## 1.4. El diseño conceptual y lógico de las bases de datos

Cuando se diseñan bases de datos es imprescindible acotar el contexto o universo del discurso en el que se integran. Para que los datos sean fieles se debe conceptualizar a través de ideas y definiciones que den soporte a este microuniverso. Esta representación se denomina **modelo conceptual**. Partiendo de este modelo se obtendrá una descripción de los datos compuesta por un conjunto de relaciones denominado **esquema relacional de datos**. Por último, este esquema se traducirá a estructuras físicas de almacenamiento.

Para llevar a cabo este procedimiento es necesario disponer de un conjunto de herramientas para diseñar los datos y sus relaciones y otras que permitan llevar las acciones con las bases de datos. La primera es el modelo de datos y la segunda, el **sistema gestor de base de datos**. Por tanto, es el **modelo de datos** el que permite el diseño de una base de datos. Los pasos generales en el diseño de una base de datos se pueden resumir en los siguientes:

- Describir en lenguaje natural los requisitos que debe cumplir la base de datos.
- Diseñar el diagrama conceptual. El diagrama entidad-relación o diagrama de Chen modeliza el problema mediante entidades asociadas por relaciones.
- Elegir el modelo de datos, por ejemplo, relacional.
- Transformar el diseño conceptual a un modelo relacional obteniendo así un conjunto de tablas relacionadas entre sí.
- Normalizar aquellos elementos que tienen defectos de diseño atendiendo a diferentes reglas.
- Optimizar la solución obtenida en fases anteriores según criterios de almacenamiento físico, espacio de disco, tiempo de acceso a datos, etcétera.

El modelo de datos más extendido es el denominado **entidad-relación**, introducido por Chen en 1976. Este modelo parte de una situación real definiendo los tipos de entidades y las interrelaciones entre ellos. El producto que se obtiene es un grafo denominado **diagrama entidad-relación**. Los elementos principales de este modelo son los tipos de entidades, los tipos de interrelaciones y los atributos.

### 1.4.1. Modelo entidad/relación

A través de esta técnica se representa y se define todos los datos que se introducen, alma- cenan, transforman y producen independientemente de la tecnología que se pretenda usar. El modelo de datos es el medio con el que comunicar el significado de los datos, las relaciones entre ellos y las reglas de negocio. Las ventajas de realizar un modelo de datos son:

- Control de los posibles errores.
- Elaboración de estructuras de datos independientes de la arquitectura física.
- Entendimiento de los datos.
- Perfeccionamiento del mantenimiento.

La técnica persigue la representación a través de un diagrama que se centra en los datos y que no dependen de cómo se transforman y sin detenerse en aspectos sobre la eficiencia.

El modelo entidad/relación con un alto nivel de abstracción describe cómo se distribuyen los datos. Los elementos principales de este modelo son las entidades y las propiedades. En la literatura sobre el modelo entidad/relación existen unos elementos añadidos que dan nombre al modelo entidad/relación extendido. Las extensiones al modelo básico añaden elementos como la jerarquía entre entidades.

Los elementos principales de este modelo son:

**Entidad**: es aquel objeto, real o abstracto, acerca del cual se desea almacenar información en la base de datos. La estructura genérica de un conjunto de entidades con las mismas características se denomina tipo de entidad. Existen dos clases de entidades: las regulares, que tienen existencia por sí mismas, y las débiles, cuya existencia depende de otra entidad. Las entidades deben tener existencia propia, cada ocurrencia de un tipo de entidad debe poder distinguirse de las demás y todas las ocurrencias de un tipo de entidad deben tener los mismos atributos.

**Dominio**: Es un conjunto de valores homogéneos al que se le da un nombre y tiene existencia propia independiente de cualquier entidad, relación o atributo.

**Atributo**: es una propiedad o característica de un tipo de entidad o de un tipo de interrelación y se define sobre un dominio. Cada tipo de entidad ha de tener un conjunto mínimo de atributos que identifican unívocamente cada ocurrencia del tipo de entidad. Este atributo o atributos se denomina identificador principal.

