3. Bases de datos relacionales

Acceso a Datos

Alejandro Roig Aguilar
alejandro.roig@iesalvarofalomir.org

IES Álvaro Falomir
Curso 2023-2024

Objetivo de la unidad y criterios de evaluación

RA2. Desarrolla aplicaciones que gestionan información almacenada en bases de datos relacionales identificando y utilizando mecanismos de conexión.

a) Se han valorado las ventajas e inconvenientes de utilizar conectores.

b) Se han utilizado gestores de bases de datos embebidos e independientes.

c) Se ha utilizado el conector idóneo en la aplicación.

d) Se ha establecido la conexión.

e) Se ha definido la estructura de la base de datos.

f) Se han desarrollado aplicaciones que modifican el contenido de la base de datos.

Objetivo de la unidad y criterios de evaluación

RA2. Desarrolla aplicaciones que gestionan información almacenada en bases de datos relacionales identificando y utilizando mecanismos de conexión.

g) Se han definido los objetos destinados a almacenar el resultado de las consultas.

h) Se han desarrollado aplicaciones que efectúan consultas.

i) Se han eliminado los objetos una vez finalizada su función.

j) Se han gestionado las transacciones.

k) Se han ejecutado procedimientos almacenados en la base de datos.

Bases de datos vs ficheros (I)

	Ficheros	Base de datos
Estructura	Se gestionan y organizan en el sistema de almacenamiento	Se gestiona por un software gestor de bases de datos (SGBD)
Redundancia de datos	Puede haber redundancia de datos	No hay redundancia de datos
Respaldo y recuperación	No proporciona sistemas de copia y recuperación ante pérdida de datos	Sí hay sistemas de copia y recuperación ante pérdida de datos
Procesamiento de consultas	No hay un sistema eficiente con procesamiento de consultas	Un SGBD permite un procesamiento eficiente de consultas
Consistencia	La consistencia de datos es menor	Hay mayor consistencia debido al proceso de normalización

Bases de datos vs ficheros (II)

	Ficheros	Base de datos
Restricciones de seguridad	Aporta menos seguridad	Un SGBD tiene más mecanismos de seguridad que los ficheros
Coste	Más barato que un SGBD	Más caro que un SGBD
Acceso de usuario	No permite el acceso concurrente de usuarios a ficheros	Sí se permite el acceso concurrente a los datos
Abstracción de datos	Los detalles de almacenamiento y representación son conocidos	Los detalles internos de la base de datos están ocultos al usuario
Integridad	Difícil de implementar	Fácil de implementar

Bases de datos relacionales

Es un tipo de BBDD que cumple con el modelo relacional de Codd. Los sistemas de BBDD relacionales utilizan SQL (Structured Query Language) para consultar y mantener la BBDD .

Una BBDD se compone de tablas, y cada tabla se compone de un conjunto de campos (columnas) y registros (filas).

Las tablas se relacionan entre ellas por medio de las claves primarias y las claves ajenas.

- Las claves primarias son la clave principal de un registro dentro de una tabla y deben cumplir con la integridad de los datos.
- Las claves ajenas se colocan en la tabla relacionada y contienen el mismo valor que la clave primaria del registro de la propia tabla.

