TEMA 38

MODELO DE DATOS RELACIONAL. ESTRUCTURAS. OPERACIONES. ÁLGEBRA RELACIONAL.

ÍNDICE

1. INTRODUCCIÓN

El modelo relacional es posterior a los modelos jerárquicos y de red. Nació como consecuencia de los trabajos publicados en 1970 por E. F. Codd, aunque los primeros SGBD relacionales no aparecen en el mercado hasta principios de los años ochenta.

La estructura lógica y el modo de realizar las operaciones de E/S en el enfoque relacional son distintos respecto a los enfoques jerárquico y en red.

El modelo relacional representa la Base de Datos por medio de tablas llamadas relaciones, cada una de las cuales se implementa como un fichero. Se caracteriza porque:

A la hora de operar con los datos, en lugar de hacerlo sobre registros actúa sobre la relación.

Peticiones complejas de información que pueden necesitar varios ficheros se pueden especificar de forma muy simple y sin ninguna dificultad.

Tiene el inconveniente de que su implementación puede ser más compleja. En el mercado hay diversos sistemas relacionases que pertenecen a una de las siguientes categorías: SYSTEM R, QUEL y QBE.

2. ARQUITECTURA DE UNA BASE DE DATOS RELACIONAL

En la arquitectura relacional de SYSTEM R, bajo las recomendaciones ANSI/SPARC, se pueden apreciar las siguientes partes:

1. El **nivel conceptual**. Definido por medio de una colección de relaciones, también llamadas tabla base, cada una de las cuales se representa por medio de un fichero almacenado distinto.

2. El **esquema externo**, llamado **submodelo relacional de datos** o **vista**. Es una tabla que no tiene existencia por sí misma y se deriva de una o más tablas base, a diferencia de un subesquema CODASYL, que sólo puede extraer datos de un fichero. Cada programa utiliza un buffer, llamado área de trabajo del usuario (UWA), para depositar o recuperar los datos antes de guardarlos en la Base de Datos o antes de ser procesados.

3. El **nivel interno**. Cada tabla base se representa por un fichero distinto que puede tener cualquier número de índices asociados.

4. Un **sublenguaje de datos**. Para el manejo de datos del sistema relacional. Con estos lenguajes, llamados de especificación, es posible procesar una tabla entera cada vez que se ejecuta una operación de Entrada/Salida en lugar de un registro, Los lenguajes que operan registro a registro se llaman lenguajes de navegación (jerárquicos y en red). Los sublenguajes de datos se pueden clasificar en dos grupos:

a) **Lenguaje de procedimiento**. Basado en el **álgebra relacional**, permite al usuario manipular relaciones por medio de operadores algebraicos relacionales para obtener los resultados requeridos.

b) **Lenguaje sin procedimiento**. Basado en el **cálculo relacional**, permite al usuario especificar exactamente qué datos desea, sin tener que detallar ni codificar la forma de obtenerlos a partir de las relaciones disponibles en la Base de Datos.

3. EL MODELO DE DATOS RELACIONAL

Las desventajas de los modelos jerárquico y en red condujeron a un intenso interés por el nuevo modelo de datos relacional. El modelo relacional fue descrito por primera vez por el Dr. Codd en 1970, y era un intento de simplificar la estructura de las Bases de Datos. Eliminaba las estructuras explícitas Padre/Hijo de la Base de Datos y en su lugar representaba todos los datos de la Base de Datos como sencillas tablas Fila/Columna de valores de datos.

Los primeros SGBD relacionases fallaron en implementar algunas partes clave del modelo de Codd, sólo ahora están incorporándose en productos comerciales. Conforme el concepto relacional crecía en popularidad, muchas Bases de Datos que se llamaban a sí mismas "relacionales" no lo eran de hecho. En respuesta a la corrupción del termino "relacional" el Dr. Codd, escribió un artículo en 1985 estableciendo 12 reglas a seguir por cualquier Base de Datos Relacional. Las 12 reglas de Codd han sido aceptadas desde entonces como la definición de un SGBD verdaderamente relacional, sin embargo, es más fácil la siguiente definición informal:

Una Base de Datos Relacional es una Base de Datos en donde todos los datos visibles al usuario están organizados estrictamente como tablas de valores, y en donde todas las operaciones de la Base de Datos operan sobre estas tablas.

Esta definición está destinada específicamente a eliminar estructuras tales como los punteros incorporados de una Base de Datos jerárquica o en red. Un SGBD Relacional puede representar relaciones Padre/Hijo, pero éstas se representan estrictamente por los valores contenidos en las tablas de la Base de Datos.