**Relación**: es una asociación o correspondencia existente entre una o varias entidades. La relación puede ser regular, si asocia tipos de entidad regulares, o débil, si asocia un tipo de entidad débil con un tipo de entidad regular. Dentro de las relaciones débiles se distinguen la dependencia en existencia y la dependencia en identificación. Se dice que la dependencia es en existencia cuando las ocurrencias de un tipo de entidad débil no pueden existir sin la ocurrencia de la entidad regular de la que dependen. Se dice que la dependencia es en identificación cuando, además de lo anterior, las ocurrencias del tipo de entidad débil no se pueden identificar sólo mediante sus propios atributos, sino que se les tiene que añadir el identificador de la ocurrencia de la entidad regular de la cual dependen. Una relación se caracteriza por su nombre único, el tipo de correspondencia y la cardinalidad. El tipo de correspondencia hace alusión al número máximo de ocurrencias de cada tipo de entidad que intervienen en una ocurrencia de la relación, y son 1:1, 1:N y NM.

**Cardinalidad**: representa el número de ocurrencias en cada uno de los tipos de entidad que participa en la interrelación y se indicará la mínima y la máxima.

**Generalización**: es un abstractor compuesto de un tipo de entidad llamado super- tipo que contiene todas las propiedades comunes de otro conjunto de tipos de entidad que forman los subtipos. El atributo discriminante del supertipo se usa para insertar las ocurrencias en los subtipos.

Existen muchas herramientas que ayudan a desarrollar esquemas conceptuales en bases de datos relacionales. Una de las herramientas más avanzadas es DataModeler, un producto de Oracle que puede usarse sin necesidad de emplear ningún sistema gestor de bases de datos, ni su producto

SQL Developer, que integra junto a este en su entorno de desarrollo. Es decir, en un ordenador se puede tener instalado Data Modeler con el fin de obtener los esquemas relacionales y otras potentes funciones que se verán más adelante.

## 1.4.2. La teoría de la normalización

La teoría de la normalización utiliza un conjunto de reglas que pretenden eliminar las posibles relaciones entre atributos que puedan provocar problemas en la actualización de los datos. Estas reglas se conocen como formas normales y son las siguientes:

**Primera forma normal**: se dice que un tipo de entidad está en primera forma normal si todos sus atributos solo toman un único valor del dominio de valores.

**Segunda forma normal**: un tipo de entidad está en segunda forma normal si está en primera y cualquier atributo que no forme parte de la clave primaria depende totalmente de todos los atributos que forman la clave primaria.

**Tercera forma normal**: un tipo de entidad está en tercera forma normal si está en segunda y los atributos dependen solo de la clave primaria.

## 1.4.3. La fase del diseño lógico

Una vez modelado conceptualmente el universo del discurso se obtiene un diagrama conceptual que representa los elementos de los que se quiere almacenar información, sus propiedades y cómo se relacionan entre ellos. En la siguiente fase de diseño, la etapa de diseño lógico, se deben transformar estos elementos en otros que están más cerca del paradigma en el que interesa trabajar, por ejemplo, en un paradigma relacional. Por tanto, cada uno de esos elementos deben transformarse para conseguir un conjunto de tablas compuestas por columnas y que se relacionan entre ellas.

Para desarrollar estas transformaciones se utilizará un conjunto de heurísticas que se convierten en reglas automáticas para cada uno de los elementos del diagrama entidad/ relación extendido. Por tanto, en esta fase se llevan a cabo, con el conocimiento de todas las reglas de transformación, cada uno de los cambios posibles.

De este modo, el diseño lógico es la etapa del proceso de diseño de una base de datos en la que se obtiene la representación de su estructura en términos de almacenamiento. El modelo entidad relación se usa para el modelado conceptual y el relacional para el lógico.

A continuación, se detallan las transformaciones más importantes.

### Transformación de entidades

Cada entidad se convierte en una tabla o relación en el modelo relacional. La tabla tendrá el mismo nombre que la entidad de la que proviene. La tabla se crea usando el comando CREATE TABLE.

Todos los SGBD implementan este comando. Las diferencias entre un SGBD y otro con respecto a los distintos tipos de datos deben ser analizadas.

## Transformación de dominios

Un dominio del modelo E/R se transforma en un dominio en el modelo relacional. No todos los SGBD implementan esta operación.

## Transformación de atributos

Cada atributo de una entidad se transforma en una columna de una tabla. Atendiendo al tipo de atributos se puede tener:

**Identificadores**: se transforman en una columna, que es la clave primaria de la tabla. En SQL, la condición de clave primaria se representará colocando la cláusula PRIMARY KEY al lado del nombre de la columna en la que se ha convertido el atributo (dentro de la sentencia CREATE TABLE con la que se crea la tabla en la que está incluida la columna).

