

TEMA 64: EL MODELO RELACIONAL. EL LENGUAJE SQL. NORMAS Y ESTÁNDARES PARA LA INTEROPERABILIDAD ENTRE GESTORES DE BASES DE DATOS RELACIONALES

Actualizado a 23/04/2023

1. EL MODELO RELACIONAL

Propuesto por E.F. Codd en 1970, es un modelo de datos basado en el concepto matemático de relación de la teoría de conjuntos, donde los datos se estructuran en forma de relaciones modeladas mediante tablas.

LAS 12 REGLAS DE CODD

Las 12 reglas de Codd definen los requerimientos que un sistema de administración de BB.DD. ha de cumplir para ser considerado relacional.

0. Gestión de una BDR	Usa exclusivamente sus capacidades relacionales para gestionar por completo las BB.DD.
1. Representación de la información	Toda la información almacenada debe representarse explícitamente a nivel lógico, y de manera única, por medio de valores en tablas .
2. Acceso garantizado	Para todos y cada uno de los datos se garantiza que son lógicamente accesibles, sin ambigüedades, mediante una combinación de nombre de tabla, valor de clave primaria y nombre de una columna .
3. Tratamiento sistemático de valores nulos	Los valores nulos (información desconocida o inaplicable) han de ser tratados sistemáticamente por el sistema, el cual ha de ofrecer las facilidades necesarias para su tratamiento.
4. Catálogo dinámico en línea basado en el modelo relacional	La representación de la metainformación (descripción BD) debe ser igual a la de los otros datos y su acceso debe poder realizarse por medio del mismo lenguaje relacional que se utiliza para los demás datos; es decir, el modelo de datos para la metainformación debe ser el relacional.
5. Sublenguaje completo de datos	Debe existir un lenguaje que permita un completo manejo de la BD (definición de datos, definición de vistas, manipulación de datos, restricciones de integridad, gestión de autorizaciones y gestión de transacciones).
6. Actualización de vistas	Toda vista teóricamente actualizable debe poder ser actualizada por el sistema.
7. Inserciones, modificaciones y eliminaciones de alto nivel	Todas las operaciones de manipulación de datos (consulta, inserción, modificación y borrado) deben operar sobre conjuntos de filas (lenguaje no navegacional) y no sólo sobre registros individuales.
8. Independencia física de los datos	El acceso lógico a los datos debe mantenerse incluso cuando cambien los métodos de acceso o la forma de almacenamiento.
9. Independencia lógica de los datos	Los programas de aplicación y las operaciones a través de terminal no deben verse afectados por cambios realizados en las tablas que estén permitidos teóricamente y que preserven la información.
10. Independencia de la integridad	Las reglas de integridad de una BD deben ser definibles por medio de un sublenguaje de datos relacional y habrán de almacenarse en el catálogo de la BD (metabase), no en los programas de aplicación.
11. Independencia de la distribución	El usuario final no ha de ver que los datos están distribuidos en varias ubicaciones. Los usuarios deben tener siempre la impresión de que los datos se encuentran en un solo lugar.
12. Regla de la no subversión	Si un SGBD soporta un lenguaje de bajo nivel que permite el acceso fila a fila (un solo registro cada vez), éste no puede utilizarse para saltarse las reglas de integridad expresadas por el lenguaje de más alto nivel (múltiples registros a la vez).

