

Resumen Bases de Datos 1

Clase 1

Modelo de datos

- Provee una notación para describir los datos
- Se define a partir de
 - Estructura de los datos
 - Restricciones sobre los datos
 - Operaciones con los datos (optativo)
- Conjunto de conceptos que pueden usarse para describir la estructura de una base de datos

Tipos de Modelos de Datos

- Modelos lógicos
 - Basado en objetos
 - Modelo de Entidades y Relaciones
 - Modelo Basado en Objetos
 - Basados en registros
 - Modelo Relacional
- Modelos Físicos

Modelo de Entidades y Relaciones

Se define a partir de:

- Estructura
 - Entidad
 - Es una "cosa o concepto" que puede ser identificada y distinguible de otra "cosa o concepto".
 - Relación
 - Es una asociación de entidades.
 - Ejemplos: Juan con dni 1234567 es_dueño_de un auto modelo 2015 cuya patente es PRI.
 - Atributo
 - Representa información acerca de una entidad o una relación.
 - Ejemplos: nombre, dni, modelo, patente.
- Restricciones

- Cardinalidad
 - Determina el número de veces en el que puede participar una entidad en una relación.
 - Indica dependencia (importancia de la cardinalidad mínima).
 - total o de existencia: participación obligatoria (al menos uno).
 - parcial: participación no obligatoria (puede ser cero).
 - puede ser:
 - Uno a uno.
 - Uno a muchos
 - Muchos a muchos
- Identificador o clave
 - Restricción de unicidad del valor del atributo
 - Sirven para identificar de manera única a una entidad
 - Toda entidad posee al menos una posible clave o identificador
 - Puede ser:
 - Simple
 - Compuesto
- Grado
 - Representa el número máximo de veces que una entidad puede estar relacionada con otra.
 - Ejemplos:
 - 1,N (grado N)
 - 1,1 (grado 1)
- Acerca de los nombres
 - No se pueden repetir los nombres de los atributos en una misma entidad ni en una misma relación.
 - No se pueden repetir nombres ni para entidades, ni para relacionales, ni para ninguna de ellas

Definición

Conjunto de entidades:

Es un conjunto de entidades del mismo tipo

Ejemplos:

El conjunto de todas las personas que poseen un nombre y tienen un dni puede llamarse PERSONA

El conjunto de todos los autos que poseen información del modelo y de la patente puede llamarse AUTO

Definición

Dominio de un atributo:

Conjunto de valores que puede tomar un atributo en particular

Ejemplo: nombre puede ser una cadena de máximo 50 letras del abecedario

Definición

Conjunto de relaciones

Es un conjunto de relaciones del mismo tipo

Ejemplo:

ES_DUEÑO_DE es un conjunto de relaciones entre las entidades PERSONA Y AUTO

Important

Los términos entidad y conjunto de entidades serán intercambiables, haciendo abuso del vocabulario

Los términos relación y conjunto de relaciones serán intercambiables, haciendo abuso del vocabulario

Tener en cuenta que toda entidad debe, al menos, tener un atributo

Notación Gráfica

- Diagrama de entidades y relaciones

- Representación gráfica de la estructura de los datos

- Entidad



- Relación



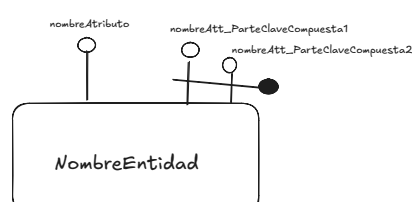
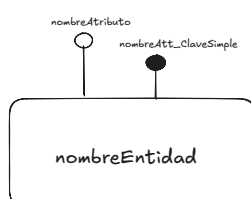
- Atributo



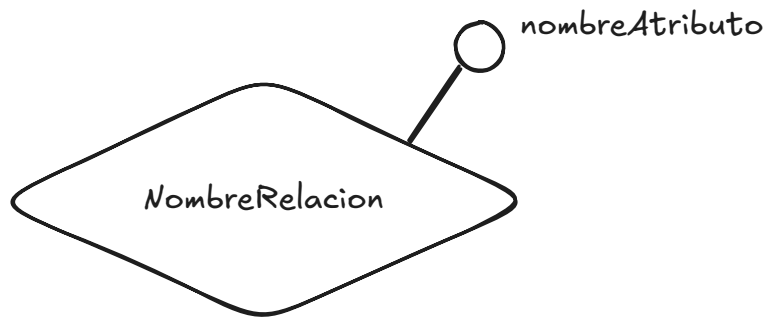
- Cardinalidad

- (cardMin,cardMax)

