

Diseño físico relacional

Contenidos

- » Herramientas gráficas y de texto proporcionadas por los SGBD
- » El lenguaje de definición de datos
- » Creación, modificación y eliminación de BBDD
- » Creación, modificación y eliminación de tablas
- » Implementación de restricciones

Objetivos

- » Definir las estructuras físicas de almacenamiento
- » Crear tablas
- » Seleccionar tipos de datos adecuados
- » Definir campos claves en las tablas
- » Implantar las restricciones establecidas en el diseño lógico
- » Verificación mediante conjuntos de pruebas
- » Uso de asistentes y herramientas gráficas
- » Uso del lenguaje de definición de datos (DDL)
- » Definir y documentar el diccionario de datos

En este tema se detalla el proceso de implantación definitiva, o diseño físico, de la base de datos en un sistema informático. Se describe el uso del lenguaje SQL distinguiendo las peculiaridades de los principales SGBD.

3.1. Notación para la sintaxis

En informática, cuando se quiere utilizar cualquier tipo de lenguaje de programación, se necesita una sintaxis para definir cómo construir sentencias en ese lenguaje de programación. Para expresar la sintaxis se utiliza una *notación*. Esta notación está compuesta por componentes léxicos o *tokens*. Estos tokens pueden ser palabras clave del lenguaje, definiciones de otros elementos sintácticos más básicos, expresiones, variables, etc. La notación utilizada en este libro es la siguiente:

- Palabras en mayúsculas. Estas son las palabras reservadas del lenguaje. Por ejemplo SELECT, DROP, CREATE son palabras reservadas, esto quiere decir que no pueden utilizarse para nombrar objetos de la base de datos porque tienen una misión específica.
- Palabras en minúscula. Se utiliza para realizar descripciones de sintaxis más en detalle. Por ejemplo, el token *especificacion_de_filtro* se puede desplegar en más definiciones para realizar filtros en las consultas.
- Corchetes. Un elemento sintáctico entre corchetes indica opcionalidad. Es decir, lo que está encerrado entre corchetes se puede incorporar a la sentencia o no, dependiendo de lo que el programador quiera expresar. Por ejemplo, en la definición *CREATE [TEMPORARY] TABLE*, se puede indicar de forma opcional el token TEMPORARY para crear una tabla temporal, que solo durará en memoria mientras el usuario permanezca conectado. Si varios elementos van separados mediante el token pipe "|", se puede elegir uno de ellos.
- Llaves. Indica alternativa obligatoria. Se debe elegir entre los elementos separados mediante el token pipe "|". Por ejemplo, en la definición de sintaxis para crear una base de datos, *CREATE {DATABASE | SCHEMA} nombre_bd*, hay que escribir uno de los dos token entre llaves. Se puede optar bien por CREATE DATABASE nombre_bd o por CREATE SCHEMA nombre_bd.
- Puntos suspensivos. Significa repetición, es decir, el último elemento sintáctico puede repetirse varias veces. Por ejemplo, para codificar una consulta se usa la definición *SELECT columna [,columna] ... FROM tabla*. Los puntos suspensivos significan que se puede repetir el token *[,columna]* tantas veces como se deseé. Así, es posible escribir *SELECT Nombre, Direccion, Codigo FROM Clientes*.

3.2. Herramientas gráficas proporcionadas por los SGBD

Es muy sencillo manipular una base de datos compleja si se dispone de un interfaz gráfica de usuario que ayude al DBA, Database Administrator, a enviar comandos de administración de forma automática y sin necesidad de conocer su sintaxis:

3.2.1. PhpMyAdmin de MySQL

MySQL dispone de un interfaz basada en páginas web llamada *PhpMyAdmin*, que a través de un servidor web, por ejemplo Apache, permite administrar las bases de datos de un servidor desde cualquier equipo de la red.

The screenshot shows the PhpMyAdmin interface for the 'jardineria' database. The left sidebar lists the database structure with 8 tables: Clientes, DetallePedidos, Empleados, GamasProductos, Oficinas, Pagos, Pedidos, and Productos. The main area displays a grid of table information with columns: Tabla, Acción, Registros¹, Tipo, Cotejamiento, Tamaño, and Residuo a depurar. Below the grid, there are buttons for 'Marcar todos/as / Desmarcar todos' and 'Para los elementos que están marcados'. At the bottom, there are links for 'Vista de impresión' and 'Diccionario de datos', and a form for 'Crear nueva tabla en la base de datos jardineria' with fields for 'Nombre:' and 'Número de campos:'.

Tabla	Acción	Registros ¹	Tipo	Cotejamiento	Tamaño	Residuo a depurar
Clientes	X	36	InnoDB	latin1_swedish_ci	16.0 KB	
DetallePedidos	X	295	InnoDB	latin1_swedish_ci	16.0 KB	
Empleados	X	31	InnoDB	latin1_swedish_ci	16.0 KB	
GamasProductos	X	1	InnoDB	latin1_swedish_ci	16.0 KB	
Oficinas	X	9	InnoDB	latin1_swedish_ci	16.0 KB	
Pagos	X	26	InnoDB	latin1_swedish_ci	16.0 KB	
Pedidos	X	115	InnoDB	latin1_swedish_ci	16.0 KB	
Productos	X	276	InnoDB	latin1_swedish_ci	128.0 KB	
8 tabla(s)	Número de filas	789	MyISAM	latin1_swedish_ci	240.0 KB	8 Byt

Figura 3.1: Interfaz web de PhpMyAdmin.

Este software dispone de opciones para realizar prácticamente cualquier opción que se pueda realizar vía SQL. Permite gestionar las bases de datos de un servidor, crear, borrar y modificar tablas, lanzar comandos SQL, exportar e importar información, recopilar estadísticas, hacer copias de seguridad, etc. Además, dispone de un pequeño diseñador, tipo *MySQL Workbench* que permite gestionar las relaciones de las tablas.

3.2.2. Oracle Enterprise Manager y Grid Control

El SGBD Oracle dispone de dos herramientas gráficas, ambas con interfaz web y montadas sobre un servidor web propietario de Oracle.

Enterprise Manager. Esta herramienta está incorporada directamente en el software de Oracle, y es configurada por el asistente de creación de base de datos. Es capaz de manipular todas las funciones básicas de una base de datos (creación de tablas, usuarios, exportación e importación de información, etc.)

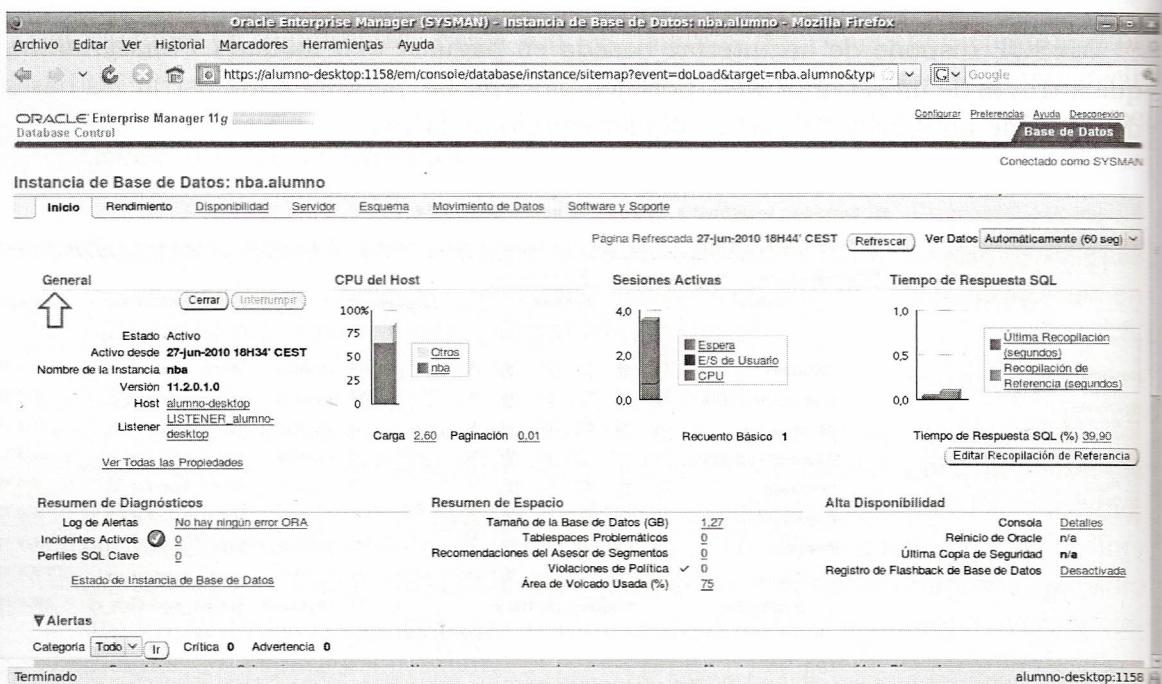


Figura 3.2: Enterprise Manager de Oracle.

◊ **Actividad 3.1:** A través de una máquina virtual con un sistema operativo de tipo Linux, por ejemplo, Ubuntu, instala los paquetes de mysql server y los paquetes de phpmyadmin. Para ello, escribe en un terminal el comando **sudo apt-get install mysql-server** y después, el comando **sudo apt-get install phpmyadmin**. Es posible que antes de realizar esta operación necesites actualizar el repositorio de paquetes mediante el comando **sudo apt-get update**. Selecciona que la instalación configure phpMyAdmin automáticamente para apache y escribe una contraseña de usuario. Cuando termine la instalación, ya puedes abrir un navegador en la máquina virtual y escribir la dirección

`http://127.0.0.1/phpmyadmin/index.php`. Introduce el usuario **phpmyadmin** y la contraseña que hayas escrito en la configuración. Finalmente, explora las opciones que te da la interfaz de usuario. Si lo prefieres, puedes descargar del blog del libro la máquina virtual de prácticas, donde ya está instalado mysql-server y phpmyadmin. Crea una base de datos que contenga dos tablas de la temática que elijas.

◊ **Actividad 3.2:** Arranca el Enterprise Manager de la máquina virtual de prácticas (Linux) y accede al servidor web para consultar cada una de las pestañas de las que dispone y examinando la información de la base de datos que te proporciona. Abre un terminal y con el usuario **oracle**, utiliza el comando **emctl start dbconsole** para arrancar el enterprise manager. A continuación, escribe en un navegador de internet la dirección `http://localhost:1158/em` o directamente pulsa dos veces en el enlace que hay en el escritorio. En el cuadro de usuario y password introduce **SYSMAN** y **manager**.