CONCEPTOS

- **Dominio:** conjunto de posibles valores que puede tomar un atributo. La unión de todos los dominios de los atributos de una relación se conoce como el **universo** de la relación.
- **Relación (o tabla):** Conjunto definido de atributos y reglas. El **esquema** de una relación es la descripción de su estructura interna con el formato Relación (Atributo1, Atributo2..., AtributoN)
- **Tupla (o fila):** conjunto de atributos relacionados que siguen la estructura de una relación.
- **Atributo:** conjunto de columnas que representan propiedades de la relación. Los atributos toman sus valores de los dominios.
- **Cardinalidad:** número de tuplas o filas.
- **Grado:** número de atributos o columnas.
- **Vista:** tabla virtual que representa el resultado de una consulta sobre una o varias relaciones (tablas). Sólo se almacena su definición y los datos se extraen dinámicamente por lo que no es muy eficiente. Para aumentar la velocidad, se crean **vistas materializadas**, que se almacenan en caché de forma real y se actualiza de forma periódica (hándicaps: ocupan espacio y riesgo de desincronización con los datos de las tablas).
- **Clave:** atributo o conjunto mínimo de atributos que identifican por su valor a cada tupla de una relación. Tipos de claves:
 - Clave simple: aquella formada por un único atributo.
 - Clave compuesta: formada por dos o más atributos.
 - Claves candidatas: conjunto de claves posibles de una relación. De entre ellas, se elige una como **Clave primaria (Primary Key)**, que identifica unívocamente a cada tupla.
 - **Clave externa (Foreign Key):** un atributo o grupo de atributos es clave externa en una relación R1 cuando es clave primaria de otra relación R2.
 - Superclave: Aquella formada por varios atributos que identifican unívocamente una tupla, pero a la que pueden pertenecer otros atributos innecesarios para la identificación unívoca de la tupla.
- **Índice:** son tablas internas de la BD para acelerar las consultas sobre las tablas. Utilizan las claves para acceder de forma rápida a cada tupla.

RESTRICCIONES

- **Restricciones inherentes al modelo de datos relacional:**
 - **Restricción de dominio:** los valores de los atributos de una relación deben ser atómicos.
 - **Restricción de clave:** en una relación no puede haber ninguna tupla repetida.
- **Restricciones de integridad o semánticas:**
 - **Regla de integridad de las entidades** (o Regla de la Clave Primaria): ningún valor componente de una clave primaria puede ser nulo y el valor de la clave no puede estar repetido.
 - **Regla de integridad referencial** (o Regla de la Clave Externa): cualquier valor no nulo de la clave externa debe tener asociado un valor en la clave primaria de la relación objetivo.

LENGUAJES FORMALES DEL MODELO DE DATOS RELACIONAL

- **Álgebra relacional:** es un sistema cerrado de operaciones definidas sobre relaciones.
 - **Operandos:** son relaciones (los resultados también).
 - **Operadores:**

OPERADORES PRIMITIVOS	
Selección (?)	Relación formada por el subconjunto de tuplas que satisface dicha expresión.
Proyección (σ)	Relación definida sobre los atributos donde se aplica, eliminando las filas repetidas.
Unión (π)	Relación formada por las tuplas que pertenezcan a r o a r' o a ambas.
Diferencia	Relación formada por el conjunto de tuplas que pertenecen a r pero no a r' .
Producto cartesiano (\times)	Relación formada concatenando cada tupla de la primera relación con cada una de las tuplas de la segunda.

OPERADORES DERIVADOS	
Intersección (\cap)	Relación formada por las tuplas que pertenezcan a ambas relaciones $r \cap s = r - (r - s)$
División ($:$)	Relación que al complementarse con las tuplas de la segunda relación todas las tuplas obtenidas forman parte de la primera relación
Combinación o Join natural (\bowtie)	Relación formada por todos los pares de tuplas que, en el producto cartesiano de ambas, cumplen una condición especificada $r \bowtie_p s = \sigma_p(r \times s)$
Outer join	<p>-Left outer join: join natural de las dos relaciones, añadiendo al resultado todas las tuplas que están en la primera de las dos relaciones para las que no aparezcan tuplas asociadas a ellas en la segunda</p> <p>-Right outer join: join natural de las dos relaciones, añadiendo al resultado todas las tuplas que están en la segunda de las dos relaciones para las que no aparezcan tuplas asociadas a ellas en la primera</p> <p>-Full outer join: join natural de las dos relaciones, añadiendo al resultado tanto las tuplas que están en la primera relación para las que no aparezcan tuplas asociadas a ellas en la segunda, como las tuplas que están en la segunda relación para las que no aparezcan tuplas asociadas a ellas en la primera</p>