4. LAS DOCE REGLAS DE CODD

1ª. **Regla de Información**. Toda información de una Base de Datos Relacional está representada explícitamente a nivel lógico y exactamente de un modo (mediante valores en tablas).

2ª. **Regla de Acceso Garantizado**. Todos y cada uno de los datos (valor atómico) de una Base de Datos Relacional se garantiza que sean lógicamente accesibles recurriendo a una combinación de nombre de tabla, valor de clave primaria y nombre de columna.

3ª. **Tratamiento sistemático de valores nulos**. Los valores nulos (distintos de la cadena de caracteres vacía o en blanco y distinta de cero o de cualquier otro número) se soportan en los SGBD completamente relacionales para representar la falta de información y la información inaplicable de un modo sistemático e independiente del tipo de datos.

4ª. **Catálogo en línea dinámico basado en el modelo relacional**. La descripción de la Base de Datos se representa a nivel lógico del mismo modo que los datos ordinarios, de modo que los usuarios autorizados puedan aplicar a su interrogación el mismo lenguaje relacional que aplican a los datos regulares.

5ª. **Regla de Sublenguaje completo de datos**. Un sistema relacional puede soportar varios modos de uso terminal (Ej. el modo de rellenar con blancos). Sin embargo, debe haber al menos un lenguaje cuyas sentencias sean expresables, mediante alguna sintaxis bien definida, como cadenas de caracteres, y que sea completo en cuanto al soporte de los puntos siguientes:

- Definición de datos.
- Definición de vistas.
- Manipulación de datos (interactivo y por programa).
- Restricciones de integridad.
- Autorización.
- Fronteras de transacciones (comienzo, cumplimentación y vuelta atrás).

6ª. **Regla de actualización de vista**. Todas las vistas que sean teóricas actualizables son también actualizables por el sistema.

7ª. **Inserción, actualización y supresión de alto nivel**. La capacidad de manejar una relación de Base de Datos o una relación derivada como un único operando se aplica no solamente a la recuperación de datos, sino también a la inserción, actualización y supresión de los datos,

8ª. **Independencia física de los datos**. Los programas de aplicación y las actividades terminales permanecen lógicamente inalterados cualquiera que sean los cambios efectuados, ya sea a las representaciones de almacenamiento o a los métodos de acceso.

9ª. **Independencia lógica de los datos**. Los programas de aplicación y las actividades terminales permanecen lógicamente inalterados cuando se efectúen sobre las tablas de base, cambios preservadores de la información de cualquier tipo que teóricamente permita alteraciones.

10ª. **Independencia de integridad**. Las restricciones de integridad específicas para una Base de Datos Relacional particular deben ser definibles en el sublenguaje de datos relacional y almacenadas en el catálogo, no en los programas de aplicación.

11ª. **Independencia de distribución**. Un SGBD Relacional tiene independencia de distribución.

12ª. **Regla de no subversión**. Si un sistema relacional tiene un lenguaje de bajo nivel (un registro cada vez), ese bajo nivel no puede ser utilizado para subvertir o suprimir las reglas de integridad y las restricciones expresadas en el lenguaje relacional de nivel superior (múltiples registros a la vez).

En su artículo de 1985 en Computer World, Codd presentó las 12 reglas que una Base de Datos debe obedecer para que sea considerada verdaderamente relacionar Desde entonces se han convertido en una definición "semioficial" de una Base de Datos Relacional.

Ningún SGBD Relacional actualmente disponible satisface totalmente las 12 reglas de Codd. De hecho, se está convirtiendo en una práctica popular elaborar "tarjetas de tanteo" para productos SGBD comerciales, que muestran lo bien o mal que éstos satisfacen cada una de las reglas.

Desgraciadamente, las reglas son subjetivas de modo que los calificadores están generalmente llenos de notas a pie de página y no revelan demasiado acerca de los productos.

5. ESTRUCTURA DE DATOS RELACIONAL

5.1. Relaciones

Definición: Dada una serie de conjuntos D_1, D_2, ... D_n (no necesariamente distintos) se dice que R es una relación sobre estos n conjuntos si es un conjunto de n tuplas ordenadas $<d_1, d_2, ... d_n>$ tales que d_1 pertenece a D_1, d_2 pertenece a D_2 .... dn pertenece a D_n. Los conjuntos D_1, D_2, ... D_n son los dominios de R. El valor de n es el grado de R. (Un **dominio** es el conjunto de valores que puede tener un atributo).

