Los componentes del modelo son:

Estructuras:

- Relación (esquema e instancia)

Restricciones:

- Clave primaria
- Unicidad de atributos
- Restricciones de dominio
- Restricción de integridad referencial

Operaciones:

- Algebra Relacional
- Cálculo Relacional (de dominios y de tuplas)
- SQL (Structured Query Language)

Atributos Clave

SUPERCLAVE:

Subconjunto de los atributos de un esquema de relación R, tal que no hay dos tuplas en toda instancia de R que tengan el mismo valor en todos los atributos del subconjunto.

CLAVE:

superclave minimal. Es decir, superclave a la cual no le sobra ningún atributo, si se elimina algún atributo deja de ser superclave.

En un esquema de relación R puede haber varios candidatos a clave y se selecciona uno de esos subconjuntos de atributos como *la clave primaria*, de modo que hay una sóla clave primaria.

Las operaciones del álgebra relacional operan sobre relaciones (conjuntos de tuplas) y producen relaciones como resultado. Dependiendo de cuántas relaciones reciba como entrada, cada operador del álgebra relacional se puede clasificar en: *unario*, si recibe una sóla relación; *binario*, si recibe dos relaciones como entrada. Todos estos operadores producen una sóla relación como resultado.

Operadores Unarios:

- Select (σ): operador de selección.
- Project (π): operador de proyección.
- Rename (ρ): operador para cambiarle el nombre a una relación o a sus atributos.

Operadores Binarios:

- Join ($\bowtie$, o $*$).
- Producto Cartesiano ($\times$).
- Unión ($\cup$), Intersección ($\cap$) y Diferencia ($-$), requieren de las relaciones de entrada que sean *compatibles bajo la unión.*
- División ($\div$).

SELECT (σ): operador de selección.

Permite seleccionar un subconjunto de las tuplas de una relación, se *seleccionan* aquellas tuplas que cumplen con la *condición de selección (selection condition)*.

$$\sigma_{<condicion\ de\ seleccion>}(R)$$

La $<$ *condicion de seleccion* $>$ es una expresión booleana que consiste de cláusulas separadas por los operadores booleanos: AND, OR y NOT.
Cada cláusula es de una de las siguientes formas:

$$< atributo1 > \quad < opComp > \quad < constante >$$

$$< atributo2 > \quad < opComp > \quad < atributo1 >$$

donde,
atributo1 y atributo2 son atributos de la relación R
constante es un valor tomado del dominio de atributo1, y
el opComp es un operador de comparación de la siguiente lista:
$=,\ <,\ \leq,\ >,\ \geq,\ \neq$

Ejemplo

Si se quiere *listar los empleados de 25 años o menos al año 2000*, se puede hacer

$$\sigma_{Anac\ \geq\ 1975}(EMPLEADO)$$

Esta consulta accesa la relación EMPLEADO y revisa cada tupla individualmente para determinar si cumple con la condición de selección. En este caso la condición es que el año de nacimiento sea mayor o igual a 1975.

Resultado de una operación de selección

Una relación resultante, del mismo grado que la relación R sobre la cual se aplica la selección, y cardinalidad:

$$|\sigma_C(R)| \ \leq \ |R|$$

Otros ejemplos

Listar las empleadas de la compañía

$$\sigma_{sexo\ =\ ``F''}(EMPLEADO)$$

Listar los empleados quienes para el 2000 tenían 50 años o más, que no tienen jefe

$$\sigma_{Anac\ <\ 1950\ AND\ superssn\ =\ null}(EMPLEADO)$$

Datos del Departamento de Administracion

$$\sigma_{dnombre\ =\ ``Administracion''}\ (DEPARTAMENTO)$$

Propiedades de la selección

- Conmutatividad:
  $\sigma_{C1}(\sigma_{C2}(R))\ =\ \sigma_{C2}(\sigma_{C1}(R))$
- Cascada de selects:
  $\sigma_{C1}(\sigma_{C2}(...(\sigma_{Cn}(R))))\ =\ \sigma_{C1\ AND\ C2\ AND....AND\ Cn}(R)$

PROJECT (π): operador de proyección.

Permite escoger un subconjunto de los atributos de cada tupla de una relación, en otras palabras elige un subconjunto vertical, mientras que la selección escoge uno horizontal.

$$\pi_{<lista\ de\ atributos>}(R)$$

La $<$ *lista de atributos* $>$ es un subconjunto de los atributos de R.

Si sólo queremos el número del departamento de Administración, el operador σ no basta. Para quedarnos con algunas columnas de una tabla, usamos el project (π), por ejemplo, la siguiente expresión nos da la columna dnumero de la relación DEPARTAMENTO.

$$\pi_{dnumero}(DEPARTAMENTO)$$

y si queremos el número del departamento de Administración, hacemos:

$$\pi_{dnumero}\ (\sigma_{dnombre\ =\ ``Administracion''}(DEPARTAMENTO))$$

Esto último es un ejemplo de una *secuencia de operaciones* o de la aplicación de operaciones del álgebra en secuencia.

