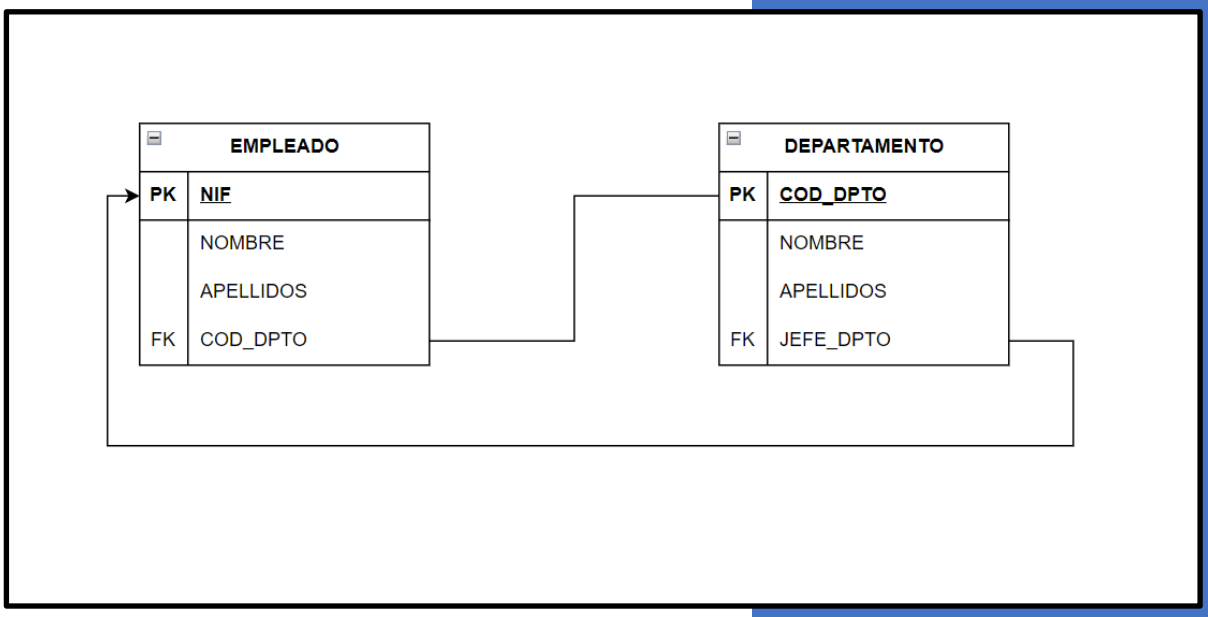
IES Playamar
Curso 2023 - 2024

Tema 3

Creación de bases de datos relacionales



Módulo: Bases de datos

Autor: Manuel Bautista

Este documento ha sido elaborado para el alumnado del módulo “Bases de datos” de los CFGS DAM y DAW del IES Playamar.

Contenido

1. Introducción

Tras el análisis de la especificación, creación del diagrama E/R y obtención del modelo lógico o relacional, el siguiente paso es la creación de la base de datos, entrando así en el nivel interno de la arquitectura ANSI/X3/SPARC.

