Diseño físico

Diseño de Bases de Datos
Grado en Ingeniería Informática

Departamento de
Lenguajes y Sistemas Informáticos

Universitat d'Alacant
Universidad de Alicante

Diseño de una BD

Diseño conceptual

↓

Diseño lógico

↓

Diseño físico

Objetivo del diseño físico

En el diseño físico debe conseguir definir las estructuras de almacenamiento y las estructuras de acceso, *de entre las que permita el SGBD elegido*, más adecuadas al uso que se le dará a los datos, para que las aplicaciones que accedan a la BD obtengan un buen rendimiento.

Muy dependiente del SGBD con el que se trabaje

Fases del diseño físico

1. Traducir el esquema lógico para el SGBD específico.
2. Diseñar la representación física.
3. Diseñar los mecanismos de seguridad.
4. Monitorizar y afinar el sistema.

1. Traducir e. lógico para el SGBD específico.

- **1.1. Diseñar las relaciones base para el SGBD específico.**
- **1.2. Diseñar las reglas de negocio para el SGBD específico.**

1. Traducir e. lógico para el SGBD específico.

Debemos conocer los siguientes aspectos:

- **Si el sistema permite la definición de dominios al usuario.**
- **Si el sistema permite la definición de claves primarias, claves ajenas y claves alternativas.**
- **Si el sistema permite definir columnas de valor no nulo.**
- **Si el sistema soporta la definición de reglas de negocio.**

1. Traducir e. lógico para el SGBD específico.

1.1. Diseñar las relaciones base para el SGBD específico

Partiendo del esquema lógico, se definirá para cada una de las relaciones:

- **su nombre,**
- **sus columnas, indicando para cada una su tipo de datos, si admite nulos y, si tiene valor por defecto y restricciones,**
- **la clave primaria,**
- **las claves ajenas, si las tiene junto con sus reglas de integridad**

1.2. Diseñar las reglas de negocio para el SGBD específico

- **Algunas se incorporan con el uso de CHECK en CREATE TABLE,**

En prácticas: el tipo de habitación del hotel sólo admite los valores I, D, DT, S

- **otras a través de disparadores**

En prácticas: las actividades del hotel si son de NIÑOS no son de ADULTO ni para TODOS los públicos, si son de ADULTO no son …

2. Diseñar la representación física.

Uno de los objetivos principales del diseño físico es almacenar los datos de modo eficiente.

¿Qué factores se tienen en cuenta para medir la eficiencia?

- **Productividad de transacciones.** Maximizar

  Número medio de transacciones que el SGBD puede procesar por minuto. Parámetro crítico en reservas de vuelos, servicios bancarios.

- **Tiempo de respuesta.** Minimizar

  Tiempo entre que se inicia una transacción y se finaliza o se obtiene una respuesta.

- **Espacio en disco** Optimizar

  Cantidad de espacio ocupado por los ficheros de la BD y sus estructuras de acceso.

2. Diseñar la representación física.

Buscar un equilibrio a la hora de satisfacer esos 3 factores.

2.1. Analizar las transacciones

- La frecuencia con que se va a ejecutar.
- Las relaciones y los atributos a los que accede la transacción, y el tipo de acceso: consulta, inserción, modificación o eliminación. Los atributos que se modifican no son buenos candidatos para construir estructuras de acceso.
- Los atributos que se utilizan en los predicados del WHERE de las sentencias SQL. Estos atributos pueden ser candidatos para construir estructuras de acceso dependiendo del tipo de predicado que se utilice.
- Si es una consulta, los atributos involucrados en el join de dos o más relaciones. Estos atributos pueden ser candidatos para construir estructuras de acceso.

2. Diseñar la representación física.

2.2. Escoger sistemas de acceso

- **Secuencial → Es necesario recorrer toda la tabla para extraer información (menos eficiente, evitarlo en lo posible).**
- **hash → Acceso directo, permite acceso a través de búsquedas de valores exactos.**
- **Índices (arboles-B, BitMap, basados en funciones)→ permite búsquedas por rango o por patrón.**

2.3. Escoger los índices primarios y secundarios

- Construir un índice sobre la clave primaria de cada relación base (lo hacen los SGBD por defecto al definir una primary key).
- No crear índices sobre relaciones pequeñas.
- Añadir un índice sobre los atributos que se utilizan para acceder con mucha frecuencia (tablas grandes).
- Añadir un índice sobre las claves ajenas que se utilicen con frecuencia para hacer joins.
- Evitar los índices sobre atributos que se modifican a menudo.
- Evitar los índices sobre atributos poco selectivos.
- Evitar los índices sobre atributos formados por tiras de caracteres largas.

2.4. Considerar la introducción de algunas modificaciones en el diseño lógico obtenido en 3FN

Se considera
esquema lógico adecuado→ 3 FN

La **normalización** pretende separar atributos relacionados lógicamente en varias tablas para **minimizar redundancia** y evitar anomalías de actualización (procesos de actualización más costosos).