- **Cálculo relacional:** es un lenguaje declarativo que permite construir expresiones equivalentes a las del álgebra relacional y es la base del lenguaje SQL. Elementos:
 - Condiciones de comparación: la comparación es un operador lógico binario que devuelve un resultado booleano (verdadero o falso).
 - Condiciones de pertenencia: verifica si una tupla pertenece o no a una relación.
 - Condiciones compuestas: composiciones de las condiciones anteriores.
 - Cuantificador existencial (\exists).
 - Cuantificador universal (\forall).
 - Expresión: permite obtener el conjunto de todas las tuplas que la verifiquen. Es el único que permite extraer información de la BD, el resto devuelven valores booleanos.

NORMALIZACIÓN DE BD

Normalización: proceso de rediseño de tablas en una BDR con el objetivo de minimizar la redundancia de datos.

- **Primera Forma Normal (1FN):** una relación está en 1FN cuando los valores que toman sus atributos son atómicos o no descomponibles.
- **Segunda Forma Normal (2FN):** una relación está en 2FN cuando está en 1FN y todos los atributos dependen de forma completa de la clave (depende de la clave y no de un subconjunto de sus atributos).

- **Tercera Forma Normal (3FN):** una relación está en 3FN cuando está en 2FN y ningún atributo secundario (aquellos que no forman parte de la clave) depende transitivamente de la clave.
 - Dependencia funcional transitiva: un atributo C tiene dependencia funcional transitiva de A, si B depende funcionalmente de A y C depende funcionalmente de B, pero A no depende funcionalmente de B.
- **Forma Normal de Boyce-Codd (FNBC):** una relación está en FNBC si está en 3FN y cada dependencia funcional no trivial ($A \rightarrow B$) tiene una clave candidata como determinante.
 - Consideremos una academia dónde un estudiante puede cursar diversas asignaturas y cada asignatura es impartida por un único profesor.
Tendríamos una tabla con las columnas: IDEstudiante, IDProfesor, IDAsignatura.
Las claves candidatas son (IDEstudiante, IDProfesor) y (IDEstudiante, IDAsignatura).
La relación está en 3FN porque no existen dependencias funcionales transitivas.
La relación no está en FNBC porque en la dependencia funcional no trivial $IDProfesor \rightarrow IDAsignatura$, el determinante (IDProfesor) no es una clave candidata.

Las anteriores conforman las formas normales básicas. Existen también las formas normales avanzadas (4FN, 5FN y 6FN), pero de menor uso; Métrica3 no contempla la normalización con FN avanzadas.

En ocasiones, tras haber normalizado la BBDD, se vuelven a desnormalizar parcialmente algunas tablas por razones de eficiencia o velocidad en las consultas (primero normalizar, después desnormalizar).

2. EL LENGUAJE SQL (ISO/IEC 9075 - SQL2)

El Lenguaje de Consulta Estructurado SQL es un lenguaje declarativo de acceso a BDR que permite realizar consultas a las BBDD, definir la estructura de los datos, modificarlos y especificar restricciones de integridad. Se compone de comandos, cláusulas, operadores y funciones de agregado.