¿Qué pasa si elegimos la columna *dnumber* de la relación $DEPT-LOCATIONS$ para hacer un project sobre ella?

Nos da una columna con los valores 1, 4, 5, 5, 5
pero eso no es una relación, pues el 5 aparece repetido 3 veces.

El project, antes de producir la relación resultante, debe verificar que no haya repetidos.

De modo que realmente la salida de $\pi_{dnumber}(DEPT-LOCATIONS)$ es una relación así:

dnumber
1
4
5

JOIN ($\bowtie$, $*$):

Permite relacionar tablas a través de atributos comunes. Un atributo común a dos tablas es aquel que está definido en ambas y toma valores del mismo dominio.

$$R \bowtie_{<condicion\ de\ join>} S$$

Relaciones Compatibles bajo la unión

Dos relaciones, $R(A_1,\ A_2,\ ..., A_n)$ y $S(B_1,\ B_2,\ ...,\ B_N)$ son *compatibles bajo la unión*, sii:

- tienen el mismo grado, y
- los atributos A_i y B_i están definidos en el mismo dominio, $\forall\ i,\ i = 1, .., n$.

En otras palabras si tienen el mismo número de atributos y los atributos correspondientes están definidos en el mismo dominio en ambas relaciones.

Operaciones de conjunto: unión ($\cup$), intersección ($\cap$) y diferencia ($-$).

Sean R y S relaciones compatibles bajo la unión,

Unión: $R \cup S$ es la relación que incluye todas las tuplas que están en R, o están en S, o están en ambas. Elimina duplicados.

Intersección: $R \cap S$ es la relación que incluye todas las tuplas que están en R y en S.

Diferencia: $R - S$ es la relación que incluye todas las tuplas que están en R, pero que no están en S.

PRODUCTO CARTESIANO ($\times$):

Sean $S(B_1,\ B_2,\ ...,\ B_n)$ y $T(C_1,\ C_2,\ ...,\ C_m)$ relaciones.
$S \times T$ es la relación de grado $n + m$ formada tomando una tupla de S y concatenándola con un tupla de T. La cardinalidad del producto cartesiano es:
$|S \times T| = |S| * |T|$.

En otras palabras, con el producto cartesiano se crea una nueva relación, que tiene todos los atributos de S, todos los atributos de T y que combina cada tupla de S con todas las tuplas de T.

Semánticamente no tiene mucho sentido esta operación, pero sirve para expresar algunas operaciones importantes del álgebra.

Por ejemplo, para obtener el equijoin de dos relaciones, se puede utilizar el producto cartesiano, de la siguiente forma:
$S \bowtie_{B_i = C_j} T \equiv \sigma_{B_i = C_j}(S \times T)$

DEFINICIONES:

Sean $R(A_1,\ A_2,\ ...,\ A_n),\ r\ =\ instancia\ de\ R$ y llamemos $A = A_1,\ A_2,\ ...,\ A_n$

Superclave: $S\ \subseteq\ A$ es una *superclave sii* $\forall\ t_1, t_2\ in\ r,\ t_1[S]\ \neq\ t_2[S]$

Clave: Sea $K \subseteq A$ una superclave, K es una *clave* si además es minimal, es decir, si al eliminar cualquier atributo de K, el conjunto de atributos resultante ya no es más una superclave.

Candidato a clave: Si una relación tiene más de una clave, a cada una de ellas se le llama *candidato a clave*.

Atributo Primo: es un atributo de una relación que es miembro de algún candidato a clave de la relación.

Atributo No Primo: es un atributo de una relación que *no es miembro* de ningún candidato a clave de la relación.

DEPENDENCIA FUNCIONAL:

Sean X y Y dos conjuntos de atributos que existen en una base de datos, en el esquema de una relación R.

Si para todo par de tuplas t_1 y t_2 en *cualquier instancia posible* r de R, se cumple la siguiente restricción:

$$t_1[X] \;=\; t_2[X] \;\Longrightarrow\; t_1[Y] \;=\; t_2[Y],$$

en otras palabras, si que las dos tuplas tengan el mismo valor en los atributos de X implica que tienen los mismos valores en los atributos de Y,
(si t_1 y t_2 coinciden en X implica que también coinciden en Y)

entonces,
existe una dependencia funcional entre X y Y, la cual se denota como

$$X \longrightarrow Y$$

Si $X \longrightarrow Y$ decimos:

1. Los valores de X determinan unívocamente los valores de Y.
2. *X determina funcionalmente a Y.*
3. Hay una dependencia funcional desde X hasta Y.
4. Y es *funcionalmente dependiente de X.*

Dependencia funcional se abrevia como *DF*. En

$$X \longrightarrow Y$$

X es el *lado izquierdo de la DF* y Y es el *lado derecho de la DF.*

NOTAS:

- Sea $R(K, Y)$ un esquema de relación, donde K es un candidato a clave de R y Y es el conjunto de todos los otros atributos de R, entonces se cumple que: $K \longrightarrow Y$.
- El hecho de que $X \longrightarrow Y$ *no dice nada acerca de* $Y \longrightarrow X$.
- Si K es candidato a clave de R, esto implica que $K \longrightarrow Z$, donde Z es cualquier subconjunto de los atributos de R.