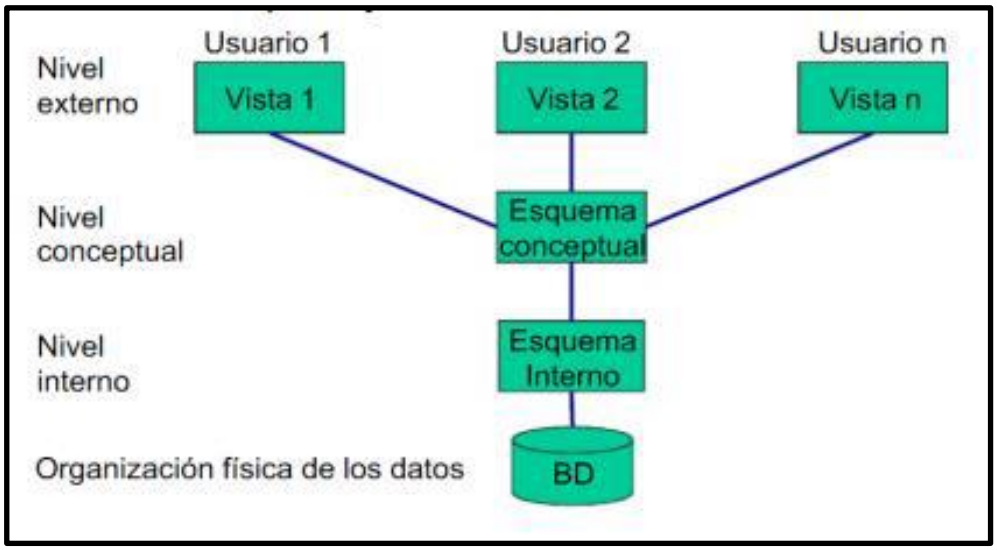


Figura 1. Arquitectura ANSI/X3/SPARC

El diseño de una base de datos es un proceso de tres pasos, donde cada paso incrementa el nivel de detalle o concreción de la base de datos:

1. En primer lugar, se define el **modelo conceptual**. Se crea el diagrama E/R donde se indica *qué* información se debe guardar.
2. En segundo lugar, se define el **modelo lógico**. Se crea el modelo relacional[1] donde se indica *qué* tablas deben ser creadas.
3. En tercer lugar, se define el **modelo físico**. Se crean las instrucciones necesarias para crear la base de datos en un SGBD. Es en este punto donde se indica *cómo* deben ser las tablas. Es decir, el tipo de datos de cada campo y las restricciones impuestas a cada uno de ellos.

Este tema se centra en el último punto. Se estudia la creación, modificación y borrado de los elementos de una base de datos por medio de dos herramientas: el administrador del SGBD de MySQL y el lenguaje SQL.

[1] Modelo relacional porque se trabaja con bases de datos relacionales.

2. El lenguaje SQL

Los SGBD relacionales incorporan un lenguaje para facilitar el acceso a los datos. A lo largo de la historia de las bases de datos, se han creado varios lenguajes, aunque de todos ellos se ha impuesto SQL, incorporado en todos los SGBD actuales.

SQL es un lenguaje para organizar, gestionar y recuperar datos almacenados en una base de datos. El nombre es un acrónimo que significa *Structured Query Language* (Lenguaje estructurado de consultas). Aunque el nombre hace referencia a las consultas (en inglés, *query*), SQL realmente es un lenguaje completo que permite:

- Definir la estructura de la base de datos.
- Añadir, modificar, borrar datos.
- Establecer los permisos sobre los datos.
- Y por supuesto, consultar la información almacenada en la base de datos.

Las características de SQL son las siguientes:

- Es un lenguaje específico de bases de datos, presente en todos los SGBD relacionales.
- Es muy simple y flexible, permitiendo un aprendizaje rápido y dando lugar a instrucciones fáciles de aprender y entender.
- Es declarativo. Las instrucciones especifican *qué es lo que se quiere obtener* en lugar de *cómo hacerlo*[2].
- Está estandarizado por ANSI. No obstante, cada fabricante suele añadir variantes al SQL que incorporan en su SGBD.
- No es sensible a mayúsculas (*case insensitive*) por lo que las instrucciones se pueden escribir en mayúsculas o minúsculas.
- Las instrucciones en SQL se indican en guiones o *scripts*, ya sean en línea o en fichero. Un guion puede contener más de una instrucción, en ese caso deben separarse con el carácter punto y coma (;).
- Es posible añadir comentarios de una línea (con # o --) o de varias líneas (encerrando el comentario entre /* y */).

[2] Los lenguajes declarativos se diferencian de los imperativos (como Java) ya que éstos deben incluir en el código la secuencia de pasos necesaria para obtener los resultados deseados.

El lenguaje SQL se divide en tres sublenguajes:

DDL	DML	DCL
CREATE ALTER DROP	SELECT INSERT UDPATE DELETE	GRANT REVOKE COMMIT ROLLBACK

- DDL (*Data Definition Language*). Incluye las instrucciones necesarias para crear la estructura de la base de datos.
- DML[3] (*Data Manipulation Language*). Incluye las instrucciones necesarias para añadir, consultar, modificar o borrar datos de la base de datos.
- DCL (*Data Control Language*). Incluye las instrucciones necesarias para gestionar los permisos de usuarios y la integridad de las transacciones sobre la base de datos.