TIPOS DE DATOS

Cadenas caracteres	CHAR (n)	Cadenas de caracteres de tamaño fijo (n) rellenas con el número de espacios necesario para llegar a n.
	VARCHAR (n)	Cadenas de caracteres de tamaño variable hasta un máximo de n.
	NCHAR (n)	Similar a CHAR (n) con soporte a alfabetización internacional.
	NVARCHAR (n)	Similar a VARCHAR (n) con soporte a alfabetización internacional.
Cadenas de bits	BIT (n)	Cadenas de n bits.
	BIT VARYING (n)	Cadenas con un máximo de n bits.
Números	INTEGER SMALLINT BIGINT	Para números enteros, de distinto tamaño máximo.
	FLOAT REAL DOUBLE PRECISION	Para números en coma flotante, con distintos tamaños para la parte entera y la decimal.
	NUMERIC (dig, dec) DECIMAL (dig, dec)	Números decimales con la precisión deseada. El parámetro 'dig' indica el total de dígitos disponibles (incluyendo la parte entera y la decimal). El parámetro 'dec' indica número de decimales.
Fecha y hora	DATE	Para fechas.
	TIME	Para horas. Normalmente se llega a una precisión de 100 ns.
	TIMETZ	TIME WITH TIME ZONE. Igual que TIME, pero incluye información sobre zona horaria.
	TIMESTAMP	Fecha y hora en un solo dato.

	TIMESTAMP TZ	TIMESTAMP WITH TIME ZONE. Igual que TIMESTAMP , pero incluye información sobre zona horaria.
--	---------------------	--

COMANDOS

- **DDL.** Permiten creación, modificación y borrado de los objetos de la BD

CREATE	Crear nuevos objetos de BD (tablas, índices...).
DROP	Eliminar objetos de la BD (elimina estructura, contenido y actualiza el diccionario).
TRUNCATE	Elimina todas las filas de una tabla de una sola vez. No permite ROLLBACK .
ALTER	Modificar las tablas agregando campos o cambiando la definición de los campos.

- **DML.** Permiten generar consultas para extraer datos de la BD y la creación, modificación y borrado de datos

SELECT	Consultar registros de la base de datos que satisfagan un criterio determinado.
INSERT	Inserción de elementos en una tabla ya creada: [INSERT INTO tabla VALUES (...)]
UPDATE	Modificar los valores de los campos y registros especificados: [UPDATE tabla SET campo1=x WHERE campo2=y]
DELETE	Eliminar registros específicos de una tabla de una base de datos: [DELETE FROM tabla WHERE campo=x]
MERGE	Combina datos de dos o más tablas en una sola tabla resultante utilizando un condición de combinación. Puede manejar INSERTs, UPDATEs y DELETEs en una sola transacción sin tener que escribir una lógica separada para cada una de ellas.

- **DCL.** Permiten asignar permisos sobre objetos de la BD a los usuarios

GRANT	Conceder permisos específicos usuarios: [GRANT operación ON nombre_objeto TO usuario]
REVOKE	Revocar privilegios ya concedidos a los usuarios: [REVOKE operación ON nombre_objeto FROM usuario]

- **TCL.** permiten controlar y gestionar transacciones para mantener la integridad de los datos.

BEGIN	Indica el inicio de una transacción
COMMIT	Los cambios realizados sobre la BD se hacen fijos únicamente al completar la transacción mediante la ejecución de COMMIT .
ROLLBACK	Elimina todos los cambios que se hayan producido en la BD desde la ejecución de la última instrucción COMMIT .

CLÁUSULAS

Condiciones de modificación utilizadas para definir los datos que se desea seleccionar o manipular.

ALL	Devuelve todos los campos de la tabla (la usada por defecto).
TOP	Devuelve un determinado número de registros de la tabla.
DISTINCT	Omite los registros cuyos campos seleccionados coincidan totalmente.
DISTINCTROW	Omite registros duplicados basándose en la totalidad del registro y no sólo en campos seleccionados.
FROM	Especificar la tabla de la cual se van a seleccionar los registros.
WHERE	Especificar las condiciones que deben reunir los registros que se van a seleccionar.
GROUP BY	Cuando se utilizan funciones de agregación, esta cláusula sirve para que se hagan sobre grupos específicos de filas en vez de sobre el total. Por ejemplo, en vez de realizar la media de todos mis usuarios, la puedo realizar agrupando por países, obteniendo un valor para cada país en vez de uno global.
HAVING	Expresar la condición que debe satisfacer cada grupo.
ORDER BY	Ordenar los registros seleccionados de acuerdo con un orden específico.
CONSTRAINT	Crear o eliminar índices.