Bases de datos relacionales

EMPLEADOS

num_emp	nom_emp
7643	GIL SAAVEDRA, FERNANDO
8257	MARTÍN FERRIS, MARÍA
9087	BORGES RULFO, JUAN

PROYECTOS

id_proy	nom_proy
2	RUEDAS CUADRADAS
5	PAPEL ELECTRÓNICO
8	PERMUTADOR CUÁNTICO

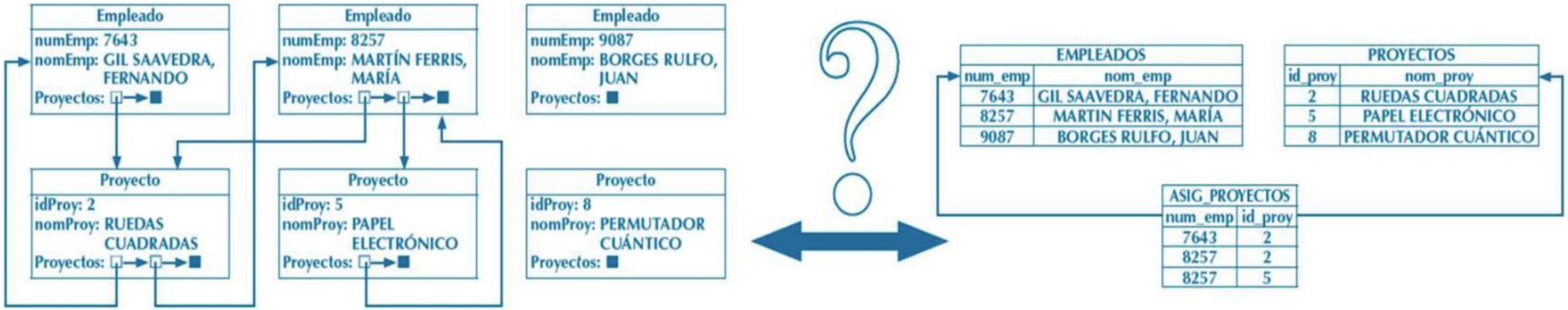
ASIG_PROYECTOS

num_emp	id_proy
7643	2
8257	2
8257	5

Desfase objeto-relacional

Existen diferencias entre el paradigma de programación orientada a objetos y las bases de datos relacionales.

- El modelo relacional de la base de datos trata de relaciones y conjuntos.
- La POO trata los datos como objetos y asociaciones entre ellos.



Desfase objeto-relacional

Esto conlleva un conjunto de problemas a resolver denominado desfase objeto-relacional, ante los que se han planteado varias soluciones:

- BD objeto-relacionales: Son BBDD relacionales con capacidad para gestionar objetos. Destacan Oracle y PostreSQL.
- Mapeo objeto-relacional (ORM): Es una solución más flexible con la ventaja de proporcionar soporte para múltiples BBDD. Existen múltiples herramientas, bibliotecas o frameworks para ORM, entre las que destaca Hibernate.

Acceso a bases de datos relacionales

JDBC (Java DataBase Conectivity) es una API basada en el estándar ODBC, que permite ejecutar operaciones sobre bases de datos desde Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de BBDD utilizado.

Un driver es un conjunto de clases encargadas de implementar los interfaces del API y acceder a la base de datos. Estos conectores los proporcionan los propios fabricantes de la BBDD.

JDBC ofrece interfaces para:

- Conectar a una BBDD
- Ejecutar una consulta
- Procesar los resultados

Conexión JDBC a BBDD

A nivel genérico, los pasos necesarios serán:

- Abrir una conexión.
- Preparar un objeto de tipo PreparedStatement, que representará a una (o varias) sentencia(s) SQL.
- A través de dicho PreparedStatement, lanzar consultas y obtener resultados.
- Liberar los recursos al terminar.
- Usar excepciones para gestionar los posibles errores.

Operaciones con JDBC

Se pueden usar tres tipos de sentencias:

- Statement: Sentencias SQL.
- PreparedStatement: Consultas precompiladas, eficientes y parametrizables, evitando ataques de SQL Injection. Es el tipo recomendado, siendo su uso prácticamente obligatorio.
- CallableStatement: Ejecutar procedimientos almacenados en la base de datos.

JDBC distingue dos tipos de consultas:

- Selección: SELECT
- Modificación: UPDATE, INSERT, DELETE y sentencias DDL.

JDBC Consultas

Una consulta de Selección (SELECT) sobre una tabla de la BBDD devuelve un conjunto de datos organizados en registros.

- Se ejecutan mediante el método executeQuery.
- El resultado se almacena en un objeto de la clase ResultSet, que solo se puede recorrer fila por fila, usando su método next().
- Si no sabemos las columnas que nos devuelve la consulta, podemos usar el método getMetaData().

Una consulta de Modificación devuelve un entero con el número de registros afectados.

- Se ejecutan mediante el método executeUpdate.

executeQuery() vs executeUpdate()

executeQuery()	executeUpdate()
executeQuery() se utiliza para recuperar información de la base de datos	**executeUpdate()** se utiliza para actualizar o modificar la base de datos
Devuelve un objeto de la clase **ResultSet**	Devuelve un valor **entero**
Se utiliza normalmente para **ejecutar consultas SELECT**	Se utiliza para ejecutar **consultas:** • **DML,** como INSERT, DELETE o UPDATE • **DDL** como CREATE y DROP

JDBC Transacciones

Una transacción es un conjunto de sentencias SQL que se ejecutan como si de una sola se tratara.

- Si alguna de las sentencias falla o produce un error, no se ejecuta ninguna más y se deshacen los cambios que hayan podido efectuar las ya ejecutadas.
- Requiere de tres nuevas instrucciones:
  - Comienzo de una transacción → setAutoCommit(false)
  - Finalización → commit()
  - Indicar que la transacción actual debe abortarse y los cambios deben ser restaurados → rollback()