Guía de Estudio Ampliada: Diseño de Bases de Datos - Modelo Conceptual

1. ¿Qué es un Modelo Conceptual?

El modelo conceptual representa el primer paso en el diseño de una base de datos. Se trata de una representación abstracta y de alto nivel de los datos que un sistema necesita almacenar. El objetivo principal es describir la información sin considerar aún los aspectos técnicos ni el lenguaje de base de datos que se utilizará más adelante.

Históricamente, el modelado conceptual aparece en los años 70, cuando las organizaciones comenzaron a gestionar grandes volúmenes de datos. Peter Chen propuso el Modelo Entidad-Relación (MER) en 1976, permitiendo representar entidades y relaciones de manera sistemática.

El modelo conceptual facilita la comunicación entre los expertos del negocio y los diseñadores del sistema. Permite validar requerimientos, descubrir errores tempranamente y establecer las reglas que regirán la integridad de los datos.

Ejemplo: En una universidad, se modelan entidades como “Estudiante”, “Materia”, “Profesor”, y “Inscripción”. Aún no se decide si se usará MySQL, Oracle o cualquier otro sistema; solo se define qué información es relevante y cómo se relaciona.

2. Sistema de Gestión de Base de Datos (DBMS)

Un DBMS es el sistema de software que permite la creación, manipulación y administración de bases de datos. Su función principal es actuar como intermediario entre los usuarios (o aplicaciones) y la base de datos.

Los DBMS aparecieron en los años 60-70 con sistemas jerárquicos (IMS de IBM). Luego surgieron los sistemas de modelo relacional, como Oracle (1979), PostgreSQL (1986) y MySQL (1995), basados en teoría matemática y lógica formal.

Funciones clave:

- Definición de estructuras (tablas, índices, restricciones).
- Manipulación de datos (inserción, consulta, modificación, eliminación).
- Seguridad y control de acceso.
- Respaldo y recuperación.

Ejemplos populares:

- **PostgreSQL:** avanzado y robusto, de código abierto.
- **MySQL:** ligero, rápido, ideal para aplicaciones web.
- **SQLite:** embebido, usado en dispositivos móviles.
- **SQL Server:** integración completa con entornos Windows.
- **Oracle:** orientado a grandes corporaciones y rendimiento masivo.

3. Entidades

Las entidades representan cosas u objetos del mundo real que tienen existencia independiente. Pueden ser personas, conceptos, lugares o eventos.

En el contexto del modelado, una entidad debe poder identificarse de manera única. Se representa generalmente con un rectángulo en los diagramas entidad-relación.

Tipos:

- **Entidades físicas:** Estudiante, Producto, Vehículo.
- **Entidades conceptuales:** Curso, Categoría, Departamento.

Una entidad se describe mediante sus atributos, y su comportamiento dentro del sistema se determina a través de las relaciones.

Ejemplo: En un sistema de biblioteca:

- Entidades: Libro, Socio, Préstamo.

Cada entidad permite estructurar la base de datos con lógica y orden, facilitando su manipulación posterior.

---

4. Atributos

Los atributos son las características o propiedades que describen a una entidad. Cada entidad puede tener uno o varios atributos.

Clasificación por estructura:

- **Simples:** indivisibles (ej. DNI, precio).
- **Compuestos:** pueden descomponerse (ej. Dirección = calle, número, ciudad).

Por cardinalidad:

- **Monovaluados:** un solo valor por entidad (ej. fecha de nacimiento).
- **Multivaluados:** múltiples valores posibles (ej. teléfonos, correos).

Ejemplo práctico: Un estudiante tiene:

- Nombre (simple)
- Dirección (compuesta)
- Correos electrónicos (multivaluado)

La correcta definición de los atributos permite normalizar adecuadamente la base de datos.

---

5. Relaciones

Las relaciones representan asociaciones entre entidades. Son esenciales para expresar interacciones entre datos.