OPERADORES

- Operadores lógicos**

AND	Evalúa dos condiciones y devuelve un valor de verdad sólo si ambas son ciertas.
OR	Evalúa dos condiciones y devuelve un valor de verdad si al menos una de las dos es cierta.
NOT	Devuelve el valor contrario de la expresión.

- Operadores de comparación**

<	Menor que	BETWEEN...AND	Especificar un intervalo de valores
>	Mayor que	LIKE	Comparación con un patrón (a LIKE '%rueba%')
<=	Menor o igual que	IN	Igual a uno de un conjunto de valores (a IN (3,4,5))
>=	Mayor o igual que	ANY	Verdadera si lo es para algún valor de los obtenidos en una subconsulta
<>	Distinto de	ALL	Es verdadera si lo es con todos los valores devueltos por la consulta subordinada y falsa en caso contrario
=	Igual que		
CASE	Expresiones condicionales	IS, IS NOT	Comparación con el valor nulo (a IS NULL; a IS NOT NULL)

- Operadores algebraicos (se usan con consultas)**

UNION	Unión de dos consultas diferentes, sin repetir filas.
UNION ALL	Unión de dos consultas diferentes, repitiendo filas.
INTERSECT	Devuelve los campos presentes en las dos tablas implicadas en la consulta.
EXCEPT	(o MINUS) Operación diferencia. Devuelve los campos presentes en la 1ª tabla que no estén en la 2ª.

FUNCIONES DE AGREGADO

Se usan dentro de una cláusula SELECT en grupos de registros para devolver un único valor que se aplica a un grupo de registros.

AVG	Calcular el promedio de los valores de un campo determinado	MAX	Devolver el valor más alto del campo especificado
COUNT	Devolver el número de registros de la selección, si se hace sobre campos específicos, no cuenta aquellos que sean nulos.	MIN	Devolver el valor más bajo del campo especificado
SUM	Devolver la suma de todos los valores del campo determinado		

RELACIÓN ENTRE SQL Y EL ALGEBRA RELACIONAL

OPERACIÓN ALGEBRA RELACIONAL	SENTENCIA SELECT ASOCIADA
SELECCIÓN $c(R)$	Select * from R where c
PROYECCION $a_1, a_2, \dots, a_n(R)$	Select a1, a2...an from R
PRODUCTO CARTESIANO $R \times S$	Select * from R, S
JOIN NATURAL $R \bowtie S$	Select * from R, S where $P_k(R) = F_k(S)$ Select * from R [inner] join S on $P_k(R) = F_k(S)$
UNION $R \cup S$	Select * from R union select * from S
INTERSECCION $R \cap S$	Select * from R intersect select * from S
DIFERENCIA $R - S$	Select * from R minus select * from S
COCIENTE R / S	NO EXISTE EN SQL, hay que hallarlo mediante otros operadores














3. IMPLEMENTACIONES COMERCIALES

Las Bases de datos relacionales se mantienen las primeras en el ranking de popularidad, pero han evolucionado a sistemas multimodelo, incorporando funcionalidades de bases de datos NoSQL. Existen modelos de bases de datos de uso general, como Oracle, SQL Server y MySQL/MariaDB, y bases de datos especializadas en nichos de mercado como IBM DB2.

En las bases de datos de uso general se le han incorporado posibilidades adicionales que inicialmente no tienen que ver con el mundo relacional. En 2020, en Oracle y SQL Server, podemos encontrar la posibilidad de:

- Tratar datos geográficos.
- Realizar índices y búsquedas en texto libre (Full Text Search).
- Machine Learning. Procesar los datos mediante scripts en Python o R.
- Datawarehousing. Posibilidad de generar cubos de análisis multidimensional (OLAP).
- Encriptar los datos de manera transparente.
- Capacidad de indexar y buscar en campos de texto en formato JSON y XML.
- Almacenar toda la BBDD en memoria creando índices especiales (In-Memory Databases).
- Almacenar los datos con una estrategia en columnas, en vez de filas, para optimizar lecturas.
- Vincular otras bases de datos y realizar consultas cruzadas.
- Big Data. Ejecutar consultas en Hadoop o Spark y almacenar o consultar los resultados.

