Temario de oposiciones

INFORMÁTICA II

Alexandre Dapena
J. Luis Villanueva
Eladio Rial

educàlia editorial

Temario de oposiciones de

INFORMÁTICA II

Alexandre Dapena
J. Luis Villanueva
Eladio Rial

educàlia
editorial

Última edición 2021

Autores: Alexandre Dapena, J. Luis Villanueva y Eladio Rial

Edita: Educàlia Editorial

Imprime: Grupo Digital 82, S.L.

ISBN: 978-84-947891-1-3

Depósito legal: V-3126-2017

Printed in Spain/Impreso en España.

Todos los derechos reservados. No está permitida la reimpresión de ninguna parte de este libro, ni de imágenes ni de texto, ni tampoco su reproducción, ni utilización, en cualquier forma o por cualquier medio, bien sea electrónico, mecánico o de otro modo, tanto conocida como los que puedan inventarse, incluyendo el fotocopiado o grabación, ni está permitido almacenarlo en un sistema de información y recuperación, sin el permiso anticipado y por escrito del editor.

Alguna de las imágenes que incluye este libro son reproducciones que se han realizado acogiéndose al derecho de cita que aparece en el artículo 32 de la Ley 22/18987, del 11 de noviembre, de la Propiedad intelectual. Educàlia Editorial agradece a todas las instituciones, tanto públicas como privadas, citadas en estas páginas, su colaboración y pide disculpas por la posible omisión involuntaria de algunas de ellas.

Educàlia Editorial, S.L.

Avda. de las Jacarandas, 2, loft 327 - 46100 Burjassot

Tel: 960 624 309 - 963 768 542 - 610 900 111

E-mail: educaliaeditorial@e-ducalia.com

www.e-ducalia.com

ÍNDICE

Tema 38. Modelo de datos relacional. Estructuras. Operaciones. Álgebra relacional.

Tema 39. Lenguajes para la definición y manipulación de datos en sistemas de base de datos relacionales. Tipos. Características. Lenguaje SQL.

Tema 40. Diseño de bases de datos relacionales.

Tema 41. Utilidades de los sistemas gestores de base de datos para el desarrollo de aplicaciones. Tipos. Características.

Tema 42. Sistemas de base de datos distribuidos.

Tema 43. Administración de sistemas de base de datos.

Tema 44. Técnicas y procedimientos para la seguridad de los datos.

Tema 45. Sistemas de información. Tipos. Características. Sistemas de información en la empresa.

Tema 46. Aplicaciones informáticas de propósito general y para la gestión empresarial. Tipos. Funciones. Características.

Tema 47. Instalación y explotación de aplicaciones informáticas. Compartición de datos.

Tema 48. Ingeniería del software. Ciclo de desarrollo del software. Tipos de ciclos de desarrollo. Metodologías de desarrollo. Características distintivas de las principales metodologías de desarrollo utilizadas en la Unión Europea.

Tema 49. Análisis de sistemas: Modelización de tratamientos. Modelo de flujo de datos y control. Técnicas descriptivas. Documentación.

Tema 50. Análisis de sistemas: Modelización conceptual de datos. Técnicas descriptivas. Documentación.

Tema 51. Análisis de sistemas: Especificación funcional del sistema. Búsqueda y descripción de requisitos funcionales. Especificación de soluciones técnicas. Análisis de viabilidad técnica y económica.

Tema 52. Diseño lógico de funciones. Definición de funciones. Descomposición modular. Técnicas descriptivas. Documentación

Tema 53. Diseño lógico de datos. Transformación del modelo conceptual a modelos lógicos. Análisis relacional de datos. Documentación.

Tema 54. Diseño de interfaces de usuario. Criterios de diseño. Descripción de interfaces. Documentación. Herramientas para la construcción de interfaces.

Tema 55. Diseño físico de datos y funciones. Criterios de diseño. Documentación.

Tema 56. Análisis y diseño orientado a objetos.

Tema 57. Calidad del software. Factores y métricas. Estrategias de prueba.

Tema 58. Ayudas automatizadas para el desarrollo de software (herramientas CASE). Tipos. Estructura. Prestaciones.