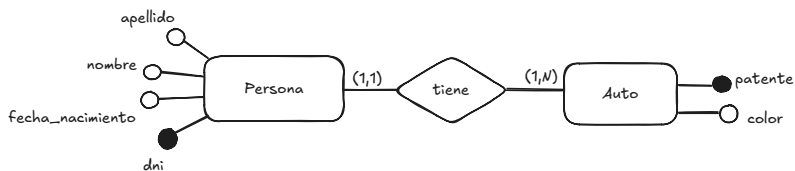
- Notación de atributos descriptores e identificadores simples y compuestos en una entidad.



- Notación de atributo en la relación.



Ejemplo de un diagrama de Entidades y relaciones.



¿Cómo se lee la restricción de cardinalidad?

Una persona tiene al menos un auto y a lo sumo n

Y un auto es poseído por una única persona

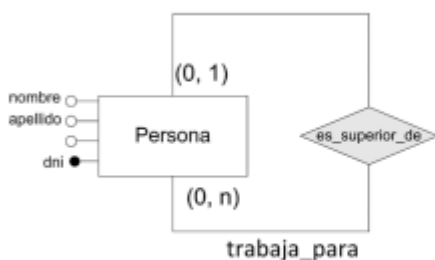
Rol de una entidad en una relación

Indica la función que tiene la entidad en la relación.

Ejemplo:

- tutor_de
Juan con dni 123456 es tutor_de Maria cuyo dni es 234567.
Esta última, tiene el rol de tutelada_por

Ejemplo de un diagrama de Entidades y Relaciones –Rol-



¿Cómo se lee la restricción de cardinalidad?

Una persona es superior de cero o muchas otras personas

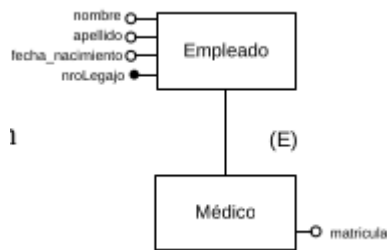
Una persona trabaja para a lo sumo una persona

Especialización

Es el resultado de tomar un subconjunto de entidades de un nivel para formar un conjunto de entidades de nivel más bajo.

Ejemplo:

- Tenemos empleados de un hospital. De los médicos nos interesa su matrícula. Puede haber empleados que no son médicos.



Generalización

Es el resultado de tomar uno o más conjuntos de entidades (de nivel más bajo) y producir un conjunto de entidades de un nivel más alto.

Ejemplo:

- Distintos tipos de cuenta: cajas de ahorro y cuentas corrientes, pero ambas son consideradas cuentas.



Mecanismos de abstracción



Generalización: No hay otro tipo de cuentas

Especialización: Podría haber otra caja de ahorro especial

Agregación

Para entenderlo mejor, supongamos el siguiente ejemplo:

- se guardan entrevistas de solicitantes de empleo a varias compañías

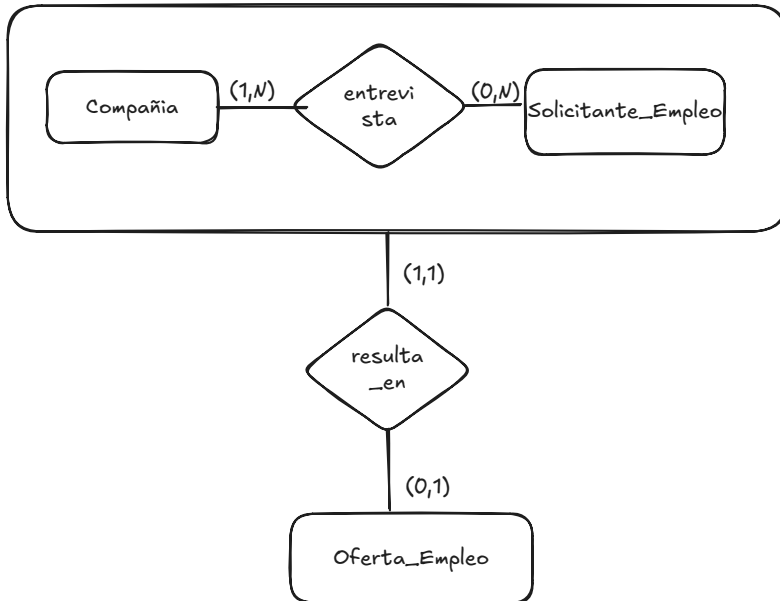


- Supongamos, además que algunas entrevistas resultan en ofertas de empleo, pero otras no.
- Un problema del modelo de entidades y relaciones es que:
 - No es posible expresar relaciones entre relaciones existentes

Agregación: Es un mecanismo de abstracción en el cual una relación binaria (junto a las dos entidades relacionadas) se trata como entidad de alto nivel

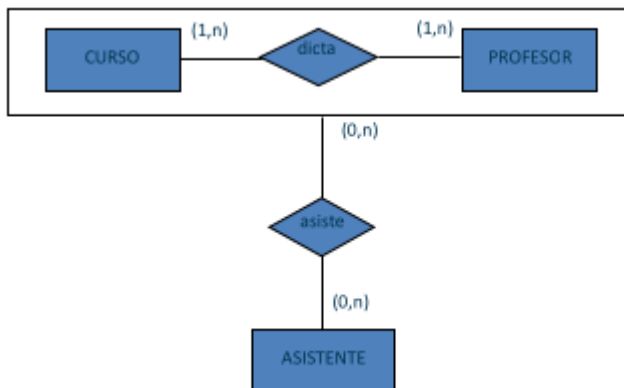
Nota: la cardinalidad máxima para cada entidad de la relación siempre es mayor a 1.

Volviendo al ejemplo anterior, se soluciona de la siguiente forma:



Otro ejemplo:

- Un profesor puede dictar uno o varios cursos. Una vez asignado un profesor a un curso es posible que se registren asistentes a dicha asignación.



Modelo Basado en Registros

Modelo de Relacional

Se define a partir de:

Estructura

- Relación
 - Representa los datos como tablas bidimensionales llamadas relaciones

- Ejemplo: Persona
- Atributo
 - El nombre de cada columna de la relación o tabla.
- Esquema
 - Está formado por el nombre de una relación y su conjunto de atributos - Ejemplo: Persona(dni, edad, nombre)
 - Nota: los atributos de un esquema son un conjunto y no una lista, por lo tanto, no hay un orden físico.
- Tupla
 - Son las filas de una relación (excepto sus encabezados).
 - Posee un solo componente para cada atributo de la relación
 - Ejemplo (123456, 54, Juan) es una tupla con tres componentes de la relación Persona

Restricciones

- Clave de una relación
 - Un conjunto de atributos conforma una clave en la relación cuando a dicho conjunto no se le permite tomar dos valores iguales en todos los atributos de la clave
 - Ejemplo: Persona(dni, edad, nombre)
- Dominio de un atributo
 - Cada componente de cada tupla debe ser atómica, es decir, debe ser un tipo elemental (no puede ser una lista, un registro, etc)
- Acerca de nombres
 - Unicidad en nombres de esquemas, relaciones y atributos dentro de un esquema

Transformación 1 a 1 de Modelos

Desde Modelo de Entidades y Relaciones al Modelo Relacional

- Los modelos de datos son independientes entre si
- Puedo crear un Modelo Relacional sin previamente haber creado un Modelo de Entidades y Relaciones
- En este tipo de transformación TODAS las entidades y Relaciones se transforman en un Esquema de Relación

Cómo convertir del modelo de entidades y relaciones (E/R) al modelo relacional.

- Convertir cada conjunto de entidades en una relación (con igual nombre) con el mismo conjunto de atributos.
- Convertir cada relación del modelo de entidades y relaciones en una relación (del modelo relacional), de igual nombre
 - Para cada entidad involucrada en la relación, se toma el o los atributos claves como

parte del esquema de la relación (del modelo relacional).