414 systems in ranking, April 2023

Rank			DBMS	Database Model	Score		
Apr 2023	Mar 2023	Apr 2022			Apr 2023	Mar 2023	Apr 2022
1.	1.	1.	Oracle +	Relational, Multi-model 	1228.28	-33.01	-26.54
2.	2.	2.	MySQL +	Relational, Multi-model 	1157.78	-25.00	-46.38
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-model 	918.52	-3.49	-19.94
4.	4.	4.	PostgreSQL +	Relational, Multi-model 	608.41	-5.41	-6.05
5.	5.	5.	MongoDB +	Document, Multi-model 	441.90	-16.89	-41.48
6.	6.	6.	Redis +	Key-value, Multi-model 	173.55	+1.10	-4.05
7.	7.	8.	IBM Db2	Relational, Multi-model 	145.49	+2.57	-14.97
8.	8.	7.	Elasticsearch	Search engine, Multi-model 	141.08	+2.01	-19.76
9.	9.	10.	SQLite +	Relational	134.54	+0.72	+1.75
10.	10.	9.	Microsoft Access	Relational	131.37	-0.69	-11.41
11.	12.	11.	Cassandra +	Wide column	111.81	-1.98	-10.19
12.	11.	14.	Snowflake +	Relational	111.12	-3.27	+21.68
13.	13.	12.	MariaDB +	Relational, Multi-model 	95.93	-0.90	-14.38
14.	14.	13.	Splunk	Search engine	85.44	-2.54	-9.81
15.	16.	15.	Microsoft Azure SQL Database	Relational, Multi-model 	79.06	+1.62	-6.72
16.	15.	16.	Amazon DynamoDB +	Multi-model 	77.45	-3.32	-5.46
17.	17.	17.	Hive	Relational	71.65	+0.74	-9.77
18.	18.	18.	Teradata	Relational, Multi-model 	61.59	-2.14	-5.98
19.	19.		Databricks	Multi-model 	60.97	+0.11	
20.	21.	24.	Google BigQuery +	Relational	53.32	-0.12	+5.34

Fuente: <https://db-engines.com/en/ranking>

Es destacable la aparición de dos nuevas categorías de Bases de datos:

- NoSQL como MongoDB, ElasticSearch, Redis, Cassandra o Solr
- BBDD diseñadas para Cloud como Amazon DynamoDB o Azure SQL Database

4. LOS ESTÁNDARES ODBC Y JDBC

Para poder ejecutar consultas SQL es necesario disponer de un driver para conectar a la base de datos, para lo que se han empleado diferentes estrategias.

ODBC (OPEN DATABASE CONNECTIVITY)

Estándar de acceso a BD que proporciona un interfaz estándar a las aplicaciones para acceder a cualquier dato independientemente del gestor de BD donde resida. Para ello inserta una capa intermedia (CLI, Call-Level Interface) entre la aplicación y el SGBD, encargada de traducir las consultas de datos desde la aplicación a comandos para ese SGBD concreto.

Cada fabricante suele proporcionar un ODBC optimizado para su base de datos.

JAVA - JDBC (JAVA DATABASE CONNECTIVITY)

Es una API que permite conectar aplicaciones Java con un gestor de BD desde el lenguaje de programación Java, independientemente del sistema operativo o del SGBD utilizado. Tipos de drivers:

- **Tipo 1 (Puente JDBC-ODBC):** convierte el método JDBC a una llamada a una función ODBC. Utiliza los drivers ODBC para conectar con la BD.