3. DDL

El lenguaje de definición de datos permite crear los distintos objetos como son: base de datos, tablas, vistas[4], índices, usuarios. Todos los objetos se tratan del mismo modo:

- Se crean con la instrucción CREATE.
- Se modifican con la instrucción ALTER.
- Se eliminan con la instrucción DROP.

3.1. Bases de datos

Para crear una base de datos se utiliza la instrucción CREATE DATABASE o CREATE SCHEMA. Al crear una base de datos, se pueden indicar dos parámetros opcionales:

- CHARACTER SET: indica el juego de caracteres utilizado en la base de datos.
- COLLATE: colación, indica cómo se trata el juego de caracteres al ordenar o comparar caracteres.

[3] DML se estudia en los siguientes temas.
[4] Las vistas se verán más adelante junto con las consultas.

Si estos parámetros no se indican, se toman los valores por defecto (CHARACTER SET = utf8mb4; COLLATE = utf8mb4_0900_ai_ci). Estos parámetros se pueden modificar en cualquier momento con la instrucción ALTER DATABASE o ALTER SCHEMA. Finalmente, para borrar una base de datos se utiliza la instrucción DROP DATABASE o DROP SCHEMA.

CASO PRÁCTICO 1

Se necesita una base de datos Biblioteca, utilizando el juego de caracteres Latin1 y la colación latin1_spanish_ci:

```
CREATE SCHEMA Biblioteca CHARACTER SET Latin1 COLLATE latin1_spanish_ci
```

Si se prefiere utilizar los valores por defecto se hace lo siguiente:

```
CREATE SCHEMA Biblioteca
```

Una vez creada la base de datos, si fuera necesario cambiar al juego de caracteres utf8 y colación utf8_spanish_ci, se haría lo siguiente:

```
ALTER SCHEMA Biblioteca CHARACTER SET utf8 COLLATE utf8_spanish_ci
```

Finalmente, cuando ya no sea necesaria la base de datos se puede borrar así:

```
DROP SCHEMA Biblioteca
```

RECUERDA: las instrucciones anteriores también funcionan sustituyendo la palabra SCHEMA por DATABASE.

IMPORTANTE

Un SGBD puede almacenar varias bases de datos, para conocer las bases de datos existentes se puede usar la instrucción SHOW DATABASES.

En las instrucciones que se envían al SGBD hay que indicar la base de datos a usar, o bien indicar la base de datos por defecto con la instrucción USE:

```
USE Biblioteca
```

3.2. Tablas

Para crear una tabla se utiliza la instrucción CREATE TABLE. Al crear una tabla se deben indicar su nombre único y, entre paréntesis, las columnas, su clave primaria y las claves foráneas (si las tuviera).

Las columnas se pueden indicar en el orden que se prefiera. Para cada columna incluida en la tabla se debe indicar su nombre (en una tabla no puede haber dos campos con el mismo nombre, pero dos campos de tablas distintas sí pueden tener el mismo nombre), tipo y, opcionalmente, una serie de restricciones:

- AUTO_INCREMENT: para indicar que el campo es auto incrementado.
- NULL o NOT NULL: para indicar si el campo admite valores nulos o no. Por defecto, todos los campos admiten valores nulos (NULL, aunque no es necesario indicarlo) a menos que se indique lo contrario (NOT NULL).
- DEFAULT: para indicar un valor por defecto en el campo en caso de no indicarlo al crear el registro de la tabla.
- UNIQUE: para indicar que el valor del campo debe ser único. La base de datos crea automáticamente un índice para este campo.
- CHECK: para indicar una condición que debe cumplir el valor del campo para darlo por válido.
- PRIMARY KEY: para indicar que el campo es la clave primaria de la tabla.
  NOTA: esta opción sólo es válida si la clave primaria está formada por un único campo. En caso de que la clave primaria sea compuesta (dos o más campos) se debe indicar de otra forma.
- REFERENCES: para indicar que el campo es clave foránea que referencia a otra tabla.
  NOTA: esta opción sólo es válida si la clave foránea está formada por un único campo. En caso de que la clave foránea sea compuesta (dos o más campos) se debe indicar de otra forma.