- Si la relación (del modelo de entidades y relaciones) posee atributos, éstos también forman parte del esquema de la relación.
- Si una entidad está involucrada más de una vez en una relación, con diferentes roles, se renombrará el atributo para evitar nombres duplicados, adoptando el nombre del rol de la entidad en la relación

Nota: Las reglas anteriores cubren la mayoría de los casos para convertir de un modelo a otro.

Otras reglas particulares serán vistas a continuación

- Entidad



CUENTA(númeroCuenta, saldo)

- Relaciones

- (asumiendo que la entidad CLIENTE posee al atributo numeroCliente como clave, mientras que CUENTA al atributo numeroCuenta)



tiene(**numeroCliente**, numeroCuenta)

o

tiene(numeroCliente, **numeroCuenta**)



tiene(numeroCliente, **numeroCuenta**)

- Entidades y relaciones

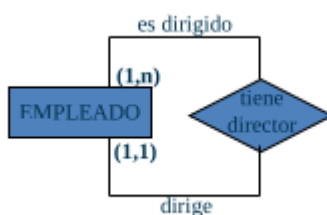


PROFESOR(**codigoProfesor**, nombre, título)

CURSO(**codigoCurso**, título, tema)

DICTA(**codigoProfesor**, **codigoCurso**, fecha)

- Rol

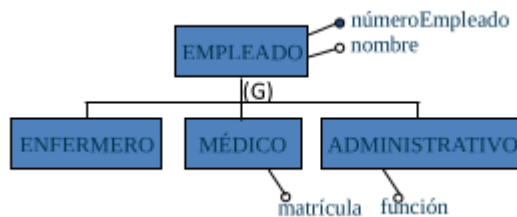


TIENE_DIRECTOR(**númeroEmpleado**, númeroDirector)

EMPLEADO(**númeroEmpleado**, nombre)

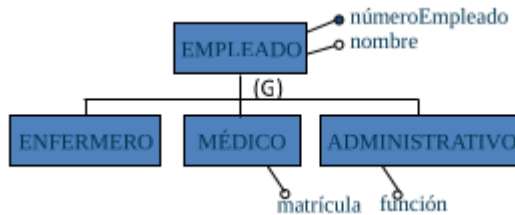
- Generalización (Tres estrategias)

- Una tabla para el conjunto de entidades de nivel más alto



EMPLEADO (**númeroEmpleado**, nombre, tipoEmpleado, matrícula, función)

- Una tabla para cada conjunto de entidades de nivel más bajo

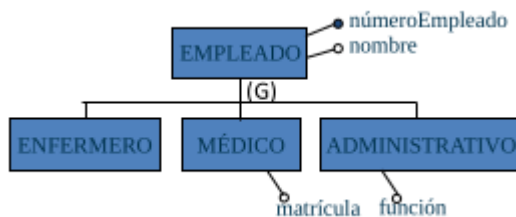


ENFERMERO(**númeroEmpleado**, nombre)

MÉDICO(**númeroEmpleado**, nombre, matrícula)

ADMINISTRATIVO(**númeroEmpleado**, nombre, función)

- Una tabla para el conjunto de entidades de nivel más alto, y una tabla para cada conjunto de entidades del nivel más bajo



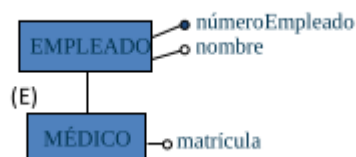
EMPLEADO(**númeroEmpleado**, nombre)

ENFERMERO(**númeroEmpleado**)

MÉDICO(**númeroEmpleado**, matrícula)

ADMINISTRATIVO(**númeroEmpleado**, función)

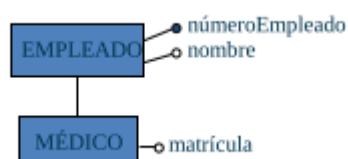
- Especialización (dos estrategias)
 - Una tabla para el conjunto de entidades de nivel más alto



EMPLEADO(**númeroEmpleado**, nombre, tipoEmpleado, matrícula)

Esta opción tiene la principal desventaja de tener que manejar un tipo para distinguir que tipo de empleado es, en este ejemplo.