Es conveniente representar una relación en forma de tabla bidimensional, Cada renglón de la tabla representa una tupla de la relación. El número de tuplas de la relación se llama cardinalidad de la relación.

También se puede definir como producto cartesiano: Dada una serie de conjuntos D_1, D_2, ... D_n (no por fuerza distintos), el producto cartesiano de estos n conjuntos, denotado por D_1 x D_2 x.... x D_n es el conjunto de todas las n tuplas ordenadas posibles $<d_1, d_2, ... d_n>$ tales que d_1 pertenece a D_1, d_2 pertenece a D_2 .... dn pertenece a D_n.

Ahora se define que R es una relación sobre los conjuntos D_1, D_2, ... D_n si es un subconjunto del producto cartesiano D_1 x D_2 x.... x D_n.

5.2. Dominios, atributos y claves

Es importante apreciar la diferencia entre un dominio, por una parte, y las columnas -atributos- que se sacan de ese dominio, por otra. Un atributo representa el uso del dominio dentro de una relación. Para

acentuar la distinción se pueden asignar nombres a los atributos que sean distintos de los dominios subyacentes.

La **clave primaria** es un identificador único de la tabla. Esto es, una columna (o combinación de columnas) con la propiedad de que, en un momento dado, no existen dos filas en la tabla conteniendo el mismo valor en la columna (o combinación de columnas).

5.3. Idea de normalización

En una Base de Datos relacional se requiere que todas las relaciones satisfagan la condición siguiente: *Que todo valor en la relación, es decir, cada valor de atributo en cada tupla sea atómico*.

En otras palabras, en cada intersección de un renglón y una columna de la tabla siempre hay exactamente un valor, nunca un conjunto de valores. (Se permite la posibilidad de valores nulos, es decir, valores especiales que representan algo desconocido o inapelable, como el caso de horas trabajadas de un empleado en vacaciones). De una relación que satisface la condición anterior se dice que está **normalizada** (O en primera forma normal).

Es elemental convertir una relación no normalizada en otra equivalente normalizada.

5.4. Restricciones de integridad

Las reglas de integridad son dos: **integridad de entidad** e **integridad referencial**. Estas reglas son generales, en el sentido de que deben aplicarse a toda Base de Datos definida de acuerdo a las prescripciones del modelo relacional. Aparte de estas dos, puede contener tantas reglas como se quiera, definidas de forma particular para ella.

Antes de enunciarlas, vamos a definir conceptos previos:

Claves candidatas: Las claves primarias definidas anteriormente no son más que un caso particular de lo que se denominan claves candidatas. Estas son precisamente un identificador único para la relación. Por definición, toda relación tiene al menos una clave candidata, pero en general, puede tener muchas. Cada una será una combinación de atributos que identifican exactamente a una tupla. Para una relación dada, nosotros elegimos una de ellas y la convertimos en clave primaria. Una clave candidata tiene dos propiedades fundamentales:

- **Unicidad**: En un momento dado, no puede haber dos tuplas en la relación en que todos los valores de los atributos que forman la clave sean exactamente iguales.
- **Minimalidad**: Ninguno de estos atributos puede ser extraído de la clave sin que se pierda la primera propiedad.

Claves externas (ajenas, foreign): Una clave externa es un atributo (o conjunto de atributos) en una relación A cuyos valores nos van a permitir buscar en la clave primaria de otra relación B.

Sobre estas definiciones podemos dar las dos restricciones de integridad:

Integridad de entidad: No se permite ningún atributo que participe en la clave primaria con valores nulos.

Integridad referencial: Si una relación A incluye una clave externa CE que corresponde con la clave primaria CP en una relación B, entonces todo valor de CE en A debe:

Ser igual al valor de CP en B.

o bien

Ser nulo.

Las dos relaciones A y B no tienen que ser necesariamente distintas.

En las dos restricciones, es fundamental el papel de los valores nulos. El valor nulo en el modelo relacional significa "propiamente inaplicable" o "información desconocida".

La justificación de las dos reglas de integridad es la siguiente:

La base de relaciones corresponde a entidades del mundo real.

Por definición, las entidades del mundo real son distinguibles. Esto es, tienen un identificador único para cada clave.

Las claves primarias llevan a cabo la función de identificación en el modelo relacional.

Así una clave primaria con valor nulo entra en contradicción con estos términos. No se pueden enunciar propiedades de objetos no definidos o parcialmente definidos.