MySQL incluye muchos tipos de datos y en la documentación de MySQL se puede encontrar información sobre todos los tipos de datos soportados. Muchos tipos de datos son equivalentes a otros, se mantienen distintos nombres por compatibilidad entre los distintos SGBD.

La siguiente tabla resume los tipos de datos más significativos:

VALOR	TIPO	DESCRIPCIÓN
Numérico	INTEGER o INT	Números enteros (4 bytes).
	DECIMAL	Números con decimales. Debe indicarse el tamaño del número y los dígitos dedicados a los decimales. Ejemplo: DECIMAL(5, 2) almacena un número entre -999,99 y 999,99.
Alfanumérico	CHAR	Cadenas de caracteres de longitud fija. Debe indicarse el tamaño. Ejemplo: CHAR(10) almacena cadenas de caracteres de longitud 10. Si la cadena es más corta se desperdicia espacio.
	VARCHAR	Cadenas de caracteres de longitud variable. Debe indicarse el tamaño. Ejemplo: VARCHAR(10) almacena cadenas de caracteres de hasta longitud 10.
Fecha	DATE	Fecha, en formato AAAA-MM-DD.
	DATETIME	Fecha y hora, en formato AAAA-MM-DD hh:mm:ss.
	TIMESTAMP	Fecha y hora, en formato AAAA-MM-DD hh:mm:ss. Tiene un rango de fechas más corto que DATETIME, se usa para almacenar el instante en el que ocurre un evento.
	TIME	Hora, en formato hh:mm:ss.
	YEAR	Año, en formato de cuatro dígitos.
Otros	BLOB	Objeto binario de gran tamaño (Binary Large OBject). Utilizado para almacenar imágenes, sonidos, vídeos…

Una vez indicados los campos de la tabla se indica la clave primaria, con la instrucción PRIMARY KEY (*campos*), siendo *campos* la secuencia de campos, separadas por comas, que forman la clave primaria. Esta forma de indicar la clave primaria vale para cualquier situación (clave primaria simple formada por un campo o clave primaria compuesta de dos o más campos).

Finalmente, se indican las claves foráneas con la instrucción FOREIGN KEY. Se deben indicar los campos que forman la clave foránea y la tabla a la que se hace referencia. Esta forma de indicar la clave foránea vale para cualquier situación (clave foránea simple formada por un campo o clave foránea compuesta de dos o más campos).

Las claves foráneas hacen referencia a registros de otras tablas de la base de datos y por tanto debe garantizarse la integridad referencial. Es decir, que el registro al que se hace referencia realmente debe existir. Para ello, el SGBD debe controlar el borrado y/o modificación del registro referenciado por la clave foránea y realizar los cambios oportunos para garantizar la integridad referencial. En la definición de la clave foránea se indican la acción a realizar en borrado y modificación de tres posibles:

- NO ACTION o RESTRICT: no se permite la operación si el registro es referenciado por clave foránea. Es la opción por defecto en caso de no indicar acción a realizar.
- CASCADE: la operación sobre el registro referenciado por clave foránea es propagado al registro que lo referencia.
- SET NULL: si se realiza una operación sobre un registro referenciado por clave foránea, el registro que lo referencia pone a NULL los campos de la clave foránea.

CASO PRÁCTICO 2

Se necesita una tabla para empleados, que incluya un identificador numérico (clave primaria, autoincremento), nombre (obligatorio, longitud 30), primer apellido (obligatorio, longitud 30), segundo apellido (opcional, longitud 30), NIF (obligatorio y único), fecha de nacimiento (opcional), salario (obligatorio, con salario mínimo de 20.000€) y código de departamento (clave foránea, si no se indica se asocia por defecto al departamento 1). Si se actualiza un código de departamento se debe actualizar la referencia en el empleado, y si se borra un departamento los empleados asociados a dicho departamento se quedan sin departamento.