Grid Control. Se ha de instalar aparte del software de Oracle. Permite gestionar múltiples bases de datos en diversos servidores, permitiendo consultar el estado y rendimiento de todas y cada una de ellas.

The screenshot shows the Oracle Grid Control 10g interface. At the top, there's a navigation bar with links for Inicio, Destinos, Despliegues, Alertas, Políticas, Trabajos, and Informes. Below the navigation bar, the main content area is divided into several sections:

- Visión General:** Shows a large circular progress bar at 100% and a button labeled "Activo(19)".
- Alertas de Todos los Destinos:** Displays counts for Crítico (3), Advertencia (7), and Errores (0).
- Violaciones de Política de Todos los Destinos:** Displays counts for Crítico (46), Advertencia (24), and Informativo (7).
- Trabajos de Todos los Destinos:** Shows Ejecuciones Problemáticas (últimos 7 días) with 0 checked and Ejecuciones Suspendidas (últimos 7 días) with 0 checked.
- Búsqueda de Destino:** A search bar with the placeholder "Buscar Todo".
- Violaciones de Política de Seguridad:** A table showing counts for Crítico (46), Advertencia (24), and Informativo (2) with a note "Nuevo en las Últimas 24 Horas" (1).
- Asesores de Parches Críticos para Directorios Raíz de Oracle:** Shows 0 critical patches for the root directory.
- Resumen de Despliegues:** A dropdown menu set to "Instalaciones de Base de Datos".
- Instalaciones de Base de Datos:** Lists Oracle Database 10g 10.1.0.4.0 and Oracle Database 10g 10.2.0.3.0.

Figura 3.3: Grid Control de Oracle.

Ambas herramientas son extremadamente *pesadas* en términos de consumo de recursos, por lo que en muchas ocasiones se ha de administrar la base de datos sin su ayuda.

3.2.3. DB2 Data Studio

Este software sustituirá en un futuro próximo a una herramienta más antigua llamada *Control Center*. Permite manipular los objetos de bases de datos DB2 e Informix y sus permisos. Soporta la administración avanzada de DB2 tanto en Windows como en Linux, y simplifica la construcción de consultas SQL. Su gran potencia es que facilita la creación de *Servicios Web* que distribuyen datos de consultas SQL a las aplicaciones cliente.

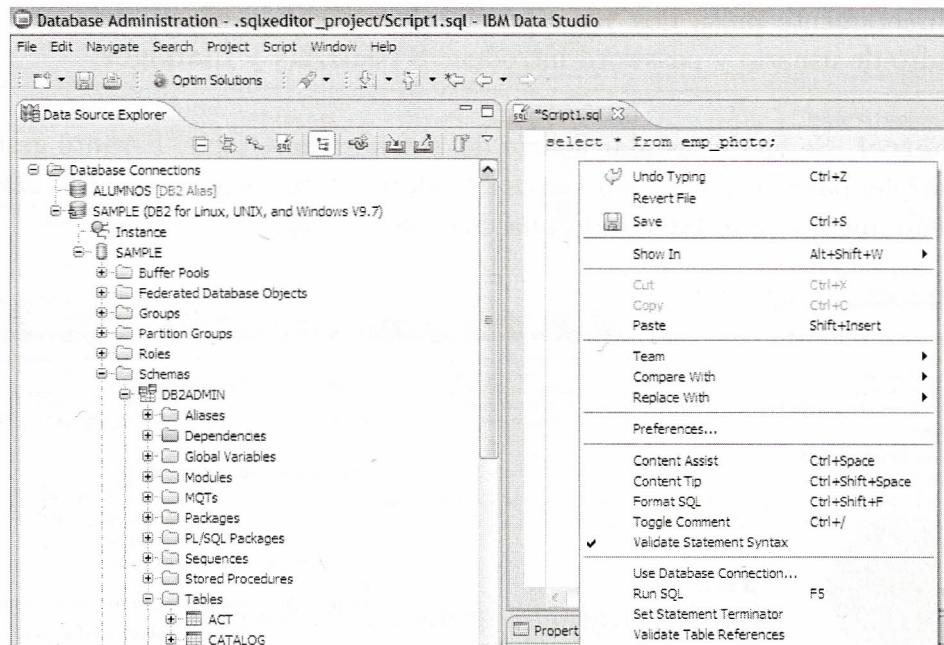


Figura 3.4: Interfaz de Data Studio para DB2.

-
- ◊ **Actividad 3.3:** De la página web <http://www-01.ibm.com/software/data/db2/> descarga el DB2 express. Deberás registrarte para poder descargar tres ficheros:

db2exc_971_WIN_x86.zip
ibm_data_studio_standalone_win.zip
db2exc_vsai_971_WIN_x86.exe

Para arrancar el asistente de la instalación, haz doble clic sobre db2exc_971_WIN_x86.zip\EXPC\image\setup.exe, pulsa sobre “Instalar un producto” y sigue los pasos del asistente para realizar una *instalación típica*. Una vez finalizada la instalación, con el *Editor de mandatos*, ejecuta el comando *list db directory* para comprobar alguno de sus parámetros.

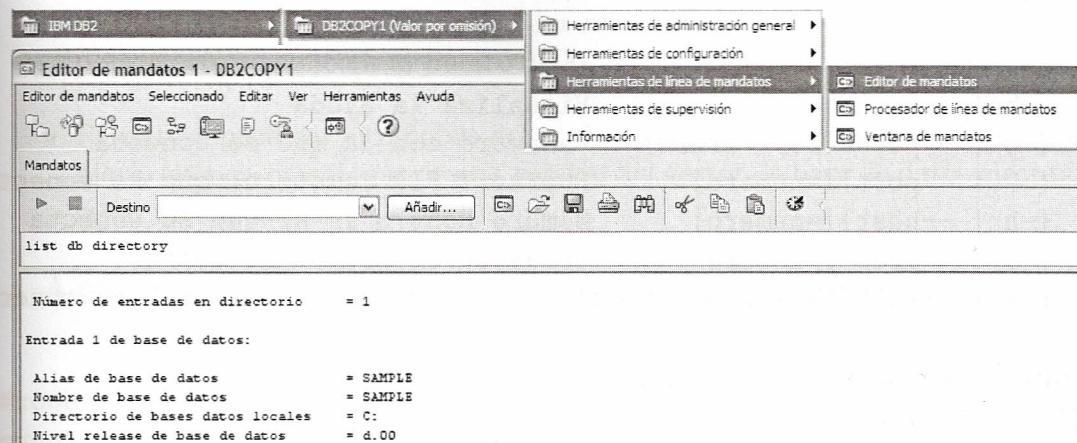


Figura 3.5: Editor de mandatos para DB2.

El consejo del buen administrador...

Muchos administradores solo conocen las herramientas gráficas de gestión y administración de una base de datos, puesto que es más cómodo y más intuitivo, y además, aprender los lenguajes de programación de bases de datos es una tarea difícil y laboriosa. Sin embargo, conocer los comandos y las instrucciones que proporciona un SGBD, otorga una visión extra que posibilita automatizar tareas rutinarias y permite solucionar problemas que no se pueden solucionar solo con las herramientas gráficas. A un administrador que conoce a la perfección todos estos comandos, le resulta muy sencillo actualizarse en los continuos cambios de versiones en estas herramientas gráficas.

3.3. Intérpretes de comandos de los SGBD

La utilidad principal de un SGBD es su intérprete de comandos. Es una aplicación cliente cuya única misión es enviar comandos al SGBD y mostrar los resultados devueltos por el SGBD en pantalla. El cliente del servidor MySQL (mysql-server) se llama *mysql*, el de Oracle se llama *sqlplus* y el de DB2 se llama *db2*. Para invo-

carlos desde el sistema operativo (Windows o Unix), tan solo hay que escribir en un terminal su nombre con ciertas opciones.

3.3.1. MySQL: El cliente de MySQL-Server

```
mysql [options] [database]
```

options:

--help	Visualiza la ayuda
{-p --password} [=frase]	Password con la que se conecta
{-P --port} [=numero]	Puerto TCPIP remoto al que se conecta
{-h --host} [=numero]	Nombre Host o IP al que se conecta
{-u --user} [=usuario]	Usuario con el que se conecta
{-s --socket} [=nombre_fich]	Fichero socket con el que se conecta

#ejemplo típico:

```
mysql -u root -p  
Enter password: *****
```

```
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 37  
Server version: 5.0.75-0ubuntu10.2 (Ubuntu)
```

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

El comando puede ir acompañado de dos tokens opcionales, *options*, que permite especificar una serie de parámetros de conexión y *database*, que especifica en qué base de datos se ejecutarán los comandos introducidos. Las opciones que aquí se muestran son solo algunas de las disponibles, para más información consultar la documentación de MySQL:

- La opción -help muestra un resumen de las opciones disponibles.
- La opción -u o -user permite especificar el usuario de conexión, por ejemplo -u paco.
- La opción -p o -password permite introducir un password para la conexión del usuario. Se puede escribir -p frase.password o solo -p para que el gestor la solicite y la oculte con asteriscos para que nadie pueda espiar.
- La opción -h o -host permite seleccionar el host donde está el gestor de base de datos. Puede ser localhost o 127.0.0.1 si la base de datos está en el ordenador

local y se desea conectar vía tcp-ip u otra dirección IP. Si se omite, por defecto se conecta al servidor por named pipes (o tuberías con nombre) que no salen a la interfaz local de la tarjeta de red y son mecanismos de comunicación más rápidos, pues son internos al ordenador.

- La opción -P o -p permite especificar el número de puerto al que conectarse (si se conecta vía TCP-IP al ordenador remoto). Si no se indica este parámetro, por defecto se conecta al puerto 3306.
- Permite especificar el nombre del socket por defecto, esto será necesario cambiarlo cuando exista más de una versión del gestor de base de datos ejecutándose en un ordenador.