Por otra parte, si un atributo existente en A refiere a una tupla en B, entonces esa tupla debe existir para que se pueda encontrar, o no tendría sentido la clave. No obstante, a veces es necesario admitir valores nulos en una clave externa; corresponderían al caso de "información desconocida".

6. FORMAS NORMALES

Las formas normales son las líneas maestras para el diseño de los registros, estas reglas están pensadas para evitar anormalidades en la puesta al día e inconsistencias en los datos, A pesar de ello, no hay obligación de seguir todas las normalizaciones hasta el final.

Antes de comenzar a enunciarlas, también hay que dar unos conceptos previos:

Dependencia funcional. Dada una relación R, se dice que el atributo Y de R es funcionalmente dependiente del atributo X de R (Se escribe R.X R.Y y se dice "R.X determina funcionalmente a R.Y") si y solo si para cada valor de X hay asociado precisamente un único valor a la vez de Y. Tanto X como Y pueden ser compuestos, es decir, grupos de atributos. Informalmente, se puede decir que Y depende de X si no puede existir dos tuplas con el mismo valor en X y distinto en Y. *Ejemplo*: El atributo *nombre* depende funcionalmente del atributo *DNI*.

Determinante es un atributo de una relación del que otro atributo cualquiera de ella es totalmente dependiente.

Hecho multivaluado es una relación muchos a muchos, como la relación hombre-empleo, en que un hombre puede tener varios empleos y un empleo (p.e. Electricista) lo pueden tener muchos hombres.

A. Primera forma normal

La primera forma normal simplemente define las tablas que forman el modelo relacional y excluye campos repetidos y grupos.

Una relación está en la primera forma normal si y sólo si los dominios de sus atributos toman sólo valores atómicos. Es decir, si no contiene atributos multivaluados.

Esta definición es equivalente a la anterior de relación normalizada e indica que una relación esta en primera forma normal si de hecho es correcta.

Ejemplo. La siguiente relación está en la primera forma normal:

R1	PIEZA	ALMACÉN	CANTIDAD	DIRECC_ALMACÉN

La clave primaria es el conjunto (**pieza, almacén**).

B. Segunda forma normal

La segunda forma normal trata de las relaciones entre los atributos clave y los no clave. Esta forma es violada cuando un campo no clave es de hecho un subconjunto de una clave.

Una relación R está en segunda forma normal si y sólo si está en primera forma normal y todo atributo no clave es completamente dependiente de la clave primaria. Es decir, si todos sus atributos dependen funcionalmente de toda la clave primaria. (Si la clave de una relación es simple ya se tiene la segunda forma normal).

La relación anterior Rl, viola esta regla, ya que, el atributo direcc-almacén no depende de la clave (pieza, almacén), sino que es función sólo de almacén. Para que se cumpla la segunda forma normal en este caso habrá que descomponer la relación en dos:

Ejemplo. Estas dos relaciones están en segunda forma normal:

R21	PIEZA	ALMACÉN	CANTIDAD

R22	ALMACÉN	DIRECC_ALMACÉN

En R22 la clave primaria será **almacén**.

C. Tercera forma normal

La tercera forma normal no se cumple cuando un campo no clave es de hecho clave de otro campo.

Ejemplo:

R23	EMPLEO	DEPARTAMENTO	LOCALIZACIÓN

CLAVE = (EMPLEO)

En este caso la clave es el atributo **empleo**, pero la **localización** depende exclusivamente del **departamento**.

R23.DEPARTAMENTO R23.LOCALIZACIÓN

Habrá que dividir la relación en otras dos para que se cumpla la tercera forma normal:

R31	EMPLEO	DEPARTAMENTO

CLAVE = (EMPLEO)

R32	DEPARTAMENTO	LOCALIZACIÓN

CLAVE=(DEPARTAMENTO)

La tercera forma normal se puede enunciar como:

Una relación está en la tercera forma normal si y sólo si está en la segunda y todo atributo no-clave es dependiente de la clave de forma no transitiva. Es decir, si está en la segunda forma normal y ningún atributo depende transitivamente de la clave primaria.

Este enunciado significa que la dependencia de Y con respecto a la clave X no se hace a través de un tercer atributo Z sino directamente:

R.X R.Y en vez de R.X R.Z R.Y

D. Forma normal de Boyce/Codd (BCNF)

La anterior definición de Codd de la tercera forma normal, tiene ciertas inadecuaciones, en concreto no trata los casos en que hay varias claves candidatas que se componen de varios atributos y están solapadas (tienen atributos comunes). Para complementarla se enuncia la forma normal de Boyce/Codd que mejora la primera definición.