La siguiente instrucción crea la tabla de empleados según las indicaciones:

```
CREATE TABLE Empleados
(
        id_empleado INT AUTO_INCREMENT NOT NULL,
        nombre VARCHAR(30) NOT NULL,
        apellido1 VARCHAR(30) NOT NULL,
        apellido2 VARCHAR(30),
        NIF CHAR(9) NOT NULL UNIQUE,
        fecha_nacimiento DATE,
        salario INT NOT NULL CHECK (salario >= 20000),
        cod_departamento INT DEFAULT 1,
        PRIMARY KEY (id_empleado),
        FOREIGN KEY cod_departamento REFERENCES Departamentos(cod)
                ON DELETE SET NULL ON UPDATE CASCADE
)
```

NOTA: se hace referencia a la tabla Departamentos, con clave primaria cod. La tabla Departamentos debe ser creada antes para que se pueda crear la tabla Empleados.

IMPORTANTE

Para conocer las tablas definidas en una base de datos, se puede utilizar la instrucción SHOW TABLES. Para conocer la estructura de una tabla, se puede usar el comando DESCRIBE o DESC:

```
DESCRIBE Empleados
```

Una vez creada una tabla, ésta puede ser modificada. Las modificaciones se realizan con la instrucción ALTER TABLE y se pueden realizar las siguientes modificaciones:

- Añadir, modificar, renombrar o eliminar campos.
- Añadir o eliminar la clave primaria, así como las claves foráneas.

Para añadir un nuevo campo a la tabla, se usa ADD. Por defecto el nuevo campo se añade al final, pero es posible añadirlo al comienzo (FIRST) o a continuación de otro campo (AFTER *campo*). Para modificar un campo se usa MODIFY. Esta opción permite cambiar el tipo del campo y/o sus restricciones. Para renombrar un campo se usa RENAME COLUMN. Finalmente, para eliminar un campo se usa DROP.

CASO PRÁCTICO 3

Se desea añadir en la tabla Empleados un nuevo campo irpf para almacenar el porcentaje de retención a aplicar en la nómina. Debe ser de tipo decimal (2 dígitos para la parte entera y dos decimales), teniendo un valor por defecto del 2%. Como va relacionado con el salario interesa añadirlo justo después de salario.

```
ALTER TABLE Empleados ADD irpf DECIMAL(4, 2) DEFAULT 2 AFTER salario
```

Se desea cambiar el campo apellido1. Actualmente es de longitud 30, se quiere ampliar a 40 caracteres.

```
ALTER TABLE Empleados MODIFY apellido1 VARCHAR(40)
```

Se desea renombrar el campo apellido1 a apellidos.

```
ALTER TABLE Empleados RENAME COLUMN apellido1 TO apellidos
```

Se desea eliminar el campo apellido2

```
ALTER TABLE Empleados DROP apellido2
```

Si fuera necesario cambiar la clave primaria de una tabla, primero se elimina la clave anterior y luego se define (añade) la nueva clave primaria.

CASO PRÁCTICO 4

Se desea definir el NIF como clave primaria de la tabla Empleados. Por tanto, el campo id_cliente ya no se necesita.

```
ALTER TABLE Empleados MODIFY id_empleado INT;
ALTER TABLE Empleados DROP PRIMARY KEY;
ALTER TABLE Empleados DROP id_cliente;
ALTER TABLE Empleados ADD PRIMARY KEY (NIF)
```

NOTA: el campo id_empleado se definió como autoincrementado. Esta propiedad debe eliminarse primero para poder borrar la clave primaria. De ahí que en primer lugar se modifique el tipo del campo.

De un modo parecido se pueden añadir nuevas claves foráneas o eliminar alguna existente.

CASO PRÁCTICO 5

Todo empleado tiene un jefe en la empresa. Se desea añadir la referencia al jefe.

```
ALTER TABLE Empleados ADD jefe CHAR(9);
ALTER TABLE Empleados ADD FOREIGN KEY (jefe) REFERENCES Empleados(NIF)
```

Finalmente, cuando no se necesita más una tabla, se puede borrar de la base de datos con la instrucción DROP TABLE. Esta instrucción borra la estructura y los datos que tenga almacenados.

CASO PRÁCTICO 6

Si hiciera falta borrar la tabla Empleados, se haría así:

```
DROP TABLE Empleados
```

IMPORTANTE

Modificar una tabla puede ser un proceso laborioso, ¿no sería mejor borrar la tabla y hacerla de nuevo? Esta opción solo debería contemplarse si la tabla aun está vacía o tiene poca información almacenada. Si la tabla ya almacena muchos datos, en ese caso es recomendable evitar su borrado y en su lugar realizar su modificación.