Tipos:

- **1:1 (uno a uno):** Un jefe dirige un solo departamento.
- **1**:N**** (uno a muchos):** Un cliente puede realizar muchos pedidos.
- **N**:M**** (muchos a muchos):** Un estudiante cursa muchas materias y cada materia es cursada por muchos estudiantes.

Las relaciones se representan como rombos en los DER. En sistemas reales, una relación N:M se implementa mediante una tabla intermedia.

Ejemplo: Entidad A: Estudiante Entidad B: Materia Relación: Inscripción (N:M) → tabla intermedia con atributos como fecha de inscripción, nota, etc.

Comprender la naturaleza de las relaciones es clave para una modelación eficiente y coherente.

---

6. Modelo Entidad-Relación (MER) y Diagrama Entidad-Relación (DER)

MER: Es un modelo teórico que describe la estructura de los datos desde un punto de vista conceptual, sin implementar todavía una base concreta. Establece las reglas de negocio, las relaciones y restricciones entre entidades.

DER: Es la representación gráfica del MER. Permite visualizar de manera clara y directa cómo está estructurada la información. Es una herramienta de comunicación y validación del diseño.

Simbología estándar:

- Rectángulos: entidades
- Elipses: atributos
- Rombos: relaciones
- Líneas: asociaciones entre ellos

Herramientas: Draw.io, Lucidchart, DBDesigner, MySQL Workbench.

Un DER bien hecho es esencial para la transición al modelo lógico relacional.

---

7. Restricciones de Integridad Referencial

La integridad referencial asegura que las relaciones entre datos en distintas tablas sean válidas y coherentes.

Conceptos clave:

- **Clave primaria:** identificador único de cada fila.
- **Clave foránea:** atributo que apunta a la clave primaria de otra tabla.

Si se elimina una fila con clave primaria referenciada, se corre el riesgo de dejar "datos huérfanos". Por eso, los DBMS permiten:

- **RESTRICT:** impide el borrado.
- **CASCADE:** borra también las filas relacionadas.
- **SET NULL:** pone el valor de la foránea en NULL.

Ejemplo en SQL:

```
FOREIGN KEY (cliente_id) REFERENCES clientes(id)
```

Esto garantiza que no haya ventas con clientes inexistentes.

---

8. Claves Primarias y Candidatas

Clave primaria: identificador único e irrepetible para cada fila de una tabla. No acepta valores nulos.

Claves candidatas: son atributos que podrían cumplir ese rol, pero solo una es seleccionada como primaria.

Criterios para elegir:

- Inmutabilidad (que no cambie con el tiempo).
- Brevedad y simplicidad.
- Evitar datos personales (nombre, email).

Ejemplo: En una tabla de empleados:

- Candidatos: legajo, DNI, email.
- Clave primaria seleccionada: legajo.

Seleccionar la clave correcta es crítico para la consistencia futura de la base.

---

9. Claves Foráneas

Una **clave foránea** (foreign key) es un campo que crea una relación entre tablas. Referencia a una clave primaria en otra tabla.

Ventajas:

- Mantenimiento de integridad referencial.
- Evita inconsistencias.
- Facilita las operaciones JOIN en SQL.

Ejemplo:

```
CREATE TABLE prestamos (
  id INT PRIMARY KEY,
  id_libro INT,
  FOREIGN KEY (id_libro) REFERENCES libros(id)
);
```

Este código impide crear un préstamo si no existe el libro.

Una base relacional correctamente normalizada siempre utiliza claves foráneas para representar relaciones.

---

10. Conclusión General

El modelo conceptual permite pensar primero en los datos, sus relaciones y restricciones, antes de preocuparse por la tecnología. Es esencial para el diseño de una base de datos bien estructurada, normalizada, eficiente y escalable.

Comprender los principios del MER, identificar correctamente las entidades, sus atributos, y las relaciones, así como manejar claves primarias y foráneas, es fundamental para cualquier analista, diseñador o desarrollador que trabaje con datos.

Invertir tiempo en el diseño conceptual evita problemas en etapas posteriores del desarrollo y mantenimiento de un sistema.