Tema 59. Gestión y control de proyectos informáticos. Estimación de recursos. Planificación temporal, y organizativa. Seguimiento.

Tema 60. Sistemas basados en el conocimiento. Representación del conocimiento. Componentes y arquitectura.

Tema 61. Redes y servicios de comunicaciones.

Tema 62. Arquitecturas de sistemas de comunicaciones. Arquitecturas basadas en niveles. Estándares.

Tema 63. Funciones y servicios del nivel físico. Tipos y medios de transmisión. Adaptación al medio de transmisión. Limitaciones a la transmisión. Estándares.

Tema 64. Funciones y servicios del nivel de enlace. Técnicas. Protocolos.

Tema 65. Funciones y servicios del nivel de red y del nivel de transporte. Técnicas. Protocolos.

Tema 66. Funciones y servicios en niveles sesión, presentación y aplicación. Protocolos. Estándares.

Tema 67. Redes de área local. Componentes. Topologías. Estándares. Protocolos.

Tema 68. Software de sistemas en red. Componentes. Funciones. Estructura.

Tema 69. Integración de sistemas. Medios de Interconexión. Protocolos de acceso a redes de área extensa.

Tema 70. Diseño de sistemas en red local. Parámetros de diseño. Instalación y configuración de sistemas en red local.

Tema 71. Explotación y administración de sistemas en red local. Facilidades de gestión.

Tema 72. La seguridad en sistemas en red. Servicios de seguridad. Técnicas y sistemas de protección. Estándares.

Tema 73. Evaluación y mejora de prestaciones en un sistema en red. Técnicas y procedimientos de medidas.

Tema 74. Sistemas multimedia

38 - Modelo de datos relacional. Estructuras. Operaciones. Álgebra relacional.

Índice

1. Introducción 1
2. El modelo relacional 2
2.1. Estructura 2
2.2. Componentes 2
2.3. Representación gráfica 3
2.4. Normalización 3
3. Operaciones 4
3.1. Álgebra Relacional 4
3.2. Cálculo Relacional 5
4. Bibliografía 6

1. Introducción

El diseño de una base de datos consiste en definir la estructura de los datos que debe tener un sistema de información. Es una tarea muy compleja, que requiere tomar decisiones a varios niveles, por lo que se divide en tres fases en las que se definen tres modelos (cada uno de ellos con mayor nivel de detalle que el anterior): el modelo conceptual, el lógico y el físico.
Cada proceso toma como entrada la salida del proceso anterior.

- En el diseño conceptual se hace una descripción de alto nivel de la estructura de la base de datos, independientemente del SGBD (Sistema Gestor de Bases de Datos) que se vaya a utilizar para manipularla. Su objetivo es describir el contenido de información de la base de datos y no las estructuras de almacenamiento que se necesitarán para manejar dicha información.
- El diseño lógico parte del resultado del diseño conceptual y da como resultado una descripción de la estructura de la base de datos en términos de las estructuras de datos que puede procesar un SGBD relacional (tablas). Es dependiente del tipo de SGBD a utilizar (en este caso, uno relacional), pero no del SGBD concreto que se utilice.
- En el diseño físico se da forma al diseño lógico para adaptarlo a un SGBD concreto, y se completa con aspectos de implementación física de ese SGBD y aspectos de optimización del rendimiento de la BD.

El objetivo de esta división en fases es obtener un diseño de base de datos que cumpla las siguientes cualidades:

- Reflejar la estructura del problema en el mundo real, representando todos los datos esperados (incluso con el paso del tiempo), y ser lo suficientemente flexible para adaptarse de forma sencilla en caso de que surjan nuevas necesidades de información.
- Evitar el almacenamiento de información redundante: no solo incrementa innecesariamente el espacio de la BD sino que puede dar lugar a inconsistencias.
- Evitar la pérdida de dependencias funcionales.
- Proporcionar un acceso eficaz a los datos.
- Mantener la integridad de los datos a lo largo del tiempo.
- Ser claro, coherente y de fácil comprensión.