También es posible renombrar el nombre de una tabla con la instrucción RENAME TABLE.

CASO PRÁCTICO 7

Se desea cambiar el nombre de la tabla Empleados a Trabajadores.

```
RENAME TABLE Empleados TO Trabajadores
```

3.3. Índices

Un índice es una estructura de la base de datos que mejora el rendimiento de las consultas permitiendo un acceso más rápido a la información. El SGBD crea automáticamente índices para cada clave primaria de una tabla y para cada campo que se ha creado con la opción UNIQUE.

Un índice se crea con la instrucción CREATE INDEX, donde se indica la tabla en la que se crea el índice, los campos que se incluyen y, para cada campo, si el índice ordena en forma ascendente (es el valor por defecto) o de forma descendente. Para eliminar un índice se utiliza la instrucción DROP INDEX.

¿POR QUÉ NO EXISTE ALTER INDEX?

Al crear un índice, el SGBD construye una estructura donde se almacena la información necesaria para acelerar los accesos a datos. Si hay que cambiar el índice dicha información queda obsoleta y no se puede aprovechar. Por tanto, para modificar un índice es necesario borrarlo y crearlo de nuevo.

CASO PRÁCTICO 8

Se desea crear un índice para los nombres y apellidos de los empleados, ordenando nombre en sentido descendente y apellidos en orden ascendente.

```
CREATE INDEX idx_nombres_apellidos ON Empleados(nombre DESC, apellidos)
```

Se desea crear un índice para los salarios.

```
CREATE INDEX idx_salario ON Empleados(salario)
```

Cuando ya no sea necesario este índice, se puede borrar.

```
DROP INDEX idx_salario ON Empleados
```

IMPORTANTE

Si bien los índices aceleran las consultas en bases de datos, conviene no abusar de ellos porque penalizan (ralentizan) el resto de operaciones sobre los datos (inserciones, modificaciones y borrado).

3.4. Usuarios

Un SGBD puede definir distintos usuarios con diferentes permisos sobre las bases de datos. Para crear un nuevo usuario se utiliza la instrucción CREATE USER, indicando su nombre y contraseña. Una vez creado un usuario, se puede renombrar y/o cambiar su contraseña. Finalmente, cuando ya no se necesite, se puede borrar.

CASO PRÁCTICO 9

Se desea crear el usuario Alumno.

CREATE USER Alumno IDENTIFIED BY 'AlumnoDeCiclo'

Se desea renombrar a AlumnoCiclo.

RENAME USER Alumno TO AlumnoCiclo

Se desea cambiar su contraseña.

SET PASSWORD FOR AlumnoCiclo = 'NuevaContraseña'

Para eliminar el usuario.

DROP USER AlumnoCiclo

4. DCL

El lenguaje de control de datos gestiona dos funciones:

- Controlar los accesos a los objetos de la base de datos, permitiendo así otorgar o retirar permisos a los distintos usuarios de la base de datos.
- Controlar las transacciones garantizando que las operaciones sobre base de datos se realizan de forma atómica.

Una vez creado un usuario, hay que indicar los permisos o privilegios que va a tener. Un usuario puede obtener permisos para manipular objetos de una base de datos con el comando GRANT. Asimismo, se le pueden denegar permisos con el comando REVOKE. También es posible indicar si el permiso es sólo para el usuario o a su vez éste puede otorgar dicho permiso a otros usuarios (WITH GRANT OPTION).

Existen muchas opciones para indicar permisos, las más comunes se resumen en la siguiente tabla:

OPCIÓN	DESCRIPCIÓN
ALL	Otorga todos los permisos (excepto GRANT OPTION).
ALTER	Permite modificar una tabla.
CREATE	Permite crear una tabla.
DELETE	Permite borrar datos de una tabla.
DROP	Permite borrar una tabla.
INDEX	Permite crear y borrar índices.
INSERT	Permite insertar datos en una tabla.
SELECT	Permite consultar datos de una tabla.
UPDATE	Permite actualizar datos de una tabla.
GRANT OPTION	Permite dar permisos a otros usuarios.