SIGNIFICADO DE LAS DF's:

Una dependencia funcional es una propiedad de la semántica de los atributos. Las DF's dicen cómo los atributos están relacionados unos con los otros, semánticamente.

¿Cómo determinamos las dependencias funcionales? (elija una)

1. Analizando la semántica de los datos.
2. Observando una extensión (instancia) específica de la relación y viendo qué pasa en las tuplas.

Las DF's dicen cuáles son las *extensiones legales* de las relaciones. Esas extensiones en las cuales las DF's no se cumplen, no son legales.

Las DF's especifican restricciones en los atributos de una relación que deben cumplirse *siempre*.

Las DF's restringen los posibles valores de los atributos.

Teoría de la Normalización

Desarrollada dentro del modelo relacional para:

1. Verificar si alguna de las situaciones patológicas se presenta en las relaciones de una BD relacional.
2. Descomponer las relaciones *mal diseñadas* en dos o más relaciones *buenas*.

Utiliza las definiciones relativas a las *claves* y las *dependencias funcionales* para establecer el concepto de *Forma Normal (NF)*.

Forma Normal Una forma normal define condiciones que debe cumplir una relación para estar en esa forma normal.

Las formas normales que vamos a estudiar son: 1NF, 2NF, 3NF y BCNF.

Primera Forma Normal (1NF): una relación está en 1NF si todos sus atributos sólo pueden tomar valores atómicos. En el modelo relacional, todas las relaciones, por definición están en 1NF.

Segunda Forma Normal (2NF): una relación R está en 2NF si todo atributo no primo de R es dependiente funcional y completamente de la clave primaria.

Si R no está en 2NF, se descompone en varias relaciones que sí están en 2NF, pero esta descomposición debe cumplir ciertas condiciones para que sea buena, pronto veremos cuáles son esas condiciones.

Tercera Forma Normal (3NF): una relación R está en 3NF si:

- está en 2NF, y
- ningún atributo no primo de R es transitivamente dependiente de la clave primaria.

Nuevamente, el procedimiento al encontrar una relación que no está en 3NF, es descomponerla en varias relaciones que sí están en 3NF.

Las definiciones anteriores no consideran si la relación tiene más de un candidato a clave.

Consideremos la siguiente relación:

ESTUDIANTE (<u>Carnet</u>, nombre, CI, carrera, dir, tel, fechaNac)

¿está en 2NF?
Si, todo atributo no primo (nombre, CI, carrera, dir, tel, fechaNac) es funcional y completamente dependiente de la clave primaria.

¿está en 3NF?
está en 2FN. ¿CI es un atributo primo o no primo?
CI es primo, $Carnet \longrightarrow CI$ y $CI \longrightarrow nombre$
en consecuencia, $Carnet \longrightarrow nombre$ es una dependencia funcional transitiva. No está en 3FN.
Pero en realidad no hay problema, pues CI también es una clave.

Cuando hay varios candidatos a clave en R, es necesario usar una definición más general que los tome en cuenta a todos.

DEFINICIONES GENERALES

Segunda Forma Normal (2NF): un esquema de relación, R, está en 2NF si todo atributo no primo A de R es dependiente funcional y completamente de todas los candidatos a clave de R.

Tercera Forma Normal (3NF): un esquema de relación R está en 3NF si para toda dependencia funcional, $X \longrightarrow A$ que se da en R, se cumple alguna de las dos condiciones siguientes:

- X es una superclave de R, o
- A es un atributo primo de R.

Consideremos la siguiente relación:

ALCALDIA (<u>idPropiedad</u>, nombreMunicipio, nroTerreno, area)

Las dependencias funcionales que se cumplen en esta relación son:

$$DF1: \; idPropiedad \longrightarrow nombreMunicipio, \; nroTerreno, \; area$$

$$DF2: \; nombreMunicipio, \; nroTerreno \longrightarrow idPropiedad, \; area$$

$$DF3: \; area \longrightarrow nombreMunicipio$$

¿está en 3NF?

Usando la definición general de 3NF, debemos verificar para cada DF, que el lado izquierdo sea superclave o que el lado derecho sea primo.

Si está en 3NF.
En DF1 y en DF2, el lado izquierdo es superclave.
En DF3, el lado derecho es primo.

¿Hay algún problema con esta relación ALCALDIA?
Si, hay redundancia en el nombre del Municipio, pues está determinado por el área y ¿si sólo hay dos áreas posibles y muchos municipios?

SOLUCION:
Descomponer la relación en:
ALCALDIA(<u>idPropiedad</u>, nroTerreno, area)
AREA (<u>area</u>, nombreMunicipio)

Boyce-Codd Normal Form (BCNF): un esquema de relación R está en BCNF si para toda dependencia funcional, $X \longrightarrow A$ que se da en R, se cumple que:

- X es una superclave de R.

BCNF es más fuerte que 3NF. Pues 3NF permite que haya una dependencia funcional donde el lado izquierdo no sea una superclave, si el lado derecho es primo.