Una relación está en la forma normal de Boyce/Codd si y sólo si todo determinante es clave candidata. Es decir, si todas las partes izquierdas de las dependencias son claves candidatas.

Toda relación en forma normal de Boyce/Codd está en la tercera forma normal por definición. La interpretación es que todo atributo que determine unívocamente otro en una relación debe poder ser clave de la misma.

E. Cuarta forma normal

Tanto la cuarta como la quinta forma normal trabajan con hechos multivaluados.

Bajo la cuarta forma normal, dos o más hechos independientes multivaluados no pueden coexistir en la misma relación.

Una relación está en cuarta forma normal, si y sólo si siempre que existe un hecho multivaluado, A B, entonces todos los atributos de la relación son funcionalmente dependientes de A. Es decir, si está en BCNF y no existe entre sus atributos ninguna dependencia multivaluada que no sea combinación de dependencias funcionales.

Por supuesto, A y B pueden ser compuestos.

En otras palabras, las únicas dependencias (simples o multivaluadas) dentro de la relación son las del tipo K X donde K es una clave candidata completa y X cualquier otro atributo.

F. Quinta forma normal

La quinta forma normal trata los casos en que la información está en distintas piezas que hay que mantener separadamente y sin redundancia.

Ejemplo. Si tenemos el caso de un concesionario de coches:

COCHES	VENDEDOR	MARCA	TIPO
	PÉREZ	RENAULT	COCHE
	PÉREZ	RENAULT	CAMIONETA
	PÉREZ	MERCEDES	COCHE
	PÉREZ	MERCEDES	CAMIONETA
	SUÁREZ	RENAULT	COCHE

Clave = (vendedor, marca, tipo)

Si suponemos que se cumple la regla: si un vendedor vende un cierto tipo de automóvil y representa una compañía entonces vende los productos de ese tipo y de esa compañía, podemos construir la tabla en base a tres relaciones:

COMPAÑÍA	VENDEDOR	MARCA
	PÉREZ	RENAULT
	PÉREZ	MERCEDES
	SUÁREZ	RENAULT

Clave = (vendedor, marca)

PRODUCTO	VENDEDOR	TIPO
	PÉREZ	COCHE
	PÉREZ	CAMIONETA
	SUÁREZ	COCHE

Clave = (vendedor, tipo)

CATÁLOGO	MARCA	TIPO
	RENAULT	COCHE
	RENAULT	CAMIONETA
	MERCEDES	COCHE
	MERCEDES	CAMIONETA

Clave = (marca, tipo)

Informalmente, podemos decir que una relación está en la quinta forma normal cuando su información no puede ser reconstruida si la dividimos en pequeños trozos.

Una relación está en la quinta forma normal si y sólo si toda dependencia en ella es una consecuencia de sus claves candidatas.

8. MANIPULACIÓN DE RELACIONES

Una vez vista la definición del modelo relacional y como crear una base de datos normalizada en él que conserve las restricciones de integridad, vamos a estudiar los lenguajes de manipulación de estas tablas.

Para la manipulación hay dos alternativas básicas: el álgebra relacional y el cálculo relacional. Ambas opciones trabajan con relaciones y dan como resultado de su trabajo otras relaciones. Como se verá, su diferencia fundamental es la siguiente: mientras el álgebra proporciona las instrucciones para construir una relación en base a otras dadas, el cálculo permite definir una relación en base a unos términos dados mediante un lenguaje que es más de construcción que de manejo.

8.1. Álgebra relacional

Las *operaciones básicas* del álgebra relacional son:

Selección: Extraer las tuplas especificadas de una relación (filas).

Proyección: Extraer atributos especificados de una relación (columnas).

Producto: Construye una relación de otras dos combinando de todas las formas posibles pares de tuplas, una de cada relación.

Unión: Construye una relación uniendo las tuplas que aparecen en una o ambas relaciones.

Intersección: Construye una relación con las tuplas que aparecen en ambas relaciones a la vez.

Diferencia: Construye una relación con las tuplas que aparecen en la primera y no en la segunda.

Enlace: Construye una relación de otras dos enlazando todas los posibles pares de tuplas, de forma que se cumpla una condición específica.

División: Tomando dos relaciones, una binaria (dos atributos) y otra unaria (uno sólo), construye una relación consistente en todos los valores de la relación binaria que tienen un enlace en la otra relación.