Si nuestro diseño conceptual es adecuado el esquema lógico resultante ya está directamente en 3FN

2. Diseñar la representación física.

2.4.1 Considerar la introducción de redundancias controladas

Esquema lógico → ~~3 FN~~

Desnormalizar para mejorar tiempo de respuesta de consultas
(consultas complejas con gran número de uniones)

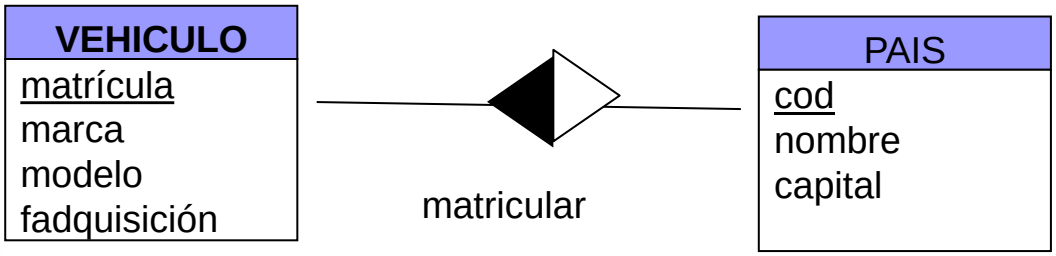
¡¡CUIDADO!! – NO RECOMENDABLE
SOLO COMO ÚLTIMO RECURSO

- Ralentiza actualizaciones
- Implementaciones más complejas
- Menos flexibilidad

2. Diseñar la representación física.

2.4.1 Considerar la introducción de redundancias controladas (situaciones candidatas)

- **En tablas que participen en consultas críticas que causen problemas de rendimiento, se puede considerar la introducción de la descripción junto con el código en la relación hijo, manteniendo la tabla de referencia para validación de datos siempre que esos atributo no sufran modificaciones frecuentes**



VEHICULO(matrícula, marca, modelo, fadquisicion, país_cod, país_nombre)
C.P. matrícula
Y controlar con un disparador posibles cambios en el nombre de un país (caso poco frecuente)

2. Diseñar la representación física.

2.4.1 Considerar la introducción de redundancias controladas (situaciones candidatas)

- **Introducción de atributos calculados y mantenerlos mediante disparadores**

Ejemplo:

- FACTURA (número, fecha, cliente, importeTotal)
  - C.P.: número
  - C. ajena: cliente→ cliente
- LÍNEA(numlin, numfact, articulo, cant, precio)
  - C.P.: (numlin, numfact)
  - C. Ajena: numfact → factura
  - C. Ajena: articulo → articulo

importeTotal: Campo que no aparecía en el modelo pero tenerlo siempre disponible y actualizado con disparadores agiliza mucho el rendimiento

2. Diseñar la representación física.

```
CREATE TRIGGER ValorTotalFact
AFTER insert or delete or update on LINEA
FOR EACH ROW
Begin
IF inserting then
UPDATE factura
SET
importeTotal=importeTotal+(:new.cant*:new.precio)
WHERE numero=:new.numFact;
End if;
```

(sigue)

2. Diseñar la representación física.

```
IF deleting then
UPDATE factura
SET importeTotal=importeTotal-(:old.cant*:old.precio)
WHERE numero=:old.numFact;
End if;
```

(sigue)

2. Diseñar la representación física.

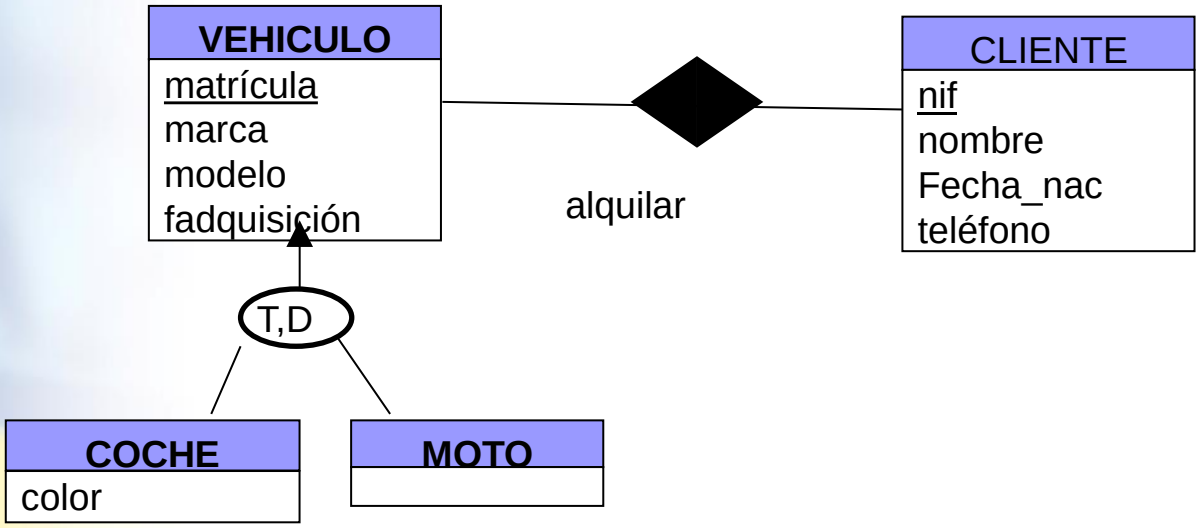
```
IF updating then
UPDATE factura
SET importeTotal=importeTotal+(:new.cant*:new.precio)
WHERE numero=:new.numFact;

UPDATE factura
SET importeTotal=importeTotal-(:old.cant*:old.precio)
WHERE numero=:old.numFact;
End if;
END;
```

