punto de recuperación
log_checkpoint_interval = 10000

# Número máximo de usuarios simultáneos del S.O.
# que pueden estar conectados a ORACLE
processes = 50

# Número máximo de bloqueos de DML, uno por cada tabla
# modificada en una transacción
dml_locks = 100

# Número de bytes asignados al buffer de redo log en
# la SGA
log_buffer = 8192

# Número de secuencias que puede ser reservada en la SGA para
# accesos inmediatos
sequence_cache_entries = 10

# Número de cubetas que se usa para acelerar la consulta
# de secuencias de nueva petición en el caché
sequence_cache_hash_buckets = 10

# Activación de la auditoria
audit_trail = true

# Tamaño máximo de bloques en el S.O. de los ficheros trace
# escritos
max_dump_file_size = 10240
```

---

**Ejemplo 8.5**

Fichero initsales.ora (cont.)

```
# Indicación del idioma de la base de datos
NLS_LANGUAGE = spanish
NLS_TERRITORY = spain

# Formatos de fecha por defecto
NLS_DATE_FORMAT = dd-mm-yy

# Activación para reconocimiento de los datos del
# del usuario del S.O.
OS_AUTHENT_PREFIX=ops$
```

### 8.9.1. Número y contenido de los tablespace

El número de los `tablespace` se determina por un criterio organizativo y un criterio de rendimiento.

El criterio de organización permite:

- Facilitar las operaciones de mantenimiento: sabiendo que la unidad más pequeña de recuperación es un `tablespace`, el administrador puede crear varios `tablespace`. Cada uno de estos contendrá las tablas que necesiten ser salvadas con la misma frecuencia.
- Poder generar varias base de datos semánticamente diferentes en la misma base física.

El criterio de rendimiento reparte en distintos `tablespace` por ejemplo los datos de los índices, en definitiva se intenta facilitar las operaciones de E/S por lo que la ubicación de los `tablespace` en distintos discos tiene un componente esencial. Puesto que es muy probable no disponer de tantos discos como posibles `tablespace` se precisen, se deben repartir dichos `tablespace` entre los discos disponibles con el mayor rendimiento posible.

### 8.9.2. Creación, modificación y supresión de un tablespace

La orden del lenguaje de definición de datos `create` permite crear un nuevo `tablespace` con sus propios parámetros de almacenamiento.

**Formato:**

```
CREATE TABLESPACE nom_tablespace
DATAFILE especific_fichero [SIZE entero[K|M]]
[,especific_fichero ...] [autoextend_clausula]
[ONLINE | OFFLINE];
```

*nom\_tablespace* nombre del *tablespace* que vamos a crear.

*datafile* especifica los datos del fichero o ficheros que componen el nuevo *tablespace*.

*especif\_fichero* es el nombre completo de un fichero. Este nombre dependerá del *S.O.* que estemos usando.

*size* especifica el tamaño del fichero en *kilobytes* o en *megabytes*; si no se especifica el tamaño, este vendrá dado en *bytes*.

*autoextend\_clausula* se utiliza para activar o desactivar la ampliación automática de un nuevo fichero de datos o fichero temporal. Si no se especifica esta cláusula, estos ficheros no se ampliarán automáticamente.

*online|offline* indica si el *tablespace* está disponibles después de su creación o no. La opción por defecto es *online*.

Se puede, en cualquier momento, añadir o renombrar un fichero del *tablespace*, y modificar sus parámetros de almacenamiento. Utilizamos la orden del *DDL alter*.

#### Formato:

```
ALTER TABLESPACE nom_tablespace
{[ADD DATAFILE especific_fichero [SIZE entero[K|M]]
[, especific_fichero ...] |
RENAME DATAFILE nom_fichero_viejo [,nom_fichero_viejo...]]
TO nom_fichero_nuevo [,nom_fichero_nuevo...]] |
[ONLINE | OFFLINE] | ...}};
```

*add datafile* añade al espacio de tabla un fichero o una serie de ficheros descritos de acuerdo con una definición de fichero, que especifica los nombres y tamaños de los ficheros de base de datos.

*rename datafile* cambia el nombre de uno o más ficheros de un *tablespace*.

Si precisamos suprimir un fichero base, es necesario primero suprimir su *tablespace* y después suprimirlo físicamente por una orden del sistema operativo, así no creamos incoherencias en la base de datos.

Para suprimir un *tablespace* no hace falta primero suprimir los objetos que contiene.

#### Formato:

```
DROP TABLESPACE nom_tablespace [INCLUDING CONTENTS
[AND DATAFILES] [CASCADE CONSTRAINTS]];
```

`including contents` borra todo el contenido del `tablespace`. Se debe especificar esta cláusula pues si el `tablespace` no está vacío, Oracle dará mensajes de error diciendo que no puede borrar un `tablespace` que contiene algún objeto.

### 8.9.3. Creación de un tablespace temporal

Cuando necesitamos crear un espacio de tablas temporal utilizaremos también la orden `create`. Un espacio de tablas temporal es una reserva de espacio en la base de datos que puede contener objetos de esquema durante una sesión. Si posteriormente se asigna este espacio de tablas temporal a un determinado usuario, Oracle también lo utilizará para operaciones de ordenación en transacciones iniciadas por ese usuario.

#### Formato:

```
CREATE TEMPORARY TABLESPACE nom_tablespace
TEMPFILE especific_fichero [SIZE entero[K|M]]
[,especific_fichero ...];
```

### 8.9.4. Gestión en disco del espacio de los usuarios

Cuando el administrador de la base de datos crea un usuario puede:

- Dejarle la libertad de crear sus propios objetos en todos los `tablespace` sin cuota de disco.
- Atribuirle un `tablespace` particular sin cuota de disco.
- Atribuirle un `tablespace` particular con cuota de disco.
- Atribuirle un `tablespace` por defecto, de trabajo. Por tanto, cada vez que un usuario cree un objeto sin precisar el `tablespace`, este objeto se creará en el `tablespace` por defecto.
- Eventualmente atribuirle un `tablespace` que va a contener sus segmentos temporales.

## 8.10. Exportación e importación de datos

Oracle incorpora varias opciones de copia de seguridad y de recuperación, entre las que podemos destacar dos utilidades `export` e `import` que se utilizan conjuntamente para exportar e importar datos. Además de poder realizar copias de seguridad, podemos reorganizar una base de datos y/o hacerla migrar de una versión de Oracle a otra.

Las dos utilidades forman parte de la mayoría de los planes de copias de seguridad y de recuperación, tanto para bases de datos pequeñas como para bases de datos de desarrollo.

### 8.10.1. *Export*

Esta utilidad extrae datos de la base de datos y almacena el resultado en un fichero binario. Puede especificarse qué partes de la base de datos deben exportarse. Se puede exportar la base de datos entera, el esquema o esquemas de un usuario o de un conjunto de usuarios o un conjunto específico de tablas.

La utilidad se invoca de forma interactiva desde el *SO* y podemos indicar los parámetros o dejar que Oracle nos indique cada uno de ellos mostrándonos el valor por defecto de cada uno. En la tabla 8.4 se muestran algunos de los parámetros.

**Formato:**