El diseño lógico de base de datos es el proceso por el que a partir de un esquema conceptual (representado por medio de un modelo conceptual: el diagrama ER o EER) se obtiene un esquema lógico. El esquema lógico se representa por medio de un modelo lógico de datos (jerárquico, en red, OO,…). El modelo utilizado en las BDs relacionales es el **modelo relacional**.

El modelo relacional es un modelo matemático, basado en la teoría de conjuntos. Los elementos del modelo se representan por medio de relaciones (un concepto propio de la teoría de conjuntos), y los mecanismos para procesar la información manejada por el modelo tienen también una base matemática: **el álgebra y el cálculo relacional.**

2. El modelo relacional

Las bases del modelo relacional fueron definidas por Edgar Codd, de los laboratorios IBM, en 1970, y en ellas se definían los siguientes objetivos del modelo:

- **Independencia física y lógica**: La forma de almacenar los datos (esquema físico), no debe afectar a cómo se ha definido su estructura (esquema lógico), y la modificación de su esquema lógico no debe afectar a las aplicaciones que hacen uso de la BD (que hacen uso del esquema externo)
- **Flexibilidad**: La base de datos debe ofrecer distintas vistas en función de los usuarios y aplicaciones, y debe ser sencilla de adaptar a nuevos requisitos.
- **Uniformidad y sencillez**: Tanto en la forma de representar los datos como en la forma de procesarlos. La uniformidad en la representación se consigue utilizando una misma estructura (las tablas o relaciones, concepto procedente de la teoría de conjuntos) para representar toda la información. El procesamiento de la información se realizará también de forma uniforme por medio de mecanismos basados en el álgebra y el cálculo relacional. El modelo tiene una base matemática tanto en su representación (relaciones, basadas en la teoría de conjuntos) como en su procesamiento.

2.1. Estructura

El modelo relacional es un modelo teórico con una sólida base matemática. Su componente principal son las relaciones, que están formadas por tuplas y atributos. Este modelo teórico se implementará en SGBD relacionales por medio de tablas (relaciones), compuestas de filas (tuplas) y columnas (atributos). Cada columna almacena información sobre una propiedad determinada de la relación. Cada fila posee una ocurrencia de la relación representada por la tabla.

El modelo relacional tiene una serie de restricciones:

- En una relación no puede haber dos filas idénticas.
- Todas las filas deben tener el mismo número de atributos.
- El orden de las tuplas y de las columnas es irrelevante.
- Cada atributo sólo puede tomar un único valor (no hay atributos multivaluados sino que todos deben ser atómicos).

La representación gráfica del modelo relacional consiste en la representación de todas las relaciones que conforman el modelo y las conexiones que se establecen entre ellas. Para establecer una conexión entre dos tablas es necesario incluir, en forma de columna, en una de ellas la clave primaria de la otra. A esta columna se le llama **clave foránea**.

2.2. Componentes

- **Relación**: Una relación es un conjunto de atributos al que se le da un nombre. Se representa gráficamente como una tabla con filas (tuplas) y columnas (atributos).
- **Atributos y dominios**: Un atributo es una característica que define a una relación. El **dominio** de un atributo es el conjunto de valores permitido para ese atributo.
- **Restricciones**: Condiciones que deben cumplir los datos de una tabla. Hay varios tipos:
    - de clave: conjunto de atributos que identifican de forma única un elemento de la relación
    - de valor único (unique): restricción que impide que un atributo tenga un valor repetido. Todas las claves tienen implícita una restricción de valor único.
    - de integridad referencial: se da cuando una tabla tiene una referencia a un atributo de otra tabla, y obliga a que exista el valor del atributo en la tabla referenciada.
    - de dominio: exige que el atributo tome valores dentro del dominio definido.
    - de nulidad (NOT NULL): permite indicar si un atributo es o no obligatorio
    - de Verificación (CHECK): permiten especificar una condición que deben cumplir los valores de determinados atributos de una relación.
    - Aserciones (ASSERTION): las aserciones son otro tipo de restricción en las que la condición se establece sobre elementos de distintas relaciones. Funcionan igual que las anteriores pero sobre varias relaciones en lugar de una sola. Un ejemplo de