A continuación se muestran unos cuantos ejemplos de conexión:

```
#conexión sin usuario y password (se conecta como anónimo y sin password)
mysql
```

```
#conexión con usuario y password (se conecta como root y su password )
mysql -u root -p
Enter password: *****
```

```
#conexión con usuario y password en claro a la base de datos jardineria
mysql -u root -pPasswordDelUsuario jardineria
```

```
#conexión con usuario y password en claro a la base de datos jardineria
# del host 192.168.3.100
mysql -u root -pPasswordDelUsuario -h 192.168.3.100 jardineria
```

```
#conexión con usuario y password en claro a la base de datos jardineria
# del host 192.168.3.100 con puerto 15300
mysql -u root -pPasswordDelUsuario -h 192.168.3.100 jardineria -P 15300
```

3.3.2. Ejecución de consultas en MySQL

Para ejecutar una consulta, tan solo es necesario arrancar el cliente mysql y conectarse a una base de datos del gestor. A continuación, escribir en la consola el comando SQL que se desea ejecutar y se obtienen los resultados. Cuando una línea es precedida del carácter #, el resto de la línea se ignora. Estas líneas se llaman *comentarios* y son útiles para documentar las acciones realizadas. Por ejemplo:

```
#selecciona la version del gestor y la fecha actual
mysql> select version(),current_date();
+-----+-----+
| version() | current_date() |
+-----+-----+
| 5.0.75-0ubuntu10.2 | 2009-8-20 |
+-----+-----+
1 row in set (0.00 sec)
```

Este comando ilustra distintas cosas:

- Un comando consiste en una sentencia SQL terminada en punto y coma.
- Cuando se escribe un comando, mysql lo manda al servidor para que lo ejecute, muestra los resultados y vuelve al *prompt*, indicando que está listo para recibir más consultas.
- Muestra los resultados en forma de tabla (filas y columnas). La primera fila contiene las etiquetas para las columnas y las filas siguientes muestran los resultados de la consulta.
- Muestra las filas devueltas y cuánto tiempo tardó en ejecutarse, lo cual puede ofrecer una idea de la eficiencia del servidor, aunque estos valores pueden ser imprecisos pues dependen de muchos factores, tales como la carga del servidor, velocidad de comunicación, etc.

Las palabras claves pueden ser escritas en mayúsculas o minúsculas, y los identificadores de tabla, campo, índice, etc son sensitivos a las mayúsculas y minúsculas en las versiones de unix (no así en Windows). Por ejemplo, las siguientes consultas son equivalentes:

```
mysql> SELECT VERSION(),CURRENT_DATE();
mysql> SElect Version(),current_Date();
mysql> select version(),current_date();
```

Es posible escribir más de una consulta por línea, siempre y cuando estén separadas por punto y coma:

```
mysql> select user();select now();
+-----+
| user() |
+-----+
| root@localhost |
+-----+
```

```
+-----+ CONSULTAS
| now() | bbdd_mesa>
+-----+
| 2009-10-20 09:53:29 | BBDD COMPLETAMENTE
+-----+
```

No es necesario escribir un comando en una sola línea, así que los comandos que requieran de varias líneas no son un problema. MySQL determinará donde finaliza la sentencia cuando encuentre el punto y coma, no cuando encuentre el fin de línea:

```
mysql> select user(),
->      current_date();
+-----+-----+
| user() | current_date() |
+-----+-----+
| root@localhost | 2009-10-20 |
+-----+
```

Si no se desea terminar de escribir la consulta, por ejemplo, por haber cometido un error en la sintaxis, se escribe para terminarla sin ejecutar:

```
mysql> select
->      user(),
->      \c
mysql>
```

El prompt se comunica con el usuario dándole información sobre qué ocurre, por ejemplo:

```
-> Espera la siguiente línea del comando
'> Espera que se cierre una comilla sencilla
"> Espera que se cierre una comilla doble
mysql> Listo para una nueva consulta
```

Otra forma de ejecutar comandos SQL es almacenarlos en un fichero de texto y mandarlo a ejecución mediante el comando *source*, que recibe como parámetro un fichero de comandos y ejecuta uno por uno todos los comandos que tenga el fichero:

```
#ejecución del script de creación crear_bbdd_startrek.sql
mysql> source /home/ivan/crear_bbdd_startrek.sql
```

Por último, también es posible ejecutar los comandos de un fichero de texto desde la shell, esto es, en modo batch, redirigiendo al cliente mysql un fichero de entrada. Esto es útil para tareas administrativas donde se ejecutan varios ficheros de comandos, además de otras tareas de mantenimiento del servidor a través de un shell script:

```
#ejecucion en modo batch
~$ mysql -u root -pPassWdUsuario <crear_bbdd_startrek.sql

#ejecucion en modo batch almacenando resultados
~$ mysql -u root -pPassWdUsuario <crear_bbdd_startrek.sql >resultado
```

◊ **Actividad 3.4:** Descarga de la página web de MySQL la última versión disponible para Windows. A continuación, instala el software. Asegúrate de recordar la clave que eliges para el usuario root. Ejecuta la interfaz de comandos de mysql. Podrás hacerlo desde un terminal de MS-DOS (símbolo del sistema) o desde el menú de inicio. Ejecuta una consulta para ver el usuario con el que te has conectado, la versión de base de datos y la hora actual. Realiza las mismas consultas en la máquina virtual Linux disponible en el blog del libro (<http://bbdd-garceta.blogspot.com>)

3.3.3. SQL*Plus: El intérprete de comandos de Oracle

El intérprete de comando de Oracle es el programa sqlplus. Se puede invocar desde el sistema operativo con la siguiente sintaxis:

```
sqlplus [{usuario[/password]}[@<identificador_conexión>] | / | /nolog]
          [AS {SYSDBA | SYSOPER}]
```

En Oracle, cada instancia está referenciada por uno o más identificadores de conexión. *identificador_conexión* es el nombre de esta referencia.

```
#conexión a la instancia jardineria con el usuario paco
sqlplus paco/password_paco@jardineria
```

La opción /nolog arranca sqlplus sin conectarse a la base de datos.

```
#arrancar sqlplus sin conexión
sqlplus /nolog
```

La opción / conecta a la instancia sin usuario. Suele usarse para tareas administrativas:

```
#conexión a la instancia $ORACLE_SID como SYSDBA
sqlplus / as SYSDBA
```

Cada usuario se puede conectar representando un rol; estos roles son el de administrador de la base de datos (SYSDBA) y operador de base de datos (SYSOPER). SYSDBA y SYSOPER tienen privilegios para conectarse a la base de datos cuando no está abierta, pudiendo de este modo, realizar tareas de administración y mantenimiento.

3.3.4. Ejecución de consultas en SQL*Plus

Al igual que en MySQL, una vez conectado sqlplus a una base de datos con un usuario, password y nombre de servicio, se procede a escribir las consultas terminadas en punto y coma, con las siguientes consideraciones:

- SQL*PLUS no es sensible a mayúsculas y minúsculas ni en las palabras reservadas, ni en los nombres de los objetos.
- Se puede escribir una consulta en múltiples líneas:

```
SQL> select *
  2  from
  3 nba.jugadores; --comentario de línea
```

- Los comentarios de línea se realizan precediendo una línea de dos caracteres -
- Existen los comentarios de bloque, es decir, que afectan a más de una línea. Comienzan con los caracteres /* y terminan con los caracteres */
- Existe una tabla llamada Dual que está disponible para todos los usuarios, y sirve para seleccionar variables del sistema o para evaluar una expresión:

```
SQL> select sysdate, user from dual;
SYSDATE   USER
```

```
27/06/10 SYS
```

```
SQL> select 5+4 from dual;
      5+4
```

```
9
```

¿Sabías que ... ? Existe un intérprete de comandos gráfico en Oracle llamado iSQL*Plus, con interfaz web. Se instala aparte del software de Oracle, descargando de la página web de Oracle (www.oracle.com) el paquete isqlplus.zip.

◊ **Actividad 3.5:** Arranca la máquina virtual Ubuntu y abre un terminal. Conéctate con el usuario oracle mediante el comando:

su oracle

Contraseña: oracle

A continuación, arranca sqlplus con el usuario nba y la password nba:

sqlplus nba/nba

Después, ejecuta las siguientes consultas:

```
SQL> select table_name from user_tables;  
SQL> select table_name from dba_tables;
```

Lee la sección 3.8.5 y comenta brevemente qué hacen las sentencias anteriores y en qué se diferencian.

3.4. El lenguaje de definición de datos

El sublenguaje de SQL que permite la definición de datos es el DDL (Data Definition Language). Las funciones de este sublenguaje son:

- Crear tablas, índices y otros objetos de la base de datos (como vistas, sinónimos, etc.)
- Definir las estructuras físicas donde se almacenarán los objetos de las bases de datos (espacios de tablas (tablespaces), ficheros de datos (datafiles), etc.)

Por ejemplo, para crear el esquema de la práctica resuelta 2.1 se puede escribir el siguiente código:

```
CREATE TABLE Actores(  
Codigo INTEGER PRIMARY KEY,  
Nombre VARCHAR(40),  
Fecha DATE,  
Nacionalidad VARCHAR(20),  
CodigoPersonaje INTEGER REFERENCES PERSONAJES(Codigo)  
);  
  
CREATE TABLE Personajes(  
CodigoPersonaje INTEGER,  
Nombre VARCHAR(30),  
Raza DATE,  
Grado VARCHAR(20),  
CodigoActor INTEGER REFERENCES Actores(Codigo),  
CodigoSuperior INTEGER REFERENCES Personajes(Codigo)  
);
```

En el código anterior, las palabras reservadas del lenguaje se han escrito en mayúsculas (CREATE, DATE, VARCHAR), y los nombres de objetos (Actores, Grado, Nombre) se han escrito en minúsculas para que el lector pueda identificar fácilmente qué parte de la sintaxis se debe respetar y cuál se puede variar.

DDL tiene 3 instrucciones básicas:

CREATE tipo_objeto Nombre Definición. Crea un objeto de un determinado tipo (DATABASE, TABLE, INDEX, etc.) con un nombre (por ejemplo Actores o Personajes) y una definición (CodigoPersonaje, Nombre, etc.).

DROP tipo_objeto Nombre. Elimina un tipo de objeto especificado mediante un nombre. Por ejemplo, la sentencia **DROP TABLE Actores**, borraría de la base de datos la tabla Actores junto con todos sus datos.

ALTER tipo_objeto Nombre Modificación. Modifica la definición de un objeto. Por ejemplo, la sentencia **ALTER TABLE Actores DROP COLUMN Fecha**, eliminaría la columna Fecha de la tabla Actores.