```
% exp [parámetros]
```

Parámetro	Función	Defecto
<code>compress</code>	Indica si la exportación debe comprimir segmentos fragmentados en una misma extensión. Esto afecta a las cláusulas <code>storage</code> que se almacenarán en el fichero de exportación para dichos objetos.	Y
<code>file</code>	Nombre del fichero de volcado de la exportación.	<code>expdat.dmp</code>
<code>full</code>	Exporta toda la base de datos.	N
<code>grants</code>	Indica si se exportarán las concesiones sobre los objetos de la base de datos.	Y
<code>indexes</code>	Exporta los índices de las tablas.	Y
<code>owner</code>	Lista de los usuarios a los que se les exporta datos.	Indefinido
<code>tables</code>	Nombre de las tablas que se van a exportar.	Indefinido
<code>userid</code>	Nombre de usuario/contraseña que realiza la exportación.	Ninguno

Tabla 8.4: Parámetros de la utilidad *export*

Esta utilidad tiene cuatro niveles de funcionalidad: modo completo, modo de espacio de tablas, modo usuario y modo tabla.

En el modo completo, se exporta la base de datos completa. Se lee el diccionario de datos entero y las instrucciones DDL necesarias para crear de nuevo la base de datos completa y se escriben en el fichero de volcado de exportación. Este fichero incluya órdenes de creación para todos los espacios de tablas (`tablespace`), todos los usuarios y todos los objetos, datos y privilegios de sus esquemas.

En el modo espacio de tablas, se exportarán todos los objetos contenidos en los espacios de tablas especificados, incluyendo la definición de índices de los objetos contenidos, aunque se encuentre en otro espacio de tablas.

En el modo usuario, se exportan los objetos de un usuario, además de los datos asociados. Asimismo, se exportan todas las concesiones y los índices creados por el usuario a sus objetos. Sin embargo, las concesiones y los índices creados por usuarios distintos del propietario no se exportan.

En el modo tabla se exporta una tabla especificada. La estructura de la tabla, sus índices y sus concesiones se exportan con o sin sus datos. El modo tabla puede también exportar el conjunto completo de tablas poseído por un usuario (especificando el propietario del esquema pero ningún nombre de tabla).

A modo de resumen:

- El administrador de la base de datos puede exportar:
  - Toda la base de datos.
  - Solamente las modificaciones hechas en la base después del último *export*<sup>17</sup>.
  - Sus propios objetos.
  - Los objetos de otros usuarios.
- Un usuario puede exportar:
  - Todos sus objetos.
  - Solamente algunos de sus objetos.

### 8.10.2. *Import*

La utilidad *export* realizan una lectura lógica de la base de datos. Para leer la información contenida en el fichero binario de volcado creado por la operación de exportación, hay que emplear la utilidad *import* (importar). Con *import* se pueden elegir de manera selectiva los objetos o usuarios del fichero de volcado que se deseen importar. Después, *import* tratará de insertar estos datos en la base de datos (en lugar de sobrescribir los registros existentes).

Al igual que la utilidad *export*, la importación se realiza desde el indicativo del *SO*.

**Formato:**

```
% imp [parámetros]
```

La utilidad *import* se usa para:

---

<sup>17</sup>También llamado incremental.

- Restaurar una base de datos a continuación de su pérdida total o parcial.
- Reorganizar una base de datos total o parcialmente.
- Pasar una base de datos de una versión de Oracle a otra.

En la tabla 8.5 se muestran algunos de los parámetros para esta utilidad.

Parámetro	Función	Defecto
<code>file</code>	Nombre del fichero de volcado de exportación que se va a importar.	<code>expdat.dmp</code>
<code>fromuser</code>	Listas de cuentas de la base de datos cuyos objetos deben leerse desde el fichero de volcado de exportación.	Indefinido
<code>full</code>	Importa el fichero de volcado de exportación completo.	N
<code>grants</code>	Importa los privilegio y funciones.	Y
<code>indexes</code>	Indica si importa los índices de las tablas.	Y
<code>tables</code>	Lista de tablas que se importan.	Ninguno
<code>touser</code>	Lista de cuentas de la base de datos en la que se importarán los objetos del fichero de volcado de exportación. No es necesario asignar el mismo valor a <code>fromuser</code> y <code>touser</code> .	Ninguno
<code>userid</code>	Nombre de usuario/contraseña de la cuenta que ejecuta la importación.	Indefinido

Tabla 8.5: Parámetros de la utilidad *import*

En los apéndices C y D se muestran ejemplos de cómo utilizar dichas herramientas por parte de los usuarios de forma interactiva.