**Identificadores alternativos**: se transforman en una columna a la que se le añade la restricción UNIQUE, lo cual significa que no puede haber valores repetidos en esa columna.

**Atributos no identificadores**: se transforman en una columna de la tabla.

**Atributos multivaluados**: en el modelo relacional una instancia de una relación solo toma un valor para cada atributo. Por ello, será obligatorio crear una nueva tabla que contenga a la clave primaria de la tabla anterior y al atributo multivaluado, siendo la clave primaria de la nueva tabla la concatenación de los dos atributos y marcándose la clave primaria de la tabla anterior como clave foránea en la nueva tabla.

## Transformación de las interrelaciones

Las interrelaciones del modelo E/R se transforman en nuevas estructuras en alguna tabla, ya sea como una nueva columna para relacionar las tablas que intervienen o la creación de una nueva tabla. Estas nuevas claves que adquieren las tablas tras resolver
una interrelación se llaman claves ajenas. La transformación va a depender de la multiplicidad de la interrelación:

**Interrelación N:M**: se convierte siempre en una tabla del modelo relacional cuya clave primaria, cláusula PRIMARY KEY, resulta de la concatenación de los atributos principales de los tipos de entidad que unía y además se comportan como claves ajenas. Para ello se usará la cláusula FOREIGN KEY.

**Interrelación 1:N**: se añade a la tabla con mayor cardinalidad las columnas que comportan su clave primaria. Estas nuevas columnas formarán una clave ajena hacia ella y se usará FOREIGN KEY.

También se puede convertir directamente en una tabla si tiene muchos atributos, se prevé que la relación es potencialmente del tipo N:M o la cardinalidad mínima es cero y se prevé un alto grado de instancias puestas a valores nulos.

**Interrelación 1:1**: se realizan del mismo modo que las 1:N, pero teniendo en consideración la cardinalidad mínima en ambos lados. Si son iguales, se puede propagar de cualquier tabla a la otra; sin embargo, si una es menor que la otra, se propaga de la de menor a la de mayor.

## Transformación de restricciones de entidades o atributos

Las restricciones que se hayan recogido sobre las entidades o los atributos se expresarán en el modo lógico como:

Una cláusula BETWEEN si la restricción es sobre un rango de valores.

Una cláusula IN si la restricción es sobre una lista de ocurrencias.

Una cláusula CHECK para que cumpla cierta condición, o un disparador si se nece- sita programar con múltiples condiciones.

## Transformación de interrelaciones por identificación

Se añade la clave primaria del tipo de entidad fuerte a la débil como parte de la clave primaria y como clave foránea hacia la fuerte.

## Transformación de generalizaciones

Se crea una tabla para el supertipo y otra tabla para cada subtipo. El valor discriminador se añade al supertipo y usando un disparador se incluirán sus instancias en los subtipos. Cada tabla subtipo hereda el campo clave del supertipo, que se convierte además en clave ajena que apunta al supertipo. Hay otras dos alternativas que se pueden considerar:

- Se transformará tanto el supertipo como los subtipos en una única tabla con la concatenación de todos los atributos. Esto se hará cuando no haya demasiadas propiedades en los subtipos y se puedan programar componentes denominados **disparadores** para el control de las ocurrencias de estas propiedades.
- Incluir en las tablas de los subtipos las propiedades del supertipo, es decir, se les añade a todos los subtipos las columnas del supertipo.

## Transformación de atributos derivados

Los atributos derivados se transformarán en columnas de la entidad a la que pertenezcan (como el resto de los atributos). Además, se establecerá un disparador o un procedimiento almacenado que calcule el valor del atributo cada vez que se inserta una nueva fila en la tabla o cada vez que se modifique en una fila el valor de alguno de los atributos a partir de los cuales se calcula el atributo derivado.

**Normalización del esquema obtenido**

En los esquemas relacionales que se obtienen, se aplican las heurísticas necesarias para obtener un esquema relacional ya normalizado, evitándose así las anomalías de inserción, modificación y borrado que provoca la redundancia.

## 1.5. La etapa del diseño físico

En esta etapa, la última en el proceso de desarrollo de la base de datos, se pretende conseguir un esquema interno creado a través del sistema gestor de bases de datos elegido. Este esquema interno debe funcionar según los requerimientos que los usuarios plantearon en la etapa de análisis. Además de cumplir con estos requisitos, el diseño físico contemplará los mecanismos que permitan disminuir tanto el tiempo de respuesta como el espacio de almacenamiento usado. Los procesos asociados con la seguridad también son aspectos asociados al diseño físico.