3.5. Creación de bases de datos

3.5.1. Creación de bases de datos en MySQL

En MySQL, para crear una base de datos se usa el comando CREATE DATABASE:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] nombre_db
      [especificación_create [, especificación_create] ...]
especificación_create:
      [DEFAULT] CHARACTER SET juego_caracteres
      | [DEFAULT] COLLATE nombre_colación
```

Por ejemplo, para crear la base de datos Startrek, en MySQL, habrá que teclear, en la consola de comandos, la instrucción¹:

```
CREATE DATABASE Startrek CHARACTER SET Latin1 COLLATE latin1_spanish_ci;
```

Nótese que el término a elegir puede ser o DATABASE o SCHEMA, en cualquier caso, el efecto es el mismo; MySQL usa algunos sinónimos para hacer más compatible su sistema de gestión con otros SGBD. Por otro lado, el parámetro opcional

¹Todas las sentencias mysql deben terminar en un carácter punto y coma (:)

(especificación_create) permite elegir el *juego de caracteres* que va a usar la base de datos (utf8, latin1, latin2, etc.), y la *colación*, que especifica cómo tratar el alfabeto del juego de caracteres, es decir, cómo se ordena (por ejemplo, la ñ después de la m y n y no conforme a la tabla de códigos ascii), y cómo se comparan los caracteres (por ejemplo, si Á es igual a á)

Para poder utilizar una base de datos, primero hay que establecer que las operaciones que se realicen a continuación se realicen sobre esta, es decir, hay que *usarla*. Para *usarla* se utiliza el comando *USE db_name*

Por ejemplo, para empezar a manipular la base de datos Startrek, habrá que ejecutar el comando *USE Startrek*;

Otro comando muy útil para ver cuántas bases de datos está controlando el gestor es el comando *SHOW DATABASES*, que obtiene un listado de las bases de datos activas en el servidor.

```
mysql> SHOW DATABASES;
+-----+
| Database      |
+-----+
| information_schema |
| Startrek       |
| mysql          |
| NBA            |
+-----+
```

Tras la ejecución de este comando se puede ver que, además de NBA y Star trek, hay otras dos que están siempre presentes, 'information_schema' y 'mysql' que almacenan metadatos (datos sobre los datos) e información de usuarios y permisos.

◊ **Actividad 3.6:** Ejecuta los siguientes comandos en la máquina virtual Ubuntu para crear una base de datos y una tabla:

```
mysql> CREATE DATABASE Veterinario;
mysql> USE Veterinario;
mysql> CREATE TABLE Mascotas (
    Nombre VARCHAR(10),
```



```
FechaNacim DATE,
Amo VARCHAR(40) );
```

A continuación, prueba a crear una tabla llamada Vacunas, con una fecha y una hora para cada mascota. Si es necesario, lee la Sección 3.8 

3.5.2. Creación de bases de datos en Oracle

El proceso de creación de una base de datos en Oracle es bastante más elaborado. En MySQL, una sola instancia controla todas las bases de datos, pero en Oracle, cada base de datos está asociada a una instancia. Una instancia es el conjunto de procesos de Oracle que están en ejecución en el sistema operativo y cuya misión es controlar todo lo referente a la gestión de la base de datos. Para crear una base de datos, primero hay que crear una instancia para proceder a invocar el comando *create database* pertinente. Oracle dispone de la utilidad gráfica *dbca* (*database configuration assistant*) que facilita este proceso. No obstante, a continuación se detalla el procedimiento para crear una base de datos Oracle en un sistema operativo Unix de forma manual:

```
#desde el sistema operativo
su oracle      #conecta con el usuario oracle
cp $ORACLE_HOME/dbs/init.ora $ORACLE_HOME/dbs/initjardineria.ora
export ORACLE_SID=jardineria #se conectará a la instancia jardineria
#editar los parámetros del fichero init que contiene los parámetros de
#inicio de la base de datos.
#Debe llamarse con el nombre initNombreInstancia.ora
gedit $ORACLE_HOME/dbs/initjardineria.ora
#cambiar los siguientes parámetros
    db_name='jardin'
    memory_target=100M
    processes = 40
    audit_file_dest='/var/oracle/product/11.2.0/'
    audit_trail = 'db'
    db_block_size=8192
    db_domain=''
    db_recovery_file_dest='/var/oracle/product/11.2.0/flash_recovery_area'
    db_recovery_file_dest_size=2G
    diagnostic_dest='/var/oracle/product/11.2.0'
    dispatchers='(PROTOCOL=TCP) (SERVICE=ORCLXDB)'
    open_cursors=40
    remote_login_passwordfile='EXCLUSIVE'
    undo_tablespace='UNDOTBS1'
sqlplus / as sysdba      #invocar a sqlplus como sysdba
```

```
--dentro de SQL PLUS, conectarse sin montar la base de datos
SQL> startup nomount
      ORACLE instance started.
      Total System Global Area  104611840 bytes
      Fixed Size                  1334828 bytes
      Variable Size              83886548 bytes
      Database Buffers           16777216 bytes
      Redo Buffers                2613248 bytes
--a continuación, se invoca el comando CREATE DATABASE
SQL> CREATE DATABASE jardin
MAXINSTANCES 1
MAXLOGFILES 16
MAXLOGMEMBERS 3
MAXDATAFILES 100
CHARACTER SET WE8ISO8859P15
DATAFILE '/var/oracle/datos/jardineria/system01.dbf'
      SIZE 300M EXTENT MANAGEMENT LOCAL
UNDO TABLESPACE UNDOTBS1 DATAFILE
      '/var/oracle/datos/jardineria/undotbs01.dbf'
      SIZE 550M SYSAUX DATAFILE
      '/var/oracle/datos/jardineria/sysaux01.dbf' SIZE 300M
DEFAULT TEMPORARY TABLESPACE TEMP
      TEMPFILE '/var/oracle/datos/jardineria/temp01.dbf' SIZE 100M
LOGFILE GROUP 1 ('/var/oracle/datos/jardineria redo01.log') SIZE 51200K,
GROUP 2 ('/var/oracle/datos/jardineria redo02.log') SIZE 51200K,
GROUP 3 ('/var/oracle/datos/jardineria redo03.log') SIZE 51200K;
>Database created.
--Finalmente, se crea el catalogo de metadatos
SQL> @/rdbms/admin/catalog.sql
SQL> @/rdbms/admin/catproc.sql
SQL> @/rdbms/admin/utlrp.sql
```

Las opciones de CREATE DATABASE para Oracle son las siguientes

- MAXINSTANCES indica el número máximo de instancias que pueden estar usando a la vez la base de datos objeto de la creación. Un valor distinto de 1 implicaría que la base de datos puede ser usada en RAC, Real Application Cluster, lo cual significa que la base de datos puede estar montada, abierta y en uso desde más de un servidor.
- MAXLOGFILES es el número máximo de grupos de ficheros de log que pueden crearse. Un fichero log registra todas las operaciones que realizan los usuarios.
- MAXDATAFILES indica el número máximo de ficheros de datos que pueden usarse.
- DATAFILE es el fichero para el tablespace SYSTEM.

- MAXLOGMEMBERS es el número máximo de ficheros de los miembros de un grupo. Los ficheros log se combinan en grupos.
- UNDO TABLESPACE es el tablespace que almacenará los datos originales presentes en cualquier tabla o índice antes de hacer cualquier transacción que suponga una modificación sobre aquéllos.
- SYSAUX TABLESPACE es un nuevo tablespace aparecido en la versión 10g de Oracle que permite independizar de SYSTEM objetos de aplicaciones, como el Enterprise Manager, que antiguamente se alojaban en el tablespace SYSTEM.
- DEFAULT TEMPORARY TABLESPACE es el tablespace temporal que utilizarán los usuarios que no tengan explícitamente asignado otro tablespace temporal.
- LOGFILE GROUP es la definición de los grupos de ficheros log.
- CHARACTER SET especifica el juego de caracteres que la base de datos utilizará para almacenar los datos (por ejemplo el WE8ISO8859P15 es el juego que incluye todos los caracteres presentes en los idiomas de Europa Occidental, incluyendo el signo del euro, €, acentos y la ñ).

Todo este farragoso proceso se puede simplificar utilizando el asistente de creación de base de datos de Oracle (dbca, database configuration assistant). Este asistente conduce al administrador automáticamente por el proceso de creación de una instancia con su base de datos. Además, configura automáticamente un repositorio para poder ejecutar Enterprise Manager. Para invocar este asistente tan solo hay que escribir en un terminal de un sistema operativo el comando dbca.

◊ **Actividad 3.7:** Abre la máquina virtual Ubuntu y con un terminal, haz login con el usuario oracle (su oracle). A continuación, ejecuta el comando dbca y sigue los pasos para crear una nueva instancia y su base de datos asociada.

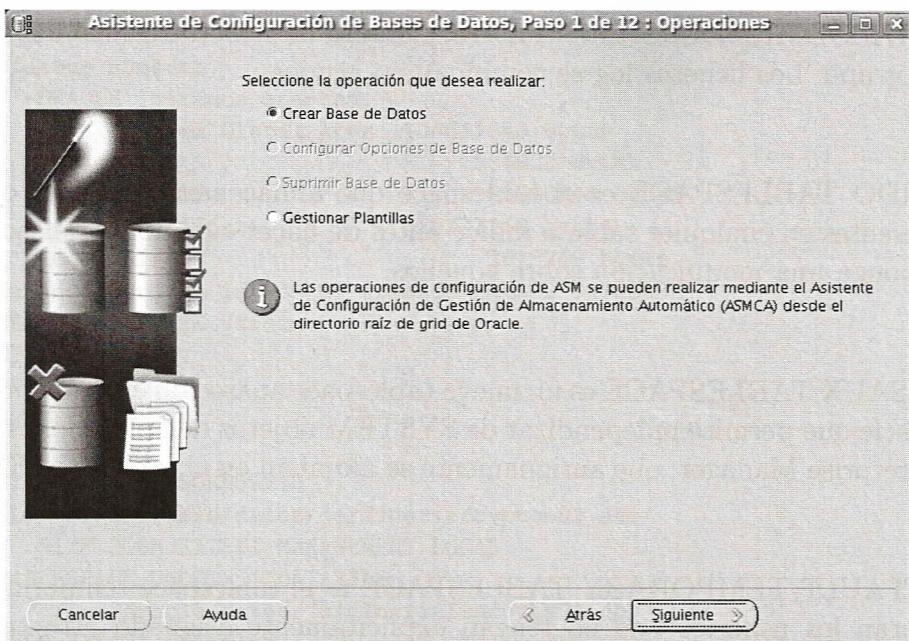


Figura 3.6: Database Configuration Assistant.

3.6. Modificación de una base de datos

El comando ALTER DATABASE permite cambiar las características de funcionamiento de una base de datos. Por ejemplo, en MySQL, tan solo se puede cambiar el juego de caracteres y su colación:

```
#cambia la colación de una base de datos
ALTER DATABASE Startrek COLLATE latin1_spanish_ci;
```

En Oracle, se puede cambiar cualquiera de sus múltiples parámetros, aquí se muestran algunos ejemplos:

```
--Cambia el tamaño del fichero de datos del sistema
SQL> ALTER DATABASE DATAFILE
      '/var/oracle/datos/jardineria/system01.dbf' SIZE 1G;
--Cambia el modo de acceso a la base de datos a solo lectura
SQL> ALTER DATABASE open read only;
--Desactiva la opción de recuperación rápida
ALTER DATABASE flashback off;
```

3.7. Borrado de bases de datos

El comando **DROP DATABASE** es el utilizado para eliminar bases de datos. En MySQL, el comando se acompaña del nombre de la base de datos a eliminar:

```
>mysql -u root -p
drop database Proveedores;
```

En Oracle, no hay que especificarlo, tan solo es necesario conectarse a la instancia y escribir:

```
>sqlplus / as sysdba
shutdown abort; --parada de la instancia
startup mount exclusive restrict; --reinicio en modo exclusivo
drop database; --borrado
exit; --salir
```

Otra forma menos elegante de borrar una base de datos consiste en borrar físicamente todos los ficheros binarios de la base de datos.

3.8. Creación de tablas

Para crear tablas se usa el comando **CREATE TABLE**:

```
CREATE [TEMPORARY] TABLE [esquema.]nombre_tabla
  [(definición_create,...)]
  [opciones_tabla]

definición_create:
  definición_columna
  | [CONSTRAINT [símbolo]] PRIMARY KEY (nombre_columna,...)
  | [CONSTRAINT [símbolo]] FOREIGN KEY (nombre_columna,...)
    [definición_referencia]

definición_columna:
  nombre_columna tipo_datos [NOT NULL | NULL] [DEFAULT valor]
    [UNIQUE [KEY] | [PRIMARY] KEY]
    [definición_referencia]

definición_referencia:
  REFERENCES nombre_tabla [(nombre_columna,...)]
    [ON DELETE {CASCADE | SET NULL | NO ACTION} ]
    [ON UPDATE {CASCADE | SET NULL | NO ACTION} ]
```

El formato que aquí se presenta es un formato reducido, compatible con la mayoría de los gestores de bases de datos, para más información sobre un comando CREATE TABLE para un gestor en particular, se debe recurrir al manual de referencia del gestor.

La porción de sentencia (*definición_create*,...) especificará la definición de los campos que va a contener la tabla y sus restricciones. Los puntos suspensivos sugieren que *definición_create* se puede repetir tantas veces como sea necesario, por ejemplo si hay una tabla con 5 columnas, habrá que especificar 5 veces '*definición_create*'.

El token opcional TEMPORARY se utiliza para crear tablas temporales (esto es, una tabla invisible al resto de usuarios y que se borrará en el momento de la desconexión del usuario que la creó). La primera cláusula que lleva la definición es *definición_columna*, donde se especificará la definición de un campo o columna de la tabla:

definición_columna:

```
nombre_columna tipo_datos [NOT NULL | NULL] [DEFAULT valor]
    [UNIQUE [KEY] | [PRIMARY] KEY]
    [definición_referencia]
```

Se indica el nombre de columna y el tipo de datos, por ejemplo, el Código Postal podría ser un tipo **NUMERIC(5,0)** de MySQL, puesto que los códigos postales tienen 5 dígitos enteros y 0 decimales.

NOT NULL y **NULL** especifican que la columna admite o no admite valores nulos, es decir si un campo puede quedar sin valor, o con valor desconocido.

DEFAULT indica el valor por defecto que toma el campo si no es especificado de forma explícita. Por ejemplo, si se declara la siguiente columna:

CodigoPostal Numeric(5,0) NOT NULL DEFAULT 28941

Se indica que el campo Código Postal es un número de 5 dígitos enteros, que no admite valores nulos, y que en caso de no especificarlo, por ejemplo, en una inserción, el valor que tomará es 28941.

UNIQUE KEY especifica la creación de un índice, esto es, una estructura de datos que permite un acceso rápido a la información de una tabla y además, controla que no haya ningún valor repetido en el campo. Se producirá un error si se intenta insertar un elemento que ya coincide con un valor anterior. A efectos prácticos, la diferencia con una clave primaria es básicamente que un campo UNIQUE puede admitir valores NULL, mientras que un campo que se define como clave primaria no puede admitir nulos (debe ser NOT NULL).

PRIMARY KEY indica que la columna definida es la clave primaria. Una tabla tan solo puede tener una clave primaria. Si se especifica a nivel de *column_definition*, la clave solo puede tener un campo, si la clave se define a nivel de tabla, la clave puede ser multicolumna.

3.8.1. Implementación de restricciones

En la sintaxis de **CREATE TABLE**, *definition_referencia* sirve para crear una clave foránea. De esta manera, se enlaza el campo a su campo origen, es decir se crea una referencia:

Si existe la siguiente tabla creada:

```
create table clientes(
    dni varchar(9) PRIMARY KEY,
    nombre varchar(50),
    direccion varchar(60)
);
```

Se puede crear otra tabla 'mascotas' que contenga el registro de las mascotas de una tienda veterinaria y con un campo que haga referencia al dueño de la mascota:

```
create table mascotas(
    codigo integer PRIMARY KEY,
    nombre varchar(50),
    raza varchar(50),
    cliente varchar(9) REFERENCES clientes(dni)
);
```

Las opciones ON DELETE y ON UPDATE establecen el comportamiento del gestor en caso de que las filas de la tabla padre (es decir, la tabla referenciada) se borren o se actualicen. Los comportamientos pueden ser CASCADE, SET NULL y NO ACTION.

- Si se usa NO ACTION y se intenta un borrado o actualización sobre la tabla padre, la operación se impide, rechazando el borrado o la actualización.
- Si se especifica CASCADE, la operación se propaga en cascada a la tabla hija, es decir, si se actualiza la tabla padre, se actualizan los registros relacionados de la tabla hija, y si se borra un registro de la tabla padre, se borran aquellos registros de la tabla hija que estén referenciando al registro borrado.

- Si se indica SET NULL, se establece a NULL la clave foránea afectada por un borrado o modificación de la tabla padre.

```
create table mascotas(
    codigo integer PRIMARY KEY,
    nombre varchar(50),
    raza varchar(50),
    cliente varchar(9) REFERENCES clientes(dni)
    ON DELETE CASCADE ON UPDATE SET NULL
);
```

Si no se especifica ON DELETE u ON UPDATE por defecto se actúa como NO ACTION.

Oracle no implementa la opción ON UPDATE por lo que hay que recurrir a otros métodos para realizar las acciones de actualización, como por ejemplo, mediante TRIGGERS (disparadores).

La siguiente parte de la sintaxis de CREATE TABLE hace referencia a las declaraciones globales sobre la tabla, esto es, restricciones, claves primarias y foráneas compuestas, etc.

```
[CONSTRAINT [símbolo]] PRIMARY KEY (nombre_columna,...)
| [CONSTRAINT [símbolo]] FOREIGN KEY (nombre_columna,...)
  [definición_referencia]
```

En SQL, todas las restricciones pueden tener un nombre para facilitar su posterior referencia, por tanto cuando aparezca el opcional 'CONSTRAINT [símbolo]' se indica que la definición de una restricción tiene un nombre.

PRIMARY KEY sirve para especificar la creación de una clave primaria a nivel de tabla. Existe la opción de crearla a nivel de columna, mediante *definición_columna* o definirla aquí. Por ejemplo, **es posible definir la siguiente tabla de dos formas:**

```
#primera forma - nivel de columna
create table vehiculo(
    matricula varchar(7) primary key,
    marca varchar(20),
    modelo varchar(20),
    precio numeric(7,2)
);
```

```
#segunda forma - nivel de tabla
create table vehiculo(
    matricula varchar(7),
    marca varchar(20),
    modelo varchar(20),
    precio numeric(7,2) 
    primary key (matricula)
);
```

Ambas formas son equivalentes, la diferencia es que mediante la segunda forma, es posible crear claves primarias compuestas por varios campos. Ejemplo:

```
#Clave primaria = dni + n_ss
create table empleado(
    dni varchar(9),
    n_ss varchar(15),
    nombre varchar(40),
    PRIMARY KEY (dni,n_ss) #compuesta
);
```

De forma análoga, también se puede crear claves foráneas a nivel de tabla, haciendo uso de:

```
[CONSTRAINT [symbol]] FOREIGN KEY (index_col_name,...)
[reference_definition]
```