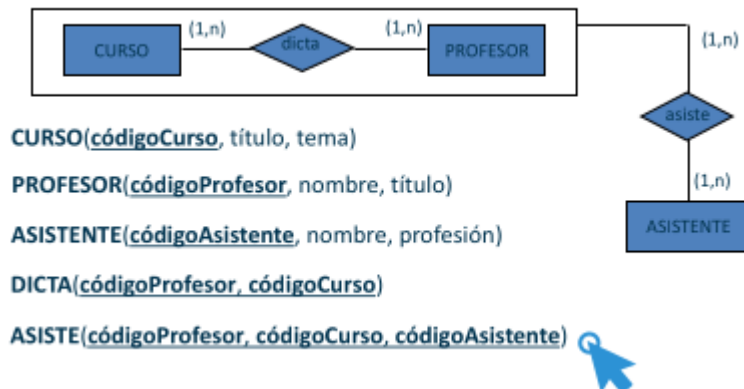
- Una tabla para el conjunto de entidades de nivel más alto, y una tabla para cada conjunto de entidades del nivel más bajo.



EMPLEADO(númeroEmpleado, nombre)

MÉDICO(númeroEmpleado, matrícula)

- Agregación
 - Todas las Entidades y Relaciones involucradas, marcando claves según cardinalidad
 - Foco en los atributos de la relación con la agregación



Clase 2

Motivación

Uno de los principales principios del diseño de bases de datos es la **Normalización**.

- Organiza los datos siguiendo reglas
 - Minimiza redundancia
 - Reduce anomalías
- Puede mejorar la mantenibilidad y según el caso, la performance
Un mal diseño puede implicar refactorizar la base de datos
- Requiere mucho tiempo y expertise
- Deuda técnica

El diseño de un esquema de bases de datos es uno de los factores más importantes de los que depende el éxito de la base de datos relacional.

La normalización es un paso clave en el diseño de una base de datos

- Toma una **relación grande** como entrada y la **descompone** en relaciones más pequeñas las cuales están libres de redundancia de datos y otras anomalías como la inserción/eliminación
 - La descomposición se realiza siguiendo reglas/pasos
- Puede ser manual o automática
Normalización se aplica a bases de datos relacionales
- El modelo relacional, sigue siendo el modelo dominante en la industria.

Síntesis

- El éxito de una base de datos relacional depende del diseño de su esquema
- Existen procesos manuales y automáticos
- Con impacto en la industria
- Tema relevante y actual
- El modelo relacional domina el mercado

Teoría de Diseño de Bases de Datos Relacionales

Conceptos generales

- Anomalía
- Dependencia Funcional
- Dependencia Funcional Trivial

Anomalía

- Problema que surge a raíz del diseño de una relación

Anomalía de redundancia

- Información que se repite innecesariamente en diferentes tuplas

Anomalía de actualización

- Se puede actualizar el valor de una tupla, sin actualizar los de otras tuplas

Anomalías de inserción

Insertar valores en ciertos atributos de una relación y no otros me produce valores nulos.

Anomalías de borrado

Borrar ciertos valores de una tupla, puede llevarme a perder la información de la tupla completa.

Dependencia Funcional

- Es una restricción entre subconjuntos de atributos de una relación.

Si dos tuplas (t_1 y t_2) de una relación R , coinciden en todos los atributos A_1, A_2, \dots, A_n ; entonces DEBEN también coincidir en los atributos B_1, B_2, \dots, B_m . Para toda tupla de R .

- Esto se escribe: $A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_m$
- Y se lee: A_1, A_2, \dots, A_n "determina funcionalmente a" B_1, B_2, \dots, B_m
 Cuando en R se cumple una d , estamos indicando una restricción sobre toda la relación R y no sólo sobre algunas tuplas de R .

Dicho de otra manera:

- Una dependencia funcional de la forma $X \rightarrow Y$ se cumple en R si:
 - Para todos los pares de tuplas t_1 y t_2 de la relación, cuando se cumple que $t_1[x]=t_2[x]$
 - Entonces se cumple $t_1[y]=t_2[y]$.

Ejemplo 1:

- Dada la relación
 - PERSONA(dni, nombre, edad, fechaNacimiento)
- Y valga en persona la
 - df: dni \rightarrow nombre, edad, fechaNac
- La df enunciada, indica que, si dos tuplas t_1 y t_2 de la relación PERSONA tienen el mismo valor en el atributo dni, deben necesariamente tener los mismos valores en los atributos nombre, edad y fechaNac.