En esta fase es necesario conocer los recursos hardware y software con los que se cuenta, además del esquema lógico obtenido en la etapa anterior. Se estudiarán las políticas de seguridad de los datos, el conjunto de aplicaciones que interactúan con la base de datos y las transacciones que esta interacción generará.

En este proceso hay que usar un conjunto de heurísticas que posibiliten optimizar el uso de los recursos y los procesos de accesos a los datos y su seguridad. Estas heurísticas o reglas de aproximación son dependientes del SGBD y se buscará la definición de las estructuras de archivos, la indexación de tablas, el uso del espacio de memoria principal, los roles de seguridad y los objetos de gestión.

Estos pasos serán testeados y puestos a evaluación hasta su refinación. Este proceso de mejora se conoce como **ajuste de la base de datos o proceso tunning**. El tunning o afinamiento es el proceso sistemático de búsqueda de origen, naturaleza y corrección de los problemas de rendimiento de un sistema de bases de datos. Se usarán diferentes comandos del SGBD para llevar a cabo estos objetivos.

El diseño físico de las consultas es una tarea importante en este proceso. Las consultas acceden a un conjunto de tablas para hacer operaciones de interés. Una consulta está compuesta por la selección de columnas que proyectar, cláusula SELECT, un conjunto de tablas combinadas por algunas columnas, cláusula FROM y unas condiciones que cumplir sujetas a las posibles columnas de las tablas seleccionadas. Aunque una consulta tiene más opciones, la sintaxis de esta es:

*SELECT columnas FROM tablas WHERE condición;*

No es posible proyectar columnas de tablas que no están indicadas en la sección FROM, del mismo modo que no se puede indicar columnas para llevar a cabo el filtrado si no están en algunas de las tablas tras la cláusula FROM.

## 1.5.1. Diseño de la representación física

Esta fase comienza con el análisis de las transacciones. Para ello es necesario conocer las consultas que se pretenden elaborar y sus transacciones. Para cada transacción se debe conocer las columnas y tablas a las que se quiere acceder y la operación que realizar: alta, baja o modificación. Las estructuras de accesos que crear dependen en gran medida de los campos que intervienen en la cláusula WHERE de una consulta SQL. Las columnas comunes que intervienen en la unión de las tablas de una consulta también son candidatas para construir estructuras de acceso.

El rendimiento de los accesos a los datos también son dependientes de las estructuras físicas construidas en los soportes de almacenamiento ultra masivos. El mecanismo de gestión más común para estas estructuras son los ficheros o árboles binarios.

Los índices son unas estructuras de datos compuestas por los identificadores únicos de las filas de una tabla acelerando los accesos a ella. Todas las tablas, cuando son creadas, son indexadas a través de las columnas que comporta la clave primaria. Esto se hace principalmente para acelerar las consultas, ya que estas hacen cruces de los campos clave de las tablas que intervienen en ellas. Además, se pueden añadir otros índices a las tablas que usen otras columnas que son frecuentemente usadas en las consultas diseñadas. Estos índices afectan a los tiempos de ejecución de las operaciones de inserción y eliminación de registros en las tablas. Además, el espacio que la tabla usa aumenta. Por lo tanto, es necesario que se ponderen las ventajas y desventajas del uso de índices. El comando para crear un índice es CREATE INDEX.

## 1.5.2. Gestión del espacio de almacenamiento

La gestión del espacio de almacenamiento de las tablas en los dispositivos de almacenamiento locales, en red o en la nube son tareas muy importantes. Si se usan arquitecturas locales, la administración recae en el propio sistema de información, de tal modo que los profesionales especializados en la administración de los sistemas físicos deben planificar con mucho detalle el uso de estos sistemas de almacenamiento. La comunicación entre los elementos de almacenamiento tales como sistemas RAID, arquitectura NAS Y SAN, entre otros, es tarea fundamental para tener accesos óptimos a los datos. Si los sistemas de almacenamiento son remotos, deben resolverse nuevas cuestiones como la comunicación, seguridad, mantenimiento y operatividad.