```
create table mascotas(
    codigo integer PRIMARY KEY,
    nombre varchar(50),
    raza varchar(50),
    cliente varchar(9),
    FOREIGN KEY (cliente) references clientes(dni)
);
```

◊ **Actividad 3.8:** Conéctate a MySQL y prueba a ejecutar los comandos CREATE TABLE anteriores. Posteriormente, conéctate a Oracle mediante SQL*Plus y ejecútalos de nuevo. ¿Son compatibles?

Para terminar, cada gestor de base de datos efectúa sus propias modificaciones al formato de la sintaxis create table. La cláusula *opciones_tabla* permite especificar las peculiaridades de cada gestor con respecto al almacenamiento en soporte físico de sus tablas. Además, cada gestor incorpora diversas características, por ejemplo, Oracle y DB2 implementan tipos de datos distintos a MySQL o a SQL Server y Access.

3.8.2. Tipos de Datos

Los tipos de datos que pueden usarse tanto para MySQL como para Oracle en la definición de una columna son los siguientes:

Tipo de dato	Naturaleza	Tamaño/formato	MySQL	Oracle
TINYINT [UNSIGNED]	Entero	1 byte	X	X
SMALLINT [UNSIGNED]	Entero	2 bytes	X	X
MEDIUMINT [UNSIGNED]	Entero	3 bytes	X	X
INT [UNSIGNED]	Entero	4 bytes	X	X
BIGINT [UNSIGNED]	Entero	8 bytes	X	X
INTEGER [UNSIGNED]	Entero	4 bytes	X	X
DOUBLE [UNSIGNED]	Real Aproximado	8 bytes	X	X
FLOAT [UNSIGNED]	Real Aproximado	4 bytes	X	X
DECIMAL(longitud,decimales)	Real Exacto	Variable	X	
NUMERIC(longitud,decimales)	Real Exacto	Variable	X	
NUMBER(longitud[,decimal])	Real Exacto	Variable		X
DATE	Fecha	'aaaa-mm-dd'	X	X
TIME	Hora	'hh:mm:ss'	X	
TIMESTAMP	Fecha y Hora	'aaaa-mm-dd hh:mm:ss'	X	X
DATETIME	Fecha y hora	'aaaa-mm-dd hh:mm:ss'	X	
CHAR(longitud)	caracteres	Longitud Fija	X	X
VARCHAR(longitud)	caracteres	Longitud Variable	X	X
CHARACTER(longitud)	caracteres	Longitud Variable		X
BLOB	Objetos binarios	Longitud Variable	X	X
TEXT	Campos Memo	Longitud Variable	X	
CLOB	Campos Memo	Longitud Variable		X
ENUM(valor1,valor2,valor3...)	Enumeraciones	Lista de valores	X	
SET(valor1, valor2, valor3...)	Conjuntos	Conjuntos de valores	X	

Cuadro 3.1: Tipos de datos en MySQL y Oracle/DB2.

Puede verse que no todos los tipos de datos están en todos los gestores. Así por ejemplo, el tipo de datos ENUM no está disponible en Oracle y si en MySQL. Se utiliza para crear enumeraciones, es decir, campos que admiten solo valores fijos, por ejemplo *Color ENUM('rojo','amarillo','verde')*. En Oracle, en su lugar, se puede implementar utilizando la directiva CHECK.

3.8.3. Características de la creación de tablas para MySQL

Las opciones de tabla para MySQL son las siguientes:

```
opciones_tabla: opción_tabla [opción_tabla] ...
opción_tabla:
  ENGINE = nombre_motor
  | AUTO_INCREMENT = valor
  | [DEFAULT] CHARACTER SET juego_caracteres [COLLATE colación]
  | CHECKSUM = {0 | 1}
  | COMMENT = 'string'
  | MAX_ROWS = valor
  | MIN_ROWS = valor
```

El almacenamiento físico de una tabla en MySQL está controlada por un software especial denominado *Motor de almacenamiento*. Mediante la opción 'ENGINE=nombre_motor' se indica el motor de almacenamiento para la tabla. Puede ser, entre otros, *innodb*, que son tablas transaccionales con bloqueo de registro y claves foráneas, *myIsam* por defecto usado por MySQL y que genera tablas operadas a gran velocidad, pero sin control de integridad referencial y *Memory*, que genera tablas que están almacenadas en memoria en lugar de un archivo físico.

La opción AUTO_INCREMENT permite indicar el valor inicial para campos de tipo AUTO_INCREMENT. AUTO_INCREMENT indica que ese campo es auto-incrementado después de cada inserción. Un campo definido como auto_increment debe ser numérico; de esta manera, cuando en ese campo se indica el valor NULL en una inserción, el campo toma automáticamente el último valor incrementado en una unidad. Este campo es muy útil para los campos *código* donde se especifican valores clave autogenerados. En MySQL para definir un campo AUTO_INCREMENT se especifica la palabra clave justo a continuación del tipo de datos. Para simular este comportamiento en Oracle, se utiliza una secuencia (SEQUENCE).

'[DEFAULT] CHARACTER SET' Especifica el conjunto de caracteres para la tabla y COLLATE define la colación por defecto de la tabla. Si se especifica CHECKSUM, MySQL mantiene una suma de verificación para todos los registros. El comando **CHECKSUM TABLE** muestra esa suma de verificación (solo para motores de almacenamiento MyISAM).

COMMENT es un comentario para la tabla, hasta 60 caracteres. También es posible crear comentarios para cada una de las columnas. Mediante el token COMMENT

se puede documentar el diccionario de datos. MySQL dispone de este token para comentar no solo tablas, sino también columnas.

MAX_ROWS es el máximo número de registros que se quiere almacenar en la tabla. No es un límite absoluto, sino un indicador que la tabla debe ser capaz de almacenar al menos estos registros. MIN_ROWS es el mínimo número de registros que se planea almacenar en la tabla.

Por último, al igual que en la creación de base de datos, MySQL dispone de la cláusula *IF NOT EXISTS*, para que solamente se cree la tabla si no está creada previamente.

```
#Ejemplo de creación de tabla en MySQL
create table if not exists Pedido(
    codigo int auto_increment primary key,
    fecha datetime,
    estado enum('Pendiente','Entregado','Rechazado')
)
comment = 'tabla de pedidos a proveedores',
auto_increment = 10000
max_rows=1000000
checksum=1
engine=innodb;
```



3.8.4. Características de la creación de tablas para Oracle

En Oracle, la mayoría de las opciones tienen que ver con su almacenamiento físico. Por ejemplo, las tablas deben ser almacenadas en un contenedor llamado *tablespace* (Espacio de tablas). Por defecto, si no se indican opciones de almacenamiento, la tabla se ubica en el tablespace del usuario, pero si se quiere ubicar en otro tablespace, se puede incluir la opción *tablespace nombre* para designar otro tablespace. Además, se puede definir cómo se reserva el espacio en disco para controlar el crecimiento desmedido del tamaño de una tabla mediante la cláusula *storage*. Estas y muchas otras opciones, son propias de Oracle. Por último, cabe destacar la capacidad de Oracle de permitir la creación de restricciones de tipo CHECK, que permite validar el valor de un campo mediante una expresión.

```
--Ejemplo de creación de tablas con opciones propias de Oracle
create table Pedido(
    codigo integer primary key,
    fecha date,
    estado varchar(10),
```

```

constraint c_estado
    check (estado IN ('Pendiente', 'Entregado', 'Rechazado'))
)
tablespace Administracion
storage (initial 100k next 100k minextents 1
        maxextents unlimited pctincrease 0);

```

3.8.5. Consulta de las tablas de una base de datos

Para consultar las tablas disponibles de una base de datos en MySQL, se utiliza el comando SHOW TABLES

```

mysql> show tables;
+-----+-----+
| Tables_in_jardineria |
+-----+-----+
| Clientes              |
| DetallePedidos         |
| ...                   |
| Productos             |
+-----+-----+

```

En Oracle, se pueden consultar las vistas user_tables, dba_tables y all_tables. En Oracle, las vistas que comienzan por user_., aportan información sobre los objetos que posee el usuario conectado. Las vistas que comienzan por dba_ son solo accesibles por los administradores de bases de datos y muestran información sobre todos los objetos. Finalmente, las vistas que comienzan por all_ muestran la información sobre los objetos a los que el usuario tiene acceso, sean suyos o no.

```

SQL> select table_name from user_tables;
TABLE_NAME
-----
PARTIDOS
ESTADISTICAS
JUGADORES
EQUIPOS

```

3.8.6. Consulta de la estructura de una tabla

Para conocer la estructura de una tabla ya creada es posible utilizar el comando

DESCRIBE [esquema.]nombre_tabla

Este comando muestra un listado con las columnas de la tabla, aportando información sobre los tipos de datos, restricciones, etc.

```
SQL> describe nba.equipos;
          Nombre          Nulo      Tipo
-----+-----+-----+
NOMBRE           NOT NULL VARCHAR2(20)
CIUDAD          VARCHAR2(20)
CONFERENCIA     VARCHAR2(4)
DIVISION        VARCHAR2(9)
```

3.9. Modificación de tablas

El comando para modificar una tabla es **ALTER TABLE** y su sintaxis también es bastante compleja:

```
ALTER TABLE nombre_tabla
    especificación_alter [, especificación_alter] ...
```

especificación_alter:

- ADD** definición_columna [**FIRST** | **AFTER** nombre_columna]
| ADD (definición_columna,...)
| ADD [**CONSTRAINT** [símbolo]]
 PRIMARY KEY (nombre_columna,...)
| ADD [**CONSTRAINT** [símbolo]]
 UNIQUE (nombre_columna,...)
| ADD [**CONSTRAINT** [símbolo]]
 FOREIGN KEY (nombre_columna,...)
 [definición_referencia]
- CHANGE** [COLUMN] anterior_nombre_columna definición_columna
 [FIRST|AFTER nombre_columna]
 | ~~RENAME COLUMN anterior_nombre_columna TO nuevo_nombre_columna~~
- MODIFY** definición_columna [**FIRST** | **AFTER** nombre_columna]
| **DROP COLUMN** nombre_columna
| **DROP PRIMARY KEY**
| **DROP FOREIGN KEY** fk_símbolo
| opciones_tabla



Al ver esta sintaxis se puede caer en la tentación de pensar que es mejor borrar una tabla y volver a crearla haciendo las modificaciones oportunas. Esto es posible si la tabla no tiene datos, pero... ¿qué ocurre si hay un centenar o un millar de registros? No queda otra opción que ejecutar una sentencia ALTER TABLE para evitar la pérdida de datos.

- La opción ADD permite añadir una columna, se puede especificar el lugar donde se va a insertar mediante las cláusulas AFTER (después de una columna) y FIRST (la primera columna). Oracle no admite las cláusulas AFTER y FIRST.
- Con la opción MODIFY se cambia el tipo de datos de una columna y se añaden restricciones.
- Con la opción DROP se pueden eliminar las restricciones de claves foráneas y primarias, dejando el tipo de dato y su contenido intacto.
- Las opciones de tabla, al igual que en CREATE TABLE varían con cada SGBD, y sirven principalmente para modificar las características del almacenamiento físico.
- Para cambiar el nombre de una columna, Oracle usa la cláusula RENAME y MySQL la cláusula CHANGE.

Por ejemplo, en MySQL, para añadir a la tabla *mascotas* el campo *especie* después del campo *Raza*, habría que ejecutar la siguiente instrucción:

```
ALTER TABLE Mascotas ADD Especie VARCHAR(10) AFTER Raza;
```

Para eliminar la columna con clave primaria *CodigoCliente* de la tabla *Clientes* en Oracle y establecer como clave primaria el campo *NIF*, hay que ejecutar las siguientes sentencias:

```
ALTER TABLE Clientes DROP PRIMARY KEY;  
ALTER TABLE Clientes DROP CodigoCliente;  
ALTER TABLE Clientes ADD COLUMN Nif VARCHAR(10) PRIMARY KEY FIRST ;
```

3.10. Borrado de tablas

El formato de instrucción en MySQL para el borrado de una tabla es muy sencillo:

```
DROP [TEMPORARY] TABLE  
    tbl_name [, tbl_name] ...
```

Ejemplo:

```
DROP TABLE Mascotas;  
DROP TABLE Clientes, Empleados;
```

En Oracle, cambia un poco pero es muy similar:

```
DROP [TEMPORARY] TABLE tbl_name [CASCADE CONSTRAINT]
```

No se permite el borrado de varias tablas en la misma sentencia, y, de forma opcional, CASCADE CONSTRAINT exige a Oracle que elimine las claves foráneas de las tablas relacionadas (en cascada).