aserción sería que “ningún empleado que trabaje en el departamento de contabilidad puede ganar más de 2500 €”.
    - Disparadores (TRIGGERS): permiten especificar una acción que se llevará a cabo en caso de que una condición sea verdadera. Se lanzan cuando se realiza una inserción, modificación o borrado de datos en una tabla.
- Clave: conjunto de atributos que identifican de forma única a una entidad. Existen varios tipos de clave
    - Superclave: cualquier conjunto de atributos que permita identificar de forma unívoca una entidad.
    - Clave candidata: cualquier superclave mínima.
    - Clave primaria: clave candidata elegida por el diseñador para identificar cada registro.
    - Clave foránea: atributo de una entidad que es clave de otra entidad. Cuando se borra un valor de la entidad referenciada, existen 4 opciones:
        - Prohibir la operación: no permitir borrar si el valor está siendo referenciado por otra tabla (en el ejemplo anterior, no se dejaría borrar la provincia)
        - Borrar en cascada: borrar todos los registros que hacen referencia a ese valor (en el ejemplo anterior, se borrarían todos los pacientes de la provincia que se elimina)
        - Colocar nulos (en el ejemplo anterior, se pondría a todos los pacientes de la provincia borrada un valor nulo en la provincia)
        - Utilizar un valor por defecto (en el ejemplo anterior, asignar una provincia “desconocido”, por ejemplo).

2.3. Representación gráfica

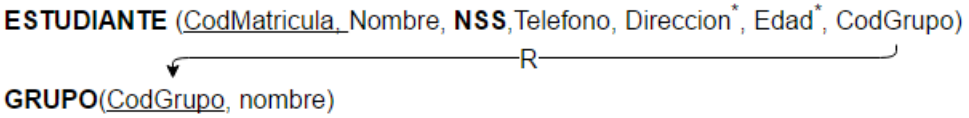
La representación gráfica del esquema relacional se puede realizar medio de dos herramientas:

- **El grafo relacional:** consiste en la representación de todas las relaciones que conforman el modelo por medio de su esquema (nombre de la relación + atributos que la componen):

$R(A_1, A_2, ..., A_n)$

Las convenciones utilizadas para la representación de este grafo son:

- El nombre de las tablas o relaciones se representa en mayúsculas
- Las claves primarias aparecen subrayadas
- Las claves alternativas aparecen en negrita
- Las claves ajenas están representadas en cursiva y referencian a la relación en la que son clave primaria mediante una flecha. En cada flecha se pueden indicar las opciones para la integridad referencial tanto en el borrado como en la modificación: C (cascada), R (restringido), N (puesta a nulos) o D (puesta a valor por defecto)
- Los atributos que pueden tomar valores nulos aparecen con un asterisco.



- **El modelo relacional:** es muy similar al grafo relacional, pero en este caso se representa cada relación como una tabla, y las claves foráneas serán enlaces entre tablas (cada clave foránea se enlazará con la clave primaria de la que depende). En los extremos de los enlaces se indica la cardinalidad de cada relación.

2.4. Normalización

El diseño lógico de una base de datos se suele abordar en dos fases: elaborar en primer lugar un modelo conceptual (por ejemplo el E/R), que se transformará posteriormente en un modelo lógico (que en este caso sería el modelo relacional). Esta aproximación puede dar lugar a relaciones que presenten problemas como

- Existencia de redundancias (con la consiguiente posibilidad de inconsistencias).
- Pérdida de dependencias funcionales (restricciones de integridad que dan lugar a interdependencias entre los datos).
- Aparición de anomalías de inserción, borrado y modificación.