La gestión de espacios de tablas permite en Oracle organizar de forma lógica cada uno de los archivos de datos, en los que es posible indicar el espacio que en un inicio ocupará la tabla creada en ella. Se estudiará el espacio inicial de cada tabla, cómo crece a diario y, si es posible, analizar cuál será su tamaño máximo, qué espacio ocupará en un determinado momento y cómo gestionar estos espacios de crecimiento. Existirán tablas que crecen a gran velocidad y que ocupan poco o mucho espacio con respecto a la totalidad del espacio que requiere la base de datos. Acciones útiles para esto son la creación de espacios de tablas, asignación de espacio inicial, gestión de las extensio- nes de crecimiento dentro de un archivo de datos, etc. Los objetos más importantes que se usan para esta gestión son los espacios de tabla, TABLESPACE, los esquemas de almacenamiento y archivos de datos, DATAFILE, y las extensiones y segmentos.

## 1.5.3. Procesos de optimización en el nivel físico

Una técnica que optimiza los accesos es la agrupación de tablas, o también llamada **clustering**. Este proceso tiene como fin agrupar tablas a las que se accede en las mismas operaciones, organizándose de forma física en áreas cercanas en los soportes para así acelerar los accesos a los datos que contienen.

Por motivos de implementación, en la etapa física se pueden tomar decisiones sobre la redundancia de datos con el fin de aumentar la eficiencia temporal en las acciones de accesos a datos. La redundancia de datos evita buscar información extra en tablas gigantes, lo que aumentaría el tiempo en las operaciones asociadas a estas tablas. Para evitar esos accesos, los datos que se requieren pueden estar incluidos en las tablas iniciales. Este proceso se puede llamar **desnormalización de tablas**. La teoría de la normalización busca tablas eficientes espacialmente, pero, a veces, se pueden hacer
excepciones en busca de esta eficiencia temporal. Un ejemplo podría ser almacenar el precio de coste y venta actual de un producto, aunque este esté incluido en la tabla Precio que almacena los diferentes valores que toma un producto en cuestión.

Esta técnica hace que las actualizaciones de los datos sean más complicadas, ya que el mismo valor está en más de un lugar. Se usarán técnicas de programación para llevar el control de estos valores. Esto ralentiza estas acciones y hace que el esquema lógico pierda flexibilidad. Por ello, es importante poner en una balanza los aspectos que se desean mejorar.

## 1.5.4. Diseño de mecanismos de seguridad de los datos y de monitorización.

Los usuarios de la base de datos accederán de la totalidad de esta solo a aquella parte que se considere necesaria, ni más ni menos, con el fin de aumentar su seguridad. Esta visión parcial de todo el esquema se consigue a través de un elemento denominado **vista**. Las vistas son mecanismos que permiten construir la visión que tiene un usuario de toda la base de datos, es decir, su esquema lógico. Junto a esto, existirán unas reglas que definan el modo en que un usuario puede explotar la arquitectura, los roles, y el esquema lógico, los perfiles.

Para cada usuario se establecen roles para la arquitectura y perfiles con privilegios para cada fila, objeto, columna, tabla, espacio de tabla, archivo de datos, etc., que se considere necesario.

La **monitorización de bases de datos** consiste en realizar un seguimiento proactivo del estado y del rendimiento minimizando el tiempo de inactividad y ejecutando procesos que permitan solucionar problemas con esta supervisión. Los fines que se persiguen más importantes son:

- Acción completa en tiempo real.
- Administración de las alertas.
- Análisis de la ejecución de los comandos SQL.
- Análisis de las transacciones en la capa de aplicación.
- Gestión de la automatización de datos y generación de informes automáticos.

- Gestión de microservicios en la nube.
- Hacer un seguimiento de la actividad analizando las causas de un problema.
- Métrica y análisis integral.

El diseño físico no se debe concebir como un proceso secuencial que finaliza cuando se ha encontrado una configuración válida. Las bases de datos pueden sufrir variaciones a lo largo del tiempo, tanto en su tamaño como en su esquema lógico global o en el uso que se desee hacer de ellas, lo que obligará a una monitorización continua de su rendimiento (para comprobar si se va degradando) y a la introducción de ajustes que permitan adaptarse a los cambios sufridos por la base de datos o solventar las pérdidas de eficiencia que se hayan producido.

Cada proceso de ajuste supone en cierta medida una reconstrucción del diseño físico. que se haya realizado, por lo cual esta etapa de diseño volvería a comenzar de nuevo.