```
--Elimina la tabla Partidos  
DROP TABLE Partidos;  
--Elimina la tabla Jugadores y  
--borra las claves foráneas de otras tablas  
DROP TABLE Jugadores CASCADE CONSTRAINT;
```

3.11. Renombrado de tablas

Para renombrar una tabla en MySQL se usa el comando RENAME TABLE:

```
RENAME TABLE nombre_tabla TO nuevo_nombre_tabla  
    [, nombre_tabla TO nuevo_nombre_tabla] ...
```

Ejemplo:

```
RENAME TABLE Mascotas TO Animales;
```

En Oracle, no hay que indicar el token TABLE, basta con poner:

```
RENAME Jugadores TO Baloncestistas;
```

3.12. Prácticas Resueltas

Práctica 3.1: Modelo físico para stratrekfans.com v.1.0

En un fichero de texto con nombre startrek1.sql, escribe las sentencias SQL para crear el modelo físico de la práctica 2.1. Después, crea una base de datos llamada startrek en un servidor MySQL y ejecuta las sentencias con comando source. A continuación, conéctate a Oracle con el usuario administrador (sqlplus / as sysdba) y crea un usuario en una instancia de oracle con el comando:

```
CREATE USER startrek IDENTIFIED BY startrekpwd
QUOTA UNLIMITED ON USERS;
```

Dale permisos al usuario startrek para iniciar sesión y crear tablas con:

```
GRANT CREATE SESSION,CREATE TABLE TO startrek;
```

Con ese usuario, conéctate a sqlplus y ejecuta el fichero que has creado.

startrek1.sql

```
CREATE TABLE Actores(
    Codigo integer PRIMARY KEY,
    Nombre varchar(50) NOT NULL,
    Fecha DATE NOT NULL,
    Nacionalidad varchar(20) DEFAULT 'EEUU'
);

CREATE TABLE Personajes(
    Codigo integer PRIMARY KEY,
    Nombre varchar(50) NOT NULL,
    Raza varchar(20) NOT NULL,
    Grado varchar(20) NOT NULL,
    CodigoActor integer NOT NULL,
    CodigoSuperior integer NULL,
    FOREIGN KEY (CodigoActor) REFERENCES Actores(Codigo),
    FOREIGN KEY (CodigoSuperior) REFERENCES Personajes(Codigo)
);

CREATE TABLE Planetas(
    Codigo integer PRIMARY KEY,
    Galaxia varchar(50) NULL,
    Nombre varchar(50) NOT NULL
);

CREATE TABLE Capitulos(
    Temporada integer,
    Orden integer,
    Titulo varchar(50) NOT NULL,
```

```
Fecha date NOT NULL,  
PRIMARY KEY (Temporada, Orden)  
);  
  
CREATE TABLE Peliculas(  
    Codigo integer PRIMARY KEY,  
    Titulo varchar(50) NOT NULL,  
    Director varchar(30) NOT NULL,  
    Anyo Date NULL  
);  
  
CREATE TABLE PersonajesCapitulos(  
    CodigoPersonaje integer PRIMARY KEY,  
    Temporada integer NOT NULL,  
    Orden integer NOT NULL,  
    FOREIGN KEY (Temporada,Orden) REFERENCES Capitulos(Temporada,Orden)  
);  
  
CREATE TABLE PersonajesPeliculas(  
    CodigoPersonaje integer REFERENCES Personajes(Codigo),  
    CodigoPelicula integer REFERENCES Peliculas(Codigo)  
);  
  
CREATE TABLE Naves(  
    Codigo integer PRIMARY KEY,  
    NTripulantes integer NULL,  
    Nombre varchar(50) NOT NULL  
);  
  
CREATE TABLE Visitas(  
    CodigoNave integer REFERENCES Naves(Codigo),  
    CodigoPlaneta integer REFERENCES Planetas(Codigo),  
    Temporada integer NOT NULL,  
    Orden integer NOT NULL,  
    FOREIGN KEY (Temporada,Orden) REFERENCES Capitulos(Temporada,Orden)  
);
```

Con el fichero anterior llamado startrek.sql, en mysql:
mysql>CREATE DATABASE Startrek; USE Startrek;
mysql>source startrek.sql
En Oracle: sqlplus / as sysdba
SQL>CREATE USER startrek IDENTIFIED BY startrekpwd;
SQL>GRANT CREATE SESSION,CREATE TABLE TO startrek; SQL>exit
sqlplus startrek/startrekpwd
SQL>@startrek.sql

Práctica 3.2: Restricciones para startrekfans.com v.1.0

Se desea imponer las siguientes características al modelo físico, pero son dependientes del gestor de base de datos donde se van a codificar, así que se codifican en ficheros independientes. Para MySQL se desea:

- Que todas las tablas tengan el motor de almacenamiento innodb para que las claves foráneas no sean ignoradas.
- Que las claves primarias numéricas sean de tipo auto_increment.
- Que el campo Galaxia de la tabla Planetas sea una enumeración de los valores ('Via Láctea', 'Andrómeda', 'Sombrero').

Para Oracle se desea:

- Que las tablas se encuentren ubicadas en el tablespace STARTREK.
- Que el campo NTripulantes de la tabla Naves solo pueda tener un valor entre 1 y 500.
- Que el campo Galaxia de la tabla Planetas solo pueda tener los valores ('Via Láctea', 'Andrómeda', 'Sombrero').

Las sentencias SQL para efectuar los cambios en MySQL son:

Cambios en MySQL

```
#Las tablas se pueden cambiar de dos formas:  
ALTER TABLE Actores ENGINE=innodb;  
ó, si están vacias se puede hacer:  
DROP TABLE Actores;  
CREATE TABLE Actores(  
...  
) ENGINE=innodb;  
#Para cambiar a auto_increment la clave primaria  
#se usa:  
ALTER TABLE Actores MODIFY  
    Codigo integer auto_increment PRIMARY KEY;  
#Para cambiar el campo Galaxia a una enumeración:  
ALTER TABLE Planetas MODIFY  
    Galaxia ENUM ('Via Láctea', 'Andrómeda', 'Sombrero');
```

Las sentencias SQL para efectuar los cambios en Oracle son:

Cambios en Oracle

```
-- Asignar el tablespace STARTREK a las tablas, solo se puede
--hacer si están vacías, y recrearlas:
drop table actores cascade constraints;
CREATE TABLE Actores(
    Codigo integer PRIMARY KEY,
    Nombre varchar(50) NOT NULL,
    Fecha DATE NOT NULL,
    Nacionalidad varchar(20) DEFAULT 'EEUU'
) TABLESPACE STARTREK;
-- Para cambiar el campo NTripulantes:
ALTER TABLE Naves MODIFY NTripulantes integer
check (NTripulantes>=1 and NTripulantes<=500);
-- Para añadir la restricción a galaxia:
ALTER TABLE Planetas MODIFY Galaxia varchar(10)
CHECK (Galaxia IN ('Via Láctea','Andrómeda','Sombrero'));
```

◆

Práctica 3.3: Modelo físico para stratrekfans.com v.2.0

En un fichero de texto con nombre startrek2.sql, escribe las sentencias SQL para modificar el modelo físico de la práctica 2.2. Después, crea una base de datos llamada startrek en un servidor MySQL. Ejecuta las sentencias del fichero en un servidor de MySQL con el comando source. A continuación, conéctate a Oracle con el usuario startrek y ejecuta el fichero que has creado.

startrek2.sql

```
ALTER TABLE Personajes ADD Ciudad varchar(50);
ALTER TABLE Personajes ADD FNacim date;
ALTER TABLE Personajes ADD Planeta integer REFERENCES Planetas(Codigo);
ALTER TABLE Personajes ADD FUltCombate date;
ALTER TABLE Personajes ADD Mentor varchar(50);
ALTER TABLE Personajes ADD FechaGrado date;
```

Con el fichero anterior llamado startrek2.sql, en mysql:

```
mysql> USE Stratrek;
mysql>source stratrek2.sql
En Oracle: sqlplus startrek/startrekpwd
SQL>@startrek2.sql
```

Práctica 3.4: Modelo físico para stratrekfans.com v.3.0

En un fichero de texto con nombre startrek3.sql, escribe las sentencias SQL para modificar el modelo físico de la práctica 2.2. Después, crea una base de datos llamada startrek en un servidor MySQL. Ejecuta las sentencias del fichero en un servidor de MySQL con el comando source. A continuación, conéctate a Oracle con el usuario startrek y ejecuta el fichero que has creado.

startrek3.sql