- **Tipo 2 (Driver parcialmente nativo):** convierte el método JDBC a llamadas a una API nativa que debe estar en el cliente que accede a la BD.
- **Tipo 3 (JDBC middleware):** el cliente cuenta con un driver JDBC puro que conecta con un servidor intermedio encargado de traducir los métodos JDBC al lenguaje específico del SGBD.
- **Tipo 4 (Driver Java puro):** desarrollado en Java, utiliza directamente el protocolo de la BD. Ofrece mejor rendimiento, pero está ligado a un SGBD concreto.

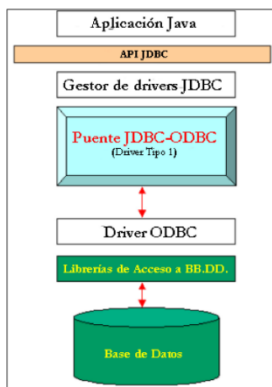


Figura 1. Driver JDBC Tipo 1

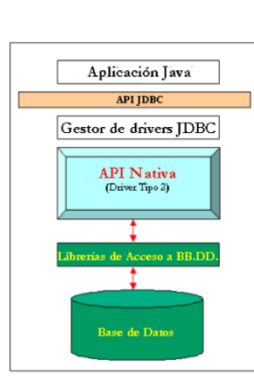


Figura 2. Driver JDBC Tipo 2

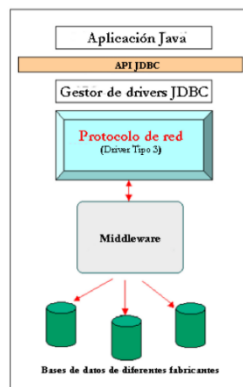


Figura 3. Driver JDBC Tipo 3

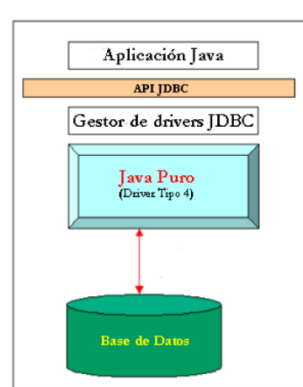


Figura 4. Driver JDBC Tipo 4

Aparte de JDBC, hay otras tecnologías más modernas de manejo de bases de datos relacionales desde Java como por ejemplo, el framework **JPA** (Java Persistence API). Su lenguaje de consulta es JPQL (Java Persistence Query Language).

También destaca **Hibernate** (software libre), herramienta para Java que facilita el mapeo de atributos entre una BD relacional y el modelo de objetos de una aplicación. Se trata de una implementación de JPA.

.NET

OLE DB (deprecated). Microsoft inicialmente ofreció una tecnología para conectarse a Bases de Datos basada en objetos COM que actualmente se considera superada con ADO.NET. Se mantiene por motivos de compatibilidad con aplicaciones desarrolladas.

La tecnología específica de Microsoft es ADO.NET que incorpora la posibilidad de cachear la información obtenida, es habitual que se integre con Entity Framework que es un mapeador objeto-relacional de Microsoft.

En el propio .NET Framework dispone de drivers específicos para conectarse a las bases de datos más habituales como Oracle o SQL Server. También se puede recurrir a los drivers del fabricante de base de datos.

Al igual que Hibernate para Java, existe **NHibernate** para .NET como herramienta para el mapeo de atributos entre una BD relacional y el modelo de objetos de una aplicación.

PHP

Con los PDO (PHP Data Object) dispone de drivers para las bases de datos más habituales. Si se desea una mayor eficiencia o aprovechar funcionalidades específicas sería necesario recurrir a los drivers del fabricante de la base de datos.

ORM (OBJECT-RELATIONAL MAPPING)

El mapeo objeto-relacional es una técnica de programación para convertir datos entre un lenguaje orientado a objetos y un lenguaje de base de datos relacional, apoyándose en un motor de persistencia. Existen varios ORM para las diferentes tecnologías:

- Java: Hibernate, iBatis, EJB, JDO
- .NET: Entity Framework, nHibernate
- PHP: Propel, Doctrine