CASO PRÁCTICO 10

Se desea dar al usuario AlumnoCiclo permisos de consulta a los campos nombre y apellidos de la tabla Empleados.

```
GRANT SELECT(nombre, apellidos) ON Empleados TO AlumnoCiclo
```

Se desea dar al alumno AlumnoCiclo permisos de inserción, consulta, borrado y modificación sobre todas las tablas de la base de datos.

```
GRANT SELECT, INSERT, DELETE, UPDATE ON * TO AlumnoCiclo
```

Se desea dar permiso completo al usuario Profesor, con opción de dar permisos también:

```
GRANT ALL ON * TO Profesor WITH GRANT OPTION
```

Se desea que el usuario AlumnoCiclo tenga permisos de sólo lectura, hay que quitarle los permisos de inserción, modificación y borrado.

```
REVOKE INSERT, DELETE, UPDATE ON * FROM AlumnoCiclo
```

DCL también incluye instrucciones para controlar las transacciones. Una transacción es un bloque de operaciones sobre base de datos que se realiza de forma atómica. Es decir, o se hacen todas o ninguna. Se da así la posibilidad de deshacer operaciones en caso de error.

Cuando un usuario inicia sesión en la base de datos comienza una nueva transacción que finaliza cuando se indica la instrucción COMMIT o se realiza una operación DDL. Una vez finalizada la transacción, comienza automáticamente una nueva.

Si en el trascurso de una transacción ocurre un error, todos los cambios realizados en la transacción se pueden deshacer con la instrucción ROLLBACK, volviendo la base de datos al estado al inicio de la transacción.

5. El administrador del SGBD

Los SGBD actuales incorporar una herramienta gráfica que facilita las tareas sobre la base de datos además de incorporar herramientas para las copias de seguridad, la exportación e importación de datos o la gestión de usuarios entre otras.

MySQL incorpora la herramienta MySQL Workbench, que permite la gestión de las bases de datos del SGBD, tanto en modo local como remoto.

IMPORTANTE

Aunque las herramientas gráficas facilitan las tareas de administración, es muy aconsejable dominar el lenguaje SQL ya que el uso de un guion en SQL permite automatizar tareas.

Una ventaja de utilizar SQL es para hacer más sencillo el despliegue de los cambios de la base de datos. La forma de trabajar más común en cualquier desarrollo de software es el siguiente:

- En un entorno de desarrollo se crea todo el código de los programas.
- Los cambios desarrollados se llevan a un entorno de preproducción para realizar las pruebas oportunas y validar la corrección del código.
- Finalmente, el nuevo código se despliega en el entorno de producción, donde se pone a disposición de los usuarios.

Trabajando de este modo, todos los cambios realizados en la base de datos de desarrollo deben replicarse en la base de datos de preproducción y producción. Realizar los cambios a mano con el administrador del SGBD supone hacer el cambio tres veces, con SQL se facilita porque, una vez conocidos los cambios realizados en desarrollo, basta con ejecutar el guion de cambios en el resto de entornos.

6. Resumen

El lenguaje SQL es la herramienta fundamental para interactuar con la información contenida en una base de datos relacional. Estandarizado por ANSI, permite al programador enviar instrucciones al SGBD sin necesidad de especificar paso a paso cómo debe llevar a cabo su tarea.

SQL se divide en el lenguaje de definición de datos (DDL), el lenguaje de manipulación de datos (DML) y el lenguaje de control de datos (DCL), destinados a trabajar con objetos de la base de datos, con el contenido de dichos objetos y con la información de control, respectivamente.

El DDL permite crear, modificar y eliminar la base de datos y sus objetos (tablas, campos, vistas, índices y usuarios). El DCL permite controlar los accesos a los objetos de la base de datos (permitiendo así otorgar o retirar permisos a los distintos usuarios de la base de datos) y gestionar las transacciones sobre las bases de datos (garantizando la atomicidad de las operaciones para preservar la integridad de la base de datos).

Las operaciones sobre bases de datos se pueden realizar también con el Administrador del SGBD, que ofrece una interfaz gráfica que simplifica el trabajo con las bases de datos.