Debido a esto, el esquema relacional debe ser siempre analizado para comprobar que no presenta los problemas anteriormente citados. La técnica para la verificación de esquemas relacionales se denomina **normalización**, y se basa en la aplicación de una serie de reglas o

formas normales, que permiten eliminar las redundancias existentes en el diseño manteniendo (en la mayoría de los casos) las dependencias funcionales.
Se dice que una relación se encuentra en determinada forma Normal, cuando la relación satisface una serie de restricciones sobre los atributos.
Para comprender las formas normales es necesario aclarar en primer lugar un par de conceptos:

- Una dependencia funcional es una conexión entre uno o más atributos: X-> Y.
- Una dependencia funcional es completa (X=>Y) si X->Y, y además Y no depende funcionalmente de ningún subconjunto de X
- Una dependencia funcional es transitiva si X->Y->Z, por lo que X->Z.

Existen seis formas normales, aunque las tres primeras en la mayor parte de los casos son suficientes para normalizar un esquema.

- **1º FN**: Una relación está en 1ºFN si todos sus atributos tienen valores atómicos (es decir, no puede tener atributos multivaluados). Ésta es una restricción inherente al modelo relacional.
- **2º FN**: Una relación está en 2ºFN si está en 1ºFN y todos los atributos que no forman parte de la clave tienen una dependencia funcional completa con respecto a la clave.
- **3ºFN**: Una relación está en 3ºFN si está en 2ºFN y no existen dependencias transitivas con respecto a un atributo que no sea clave.
- **FNBC (Boyce-Codd**): Una relación está en FNBC si está en 3ºFN y todos los atributos tienen dependencia funcional completa de la clave principal y solo de ella.
- **4ºFN**: evita las redundancias debidas a dependencias multivaluadas, que se producen cuando los valores de un conjunto de atributos dependen únicamente de los valores de otro conjunto de atributos de la misma relación: R(A,B,C), y tanto los valores de B como los de C dependen de los valores que tome A -> se representa como A->>B y A->>C. Para resolverlo se divide la relación en dos: R1(A,B) y R2(A,C)
- **5ºFN**: evita las redundancias debidas a dependencias de reunión, que se producen cuando una relación de varios conjuntos de atributos se puede construir como unión de subrelaciones entre esos conjuntos de atributos: R(A,B,C)=R1(A,B) U R2(A,C) U R3(B,C). Se resuelve dividiendo la relación en las subrelaciones que la componen.

3. Operaciones

Las operaciones de manipulación de datos en el modelo relacional son la selección (recuperación de datos existentes en la BD), inserción (insertar nuevas tuplas), modificación (modificar las tuplas existentes) y eliminación (eliminar tuplas que ya no son necesarias).
Para llevar a cabo estas operaciones, Codd propuso una serie de lenguajes con fundamento matemático denominados genéricamente “lenguajes relacionales”, que tienen como principal característica que son no procedimentales, es decir, no es preciso especificar el procedimiento a seguir para obtener la información, sino la información que se desea obtener. Estos lenguajes relacionales son el álgebra relacional y el cálculo relacional. Ambas operan sobre relaciones y dan como resultado otras relaciones. Su diferencia fundamental es que mientras el álgebra proporciona las instrucciones para construir una relación en base a otras dadas, el cálculo permite definir una relación en base a una serie de condiciones que deben cumplir los elementos de esa relación.

El álgebra relacional y el cálculo relacional tienen el mismo *poder de expresión*; es decir, todas las consultas que se pueden formular utilizando álgebra relacional pueden también formularse utilizando el cálculo relacional, y viceversa. Se dice que los lenguajes basados en el cálculo relacional son de "más alto nivel" o "más declarativos" que los basados en el álgebra relacional porque el álgebra especifica (parcialmente) el orden de las operaciones, mientras el cálculo lo traslada a un compilador o interprete que determina el orden de evaluación más eficiente.

3.1. Álgebra Relacional

Cuando Codd definió el modelo relacional, propuso una serie de operadores, a los que denominó Álgebra Relacional, con los que demostró que cualquier operación de manipulación de datos en una BD relacional se podía realizar utilizando esos operadores, combinándolos con operadores aritméticos y lógicos.
Los operadores del álgebra relacional son los siguientes:

- De consulta:
    - **Asignación (←)**: almacena el resultado de una operación en una relación.
    - **Selección (σ)**: extrae las tuplas de una relación que cumplan una restricción.