2. Diseñar la representación física.

2.4.2 Considerar el tratamiento de generalizaciones como una única tabla donde se agrupa objeto general y subtipos

Esta solución se utiliza cuando el número de atributos de los subtipos es pequeño y las relaciones que los asocian con el resto de entidades en el esquema conceptual parten casi todas del objeto general.

2. Diseñar la representación física.



VEHICULO(matrícula, marca, modelo, fadquisicion, tipo, color)

C.P. matrícula

Y controlar con disparadores los valores

2. Diseñar la representación física.

2.4.3 Considerar el uso de vistas materializadas

Esta solución se utiliza cuando la consulta es altamente compleja y los datos de los que depende no se modifican con frecuencia y es aceptado que el sistema ofrezca información refrescada cada cierto tiempo (horas/cada noche, etc.)

Por ejemplo podría valer para una consulta compleja sobre un “árbol genealógico” pero no para un sistema bancario.

2.5. Estimar la necesidad de espacio

- **Esta estimación depende del SGBD que se vaya a utilizar y del hardware.**

- **Se debe estimar el número de tuplas de cada relación y su tamaño, así como el factor de crecimiento de cada relación.**

3. Diseñar los mecanismos de seguridad.

3.1. Diseñar las vistas de los usuarios

Diseñar las vistas de los usuarios que corresponden a los esquemas lógicos locales. Las vistas, además de preservar la seguridad, mejoran la independencia de datos, reducen la complejidad y permiten que los usuarios vean la información en el formato deseado.

3.2. Diseñar las reglas de acceso

- **Identificador para el acceso**
- **Permisos para realizar determinadas operaciones.**

Próximo tema

4. Monitorizar y afinar el sistema.

- **Una vez implementado el esquema físico de la base de datos, se pone en marcha, se observan sus prestaciones y pueden ser necesarios cambios.**
- **No es estático, ante nuevos requerimientos de los usuarios puede ser necesario realizar cambios.**

4. Monitorizar y afinar el sistema.

Puede ser necesario ajustar índices

Algunas consultas tardan mucho en ejecutarse

Algunas índices no se usan ORACLE: EXPLAIN PLAN

Puede ser necesario ajustar el diseño del esquema

Fragmentación Vertical de tablas

almacenar una tabla en varias, cada una con un grupo de atributos que suelen se accedidos de forma conjunta

INVESTIGADOR(dni, nombre, fnac, dirección, cuenta_banco, tema_especialista, grupo_investigación, nivel_investigador)

DATOS_PERSONALES(dni, nombre, fnac, dirección, cuenta_banco)

DATOS_INVESTIGADOR(dni, tema_especialista, grupo_investigación, nivel_investigador)

4. Monitorizar y afinar el sistema.

Puede ser necesario ajustar índices

Algunas consultas tardan mucho en ejecutarse

Algunas índices no se usan ORACLE: EXPLAIN PLAN

Puede ser necesario ajustar el diseño del esquema

Fragmentación Vertical de tablas

almacenar una tabla en varias, cada una con un grupo de atributos que suelen se accedidos de forma conjunta

Fragmentación Horizontal de tablas

Si el número de filas es elevado se pueden dividir en varias tablas en función de algún criterio

En Oracle por ejemplo

CREATE TABLE empleado(nif, ……, fecha_alta, …)
PARTITION BY …
Se puede particionar por rango, intervalo, hash

4. Monitorizar y afinar el sistema.

Puede ser necesario ajustar índices

Algunas consultas tardan mucho en ejecutarse

Algunas índices no se usan ORACLE: EXPLAIN PLAN

Puede ser necesario ajustar el diseño del esquema

Fragmentación Vertical de tablas

almacenar una tabla en varias, cada una con un grupo de atributos que suelen se accedidos de forma conjunta

Fragmentación Horizontal de tablas

Si el número de filas es elevado se pueden dividir en varias tablas en función de algún criterio

Puede ser necesario ajustar las consultas

- Evitar DISTINCT redundantes (pueden suponer reordenamiento)
- Se pueden usar tablas temporales para evitar subconsultas correlacionadas
- Evitar comparaciones de cadenas
- …

Diseño físico

Diseño de Bases de Datos
Grado en Ingeniería Informática

Departamento de
Lenguajes y Sistemas Informáticos

Universitat d'Alacant
Universidad de Alicante