```
CREATE TABLE Lanzaderas(
    CodigoNave integer,
    Numero integer,
    Personas integer,
    PRIMARY KEY(CodigoNave,Numero),
    FOREIGN KEY(CodigoNave) REFERENCES Naves(Codigo)
);
```

Con el fichero anterior llamado startrek3.sql, en mysql:

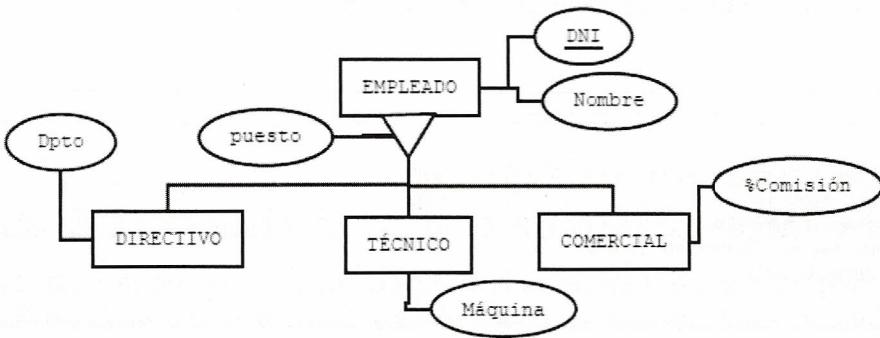
```
mysql> USE Stratrek;
mysql> source stratrek2.sql
En Oracle: sqlplus startrek/startrekpwd
SQL>@startrek2.sql
```



3.13. Prácticas Propuestas

Práctica 3.5: Modelo físico para jerarquías

Dado el diagrama E/R de la figura siguiente, genera el modelo físico, codificando en SQL todas las sentencias para crear las tablas de dos maneras posibles, generando una tabla para cada especialización, y generando una sola tabla para los empleados. Ejecuta los dos modelos tanto en Oracle como en MySQL.



Práctica 3.6: Más modelos físicos

Para los modelos lógicos creados en las prácticas 2.4, 2.5, 2.6 y 2.7, crea el modelo físico asociado mediante varios ficheros con extensión sql. En MySQL crea una base de datos para cada modelo y ejecuta el fichero correspondiente, y en Oracle, crea un usuario distinto que contenga cada uno de los esquemas de las diferentes prácticas. ◇

Práctica 3.7: Parque Ecológico

Un parque ecológico quiere informatizar la gestión de su sistema de información para poder obtener datos más concluyentes sobre las especies migratorias que se establecen en su territorio. El parque desea clasificar todas las especies animales y contabilizar el número de individuos de cada especie que se establecen en el territorio en cada época del año. Se desea que la base de datos almacene datos de nuevas especies y nuevos individuos de cada especie animal. Para cada especie, se requiere el nombre de la especie, características generales de un individuo tipo y sus períodos migratorios. Para cada individuo se tendrá en cuenta el peso, dimensiones, y un código que identifique de forma única cada individuo dentro de su especie. Esta forma de identificación, irá almacenada en un pequeño dispositivo con una batería autónoma implantada en el animal, y que servirá para detectar cuando el individuo sale o entra en el territorio gracias a unas torretas de control instaladas en el perímetro del parque. Estas torretas informan vía inalámbrica al sistema de las idas y venidas de cada individuo. De esta manera, se pueden contabilizar sus migraciones y los posibles descontroles que sufren en el periodo migratorio. Toda la información sobre migraciones de individuos de determinadas especies será enviada cada tres meses a un grupo de expertos biólogos, encargados de hacer una valoración sobre futuros períodos migratorios y posibles alteraciones del comportamiento de las especies. De los biólogos, se quiere almacenar, el nombre, la dirección, provincia y su fecha de titulación. En principio, no se conoce todavía si se implementará en Oracle o MySQL por lo que el diseño debe ser compatible para ambos gestores de bases de datos.

Práctica 3.8: BuscoPareja.net

Se va a crear una página web de contactos en el dominio BuscoPareja.net. Cuando un usuario se registra en el sistema, se almacenan sus datos personales, concretamente su email, su nombre, dirección, ciudad, país, su sexo y orientación sexual, su foto y una password que utilizará junto con su email para acceder al sistema. El usuario a continuación rellena una lista de preferencias o gustos. De cada gusto o preferencia se almacenará el tipo (Deporte, Música, Evento Social), la fecha de la última vez que hizo una actividad de ese estilo y si le gustaría o no que su pareja tuviera la misma preferencia. A partir de esta información se organizan citas entre los contactos en distintas ubicaciones. Se desea registrar las citas entre los contactos almacenando quién se cita con quién, en qué lugar y a qué hora, y si la cita fracasó. Se desea realizar el diseño conceptual, lógico y físico de la base de datos, creando el modelo físico para Oracle y para MySQL.

3.14. Resumen

Los conceptos clave de este capítulo son los siguientes:

- Las herramientas gráficas *ayudan* al informático a gestionar los componentes de una base de datos, pero además, un buen administrador debe conocer los comandos necesarios para hacer las tareas administrativas. Entre las herramientas gráficas que dispone el mercado, se encuentran PhpMyAdmin, de MySQL, Enterprise Manager y Grid Control, de Oracle y Data Studio de DB2.
- Un intérprete de comandos envía comandos en modo texto al SGDB y muestra los resultados en una consola. Estos intérpretes de comandos suelen permitir la ejecución remota de comandos a través de protocolos TCP-IP. Entre otros están, mysql y SQL*Plus e iSQL*Plus(Oracle).
- El DDL o Data Definition Language de SQL define la sintaxis de los comandos CREATE, DROP y ALTER, para crear, borrar y modificar objetos de una BBDD.
- El comando CREATE DATABASE permite crear bases de datos en los SGBD. Algunos SGBD permiten manipular varias BBDD mediante una sola instancia, otros, como Oracle, permiten solo manipular una BBDD por instancia. Además, existen asistentes gráficos (como dbca de Oracle), que permiten de forma sencilla crear bases de datos.
- El comando ALTER DATABASE permite modificar ciertos parámetros de funcionamiento de la BBDD. Estos parámetros dependen del funcionamiento y de la arquitectura del SGBD, por tanto, no son estándar.
- El comando DROP DATABASE borra una base de datos de un servidor.
- Los comandos CREATE TABLE, ALTER TABLE y DROP TABLE están definidos en el estándar SQL para poder crear, modificar y borrar tablas, y, salvo pequeñas diferencias, se pueden escribir estos comandos compatibles para la mayoría de los SGBD.
- Para implementar restricciones se usan las cláusulas PRIMARY KEY (claves primarias) y REFERENCES (para claves Foráneas). Hay dos formas de crearlas, a nivel de tabla y a nivel de columna. A nivel de tabla es posible especificar claves compuestas. Es posible también, definir otras restricciones como las UNIQUE, CHECK, NULL o NOT NULL, etc.
- Los comandos DESCRIBE y RENAME sirven para mostrar la estructura de una tabla y cambiarle el nombre a un objeto.

3.15. Test de repaso

1. ¿Qué función no permite realizar phpMyAdmin?

- a) Ejecución remota de comandos
- b) Copias de seguridad
- c) Creación y borrado de tablas
- d) Parar y arrancar el servidor

2. Oracle Grid Control no permite

- a) Controlar el estado de múltiples BBDD
- b) Controlar múltiples servidores
- c) Creación y borrado de tablas
- d) Parar y arrancar un SGBD

3. El comando CREATE DATABASE no

- a) Es una sentencia DDL
- b) Es compatible entre los distintos SGBD
- c) Permite especificar los tablespaces de una BBDD
- d) Permite especificar el juego de caracteres

4. Una clave foránea

- a) Se define con FOREIGN KEY
- b) Se define con REFERENCES
- c) Puede ser compuesta
- d) Todas las anteriores son correctas

5. Para arrancar el Enterprise Manager

- a) No hay que hacer nada, viene instalado por defecto
- b) Hay que arrancarlo con el comando emctl start dbconsole
- c) Hay que arrancarlo con el comando emctl stop dbconsole
- d) Es un entorno gráfico basado en ventanas

6. La opción ON DELETE CASCADE del comando CREATE TABLE

- a) No funciona en Oracle
- b) No funciona en MySQL
- c) Permite borrar registros relacionados en cascada
- d) Permite borrar claves foráneas en cascada

7. A una restricción

- a) No se le puede poner nombre
- b) Se le puede poner nombre mediante CONSTRAINT
- c) Se le puede poner nombre mediante RESTRICTION
- d) Se le puede poner nombre mediante REFERENCES

8. auto_increment es una opción

- a) Sólo de MySQL
- b) Para incrementar valores numéricos de forma automática
- c) Se suele aplicar a las claves primarias
- d) Todas las anteriores son correctas

9. Con el comando ALTER, no se puede

- a) Borrar una columna
- b) Modificar el tipo de dato de una columna
- c) Cambiarle el nombre a la tabla
- d) Todas las opciones anteriores son posibles

Soluciones: 1.d, 2.c, 3.b, 4.d, 5.b, 6.c, 7.b, 8.d, 9.c.

3.16. Comprueba tu aprendizaje

1. Nombra los distintos tipos de instrucciones DDL que puede haber, distinguiendo el tipo de objeto que se puede crear, borrar o modificar.
2. Pon un ejemplo de un tipo de dato numérico, otro alfanumérico y otro de fecha/hora.
3. ¿Cómo se instala phpMyAdmin en Linux? ¿Necesitas algún otro software para poder instalarlo?
4. Nombra tres herramientas gráficas y sus correspondientes gestores. Añade alguno de los que no aparezcan en el tema (por ejemplo, MMC de SQL Server).
5. ¿Por qué es fundamental para un administrador de bases de datos (DBA) conocer los comandos SQL además de saber usar las herramientas gráficas?
6. Escribe, sin mirar la sintaxis, un comando CREATE TABLE para crear una tabla de *alumnos* con 5 campos a tu elección.
7. Escribe, sin mirar la sintaxis, un comando ALTER TABLE para añadir un campo a la tabla anterior.
8. Escribe, sin mirar la sintaxis, un comando CREATE TABLE para crear la tabla *Notas* y luego, un comando ALTER TABLE para añadir una clave foránea a la tabla anterior.
9. ¿Qué tipos de columna define Oracle que no son compatibles con MySQL?
10. Define para qué sirven los siguientes tokens de la creación de tablas en MySQL:
 - ENGINE
 - AUTO_INCREMENT
 - COLLATION
 - CHARACTER SET
11. ¿Qué diferencia hay entre VARCHAR y CHAR?
12. ¿Qué diferencia hay entre un campo FLOAT y uno NUMBER?
13. Define para qué sirven los tokens de creación de bases de datos TABLESPACE y DATAFILE en Oracle.
14. Comenta para qué sirven las cláusulas ON DELETE y ON UPDATE de REFERENCES.
15. Para las cláusulas anteriores existen tres opciones SET NULL, CASCADE y NO ACTION. Comenta qué efecto tiene sobre los borrados y las modificaciones de registros en tablas relacionadas.