- **Proyección (π)**: extrae columnas de una relación.
- **Producto cartesiano (×)**: construye el producto cartesiano de dos relaciones. Existen diversas variaciones del producto cartesiano: join (θ - producto cartesiano cruzando por atributos comunes), join natural (⋈- producto cartesiano cruzando por claves foráneas) y outer join (incluye filas de una relación que no cruzan con la otra relación).
- **División (÷)**: es el operador inverso al producto cartesiano: la división de dos relaciones devuelve las tuplas de la relación "divisor" que crucen con todas las tuplas de la relación "dividendo".
- **Union (∪)**: la unión de dos relaciones con el mismo rango devuelve las tuplas que están en una u otra o en ambas.
- **Intersección (∩)**: La intersección de dos relaciones con el mismo rango devuelve las tuplas que están en ambas relaciones
- **Diferencia (−)**: la diferencia entre dos relaciones devuelve las tuplas que están en una pero no están en la otra.
- **Renombrado (*ρ*)**: permite cambiar el nombre a un campo de una relación.

- De inserción: la inserción se puede interpretar como la unión de una relación y una nueva tupla: r ← r U s;
- De modificación: se puede implementar como asignación de una expresión: r← π nomemp,sal+100 (r);
- De eliminación: la eliminación se puede implementar como una diferencia: r← r – σsal=1000 (r);

Por ejemplo, para recuperar los clientes con préstamos superiores a 10.000€:
r← π cliente.codcli,cliente.nomcli (σprestamo.importe>10000 (cliente⋈prestamo))

El lenguaje SQL implementa prácticamente todas las operaciones existentes en el álgebra relacional.

3.2. Cálculo Relacional

Existen dos variantes del cálculo relacional:

- El Cálculo Relacional de Dominios (DRC), que opera sobre variables formadas por componentes (atributos) de las tuplas.
- El Cálculo Relacional de Tuplas (TRC), cuyas variables son directamente las tuplas.

El lenguaje QBE (Query by example), utilizado en varios SGBD (a través de interfaces gráficas, como en Access o Mysqladmin) o bien por medio de consultas embebidas en un lenguaje de programación (como el JPA de hibernate), está basado en el cálculo relacional. Para recuperar información establecerá un ejemplo de las características que debe cumplir el resultado y el sistema devolverá todas las filas que cumplan la condición establecida.
El lenguaje SQL también incorpora características basadas en el cálculo relacional (como las cláusulas exists).

3.2.1. Cálculo relacional de Tuplas

El lenguaje del cálculo orientado a tuplas es no procedimental. Las consultas se indican por medio de expresiones de la forma: { t | P(t) }, donde t representa una tupla, y P(t) una propiedad que deben cumplir las tuplas.
La relación resultante estará formada por todas las tuplas que cumplan la condición P(t).
Por ejemplo, para recuperar los clientes con préstamos superiores a 10.000€:
{ cliente | ∃p ϵ PRESTAMO p[codcli]=cliente[codcli] ∧ p[importe]>10000 }

3.2.2. Cálculo relacional de dominios

Al igual que en el cálculo relacional de tuplas, las expresiones estarán formadas por variables (que en este caso se denominan dominios) que deben cumplir una propiedad: {<x1,x2,…,xn> | P(x1,x2,…,xn)}, donde xi son las variables de dominios y P(x1,x2,…,xn) es la propiedad que deben cumplir los atributos de la relación.
Por ejemplo, para recuperar los préstamos con importe superior a 10.000€:
{<codpr,codcli,codsuc,importe> | <codpr,codcli,codsuc,importe> ϵ PRESTAMO ∧ importe>10000}

4. Bibliografía

- Documentación oficial PostgreSQL: http://es.tldp.org/Postgresql-es/web/navegable/tutorial/sql.html
- Apuntes asignatura bases de datos de la Universidad de Sevilla: http://www.lsi.us.es/docencia/get.php?id=3183
- Wikipedia:
  - Modelo relacional: https://es.wikipedia.org/wiki/Modelo_relacional
  - Base de datos relacional: https://es.wikipedia.org/wiki/Base_de_datos_relacional