Ejemplos:

Selección:

COCHES	VENDEDOR	MARCA	TIPO
	PÉREZ	RENAULT	COCHE
	PÉREZ	RENAULT	CAMIONETA
	PÉREZ	MERCEDES	COCHE
	PÉREZ	MERCEDES	CAMIONETA
	SUÁREZ	RENAULT	COCHE

Clave = (vendedor, marca, tipo)

Coches where marca = "Renault":
(Selecciona de la tabla coches las tuplas en que la marca es **Renault**)

VENDEDOR	MARCA	TIPO
PÉREZ	RENAULT	COCHE
PÉREZ	RENAULT	CAMIONETA
SUÁREZ	RENAULT	COCHE

Selección y proyección:

(Coches where marca = "Renault" and tipo = "coche") [vendedor]
(Selecciona las tuplas en que la marca es RENAULT y el tipo COCHE y proyecta solo el atributo vendedor).

VENDEDOR
PÉREZ
SUÁREZ

Mediante este conjunto de ocho operaciones, se puede construir cualquier relación a partir de las ya dadas. El lenguaje resultante es altamente amigable y fácil de aprender. Plantea, sin embargo, el inconveniente de que es difícil el determinar la manera óptima de crear una relación con estas primitivas, ya que suelen existir varias opciones posibles, con costes distintos de utilización del ordenador.

8.2. Cálculo relacional

Como se ha dicho, la principal diferencia entre el álgebra y el cálculo relacional es que el álgebra construye y el cálculo define. El último ejemplo se puede enunciar desde el punto de vista algebraico:

- Restringe la relación **coches** a aquellas tuplas cuyo valor de marca sea **Renault** y el tipo de vehículo igual a **coche**.

- Proyecta sólo el atributo **vendedor**.

El cálculo la formularía de esta manera:

- Crea una relación de un único atributo **vendedor** y compuesta por los atributos **vendedor** de aquellas tuplas de **coche** en que la marca sea **Renault** y el tipo **coche**.

La característica fundamental del cálculo es el concepto de tupla variable. Esta es una variable cuyos valores permitidos son tuplas de una relación. Así, en el ejemplo anterior, si una relación está almacenada en la variable **coches**, para obtener **vendedor** sólo tenemos que escribir **Coches.Vendedor**. El seleccionar un grupo de tuplas de una relación se hace mediante las sentencias **select from where** también esta

disponible el cuantificador existencial **exits** y al cuantificador universal **forall**. El funcionamiento básico de este lenguaje se puede ver en los siguientes *ejemplos*:

Selección:

COCHES

VENDEDOR	MARCA	TIPO
PÉREZ	RENAULT	COCHE
PÉREZ	RENAULT	CAMIONETA
PÉREZ	MERCEDES	COCHE
PÉREZ	MERCEDES	CAMIONETA
SUÁREZ	RENAULT	COCHE

Clave = (vendedor, marca, tipo)

SELECT
FROM COCHES
WHERE COCHES.MARCA = "RENAULT"
(Selecciona de la tabla coches las tuplas en que la marca es Renault)

VENDEDOR	MARCA	TIPO
PÉREZ	RENAULT	COCHE
PÉREZ	RENAULT	CAMIONETA
SUÁREZ	RENAULT	COCHE

Selección y proyección:

SELECT COCHES.VENDEDOR
FROM COCHES
WHERE (COCHES.MARCA = "RENAULT" AND COCHES.TIPO = "COCHE")

(Selecciona las tuplas en que la marca es RENAULT y el tipo COCHE y proyecta solo el atributo vendedor)

VENDEDOR
PÉREZ
SUÁREZ

Selección, proyección y unión de dos relaciones:

Si además tenemos la relación:

EDAD

VENDEDOR	EDAD
PÉREZ	45
SUÁREZ	32
FERNÁNDEZ	35

Para obtener la menor edad: MIN (EDAD.EDAD) / 32

Para obtener al vendedor más viejo de productos Renault:

SELECT EDAD.VENDEDOR FROM EDAD
WHERE EDAD.VENDEDOR = MAX (SELECT EDAD.VENDEDOR, EDAD.EDAD
FROM EDAD, COCHES
WHERE (EDAD.VENDEDOR = COCHES.VENDEDOR) AND COCHES.VENDEDOR =
"RENAULT")

8. BIBLIOGRAFÍA

Gary W. Hansen
Diseño y Administración de Bases de Datos
Prentice Hall, 1998

Alberto Prieto
Introducción a la Informática
Mc Graw-Hill, 2ª edición, 1997

Alfonso Ureña López
Fundamentos de Informática
Ra-ma, 1997