Ejemplo 2:

Dada la relación: PERSONA(dni, nombre, edad, fechaNac, nroLegajo)

Donde

- Una persona posee un único número de legajo asignado
- Un número de legajo pertenece a una sola persona
- Se pueden enunciar las siguientes dfs
 - df1) dni \rightarrow nombre, edad, fechaNac, nroLegajo
 - df2) nroLegajo \rightarrow nombre, edad, fechaNac, dni

Dependencia Funcional Trivial

- Caso especial para una dependencia funcional

Es una df de la forma:

$A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_m$

Tal que:

$\{B_1, B_2, \dots, B_m\} \subseteq \{A_1, A_2, \dots, A_n\}$

Ejemplo:

Dada la relación:

- CONTRATADO(nroContratado, dni, nombrePersona, inicioActividad)

Donde valen las siguientes dependencias funcionales:

- df1) dni \rightarrow nombrePersona
- df2) nroContratado, dni \rightarrow inicioActividad

Algunas dependencias funcionales triviales válidas en CONTRATADO son:

- dft1) dni \rightarrow dni

- dft2) nroContratado, dni -> nroContratado

Clase 3

Clave de una relación

Los atributos $\{A_1, A_2, \dots, A_n\}$ son la clave de una relación R si cumplen:

- $\{A_1, A_2, \dots, A_n\}$ determinan funcionalmente a todos los restantes atributos de la relación R
- No existe un subconjunto de $\{A_1, A_2, \dots, A_n\}$ que determine funcionalmente a todos los restantes atributos de la relación R
 - Esto implica que la clave es un conjunto minimal

Ejemplo:

PERSONA(dni, nombre, edad, fechaNac)

- df1: dni->nombre,edad,fechaNac

- Clave: {dni}

Clave candidata de una relación

En caso de existir dos o más subconjuntos de atributos $\{A_1, A_2, \dots, A_n\}$, $\{B_1, B_2, \dots, B_k\}$, ..., $\{N_1, N_2, \dots, N_m\}$ en una relación R tales que:

- $\{A_1, A_2, \dots, A_n\}$ determinan funcionalmente a todos los atributos restantes de la relación R
- $\{B_1, B_2, \dots, B_k\}$, ... Y $\{N_1, N_2, \dots, N_m\}$ también por si mismos determinan al resto de los atributos de R
- No existe subconjunto de $\{A_1, A_2, \dots, A_n\}$ o $\{B_1, B_2, \dots, B_k\}$, o, $\{N_1, N_2, \dots, N_m\}$ que determine funcionalmente a todos los atributos de R
- Entonces $\{A_1, A_2, \dots, A_n\}$, $\{B_1, B_2, \dots, B_k\}$, ..., $\{N_1, N_2, \dots, N_m\}$ son CLAVES CANDIDATAS para la relación R

Clave candidata / de una relación:

Ejemplo 1:

- Dada la relación: PERSONA(dni, nombre, edad, fechaNacimiento, nroLegajo)
Donde
 - Una persona posee un único número de legajo asignado
 - Un número de legajo pertenece a una sola persona
 - Se pueden enunciar las siguientes dfs
 - df1) dni -> nombre, edad, fechaNac, nroLegajo
 - df2) nroLegajo -> nombre, edad, fechaNac, dni
- ¿Clave o claves candidatas?

Clave candidata 1 (cc1): {dni}

Clave candidata 2 (cc2): {nroLegajo}

Ejemplo 2:

- Dada la relación: PERSONA(dni, nombre, edad, fechaNacimiento, nroLegajo, carrera)
 - Donde
 - Una persona puede cursar diversas carreras
 - Nombre indica como se llama la persona
 - Una persona posee un único número de legajo asignado para cada carrera que cursa
 - Un número de legajo pertenece a una sola persona de una carrera
 - df1) dni -> nombre, edad, fechaNac
 - df2) nroLegajo, carrera -> dni
 - df3) dni, carrera -> nroLegajo
- ¿Clave o claves candidatas?
- Clave candidata 1 (cc1): {nroLegajo, carrera }
- Clave candidata 2 (cc2): {dni, carrera }

Super clave de una relación

Los atributos {A1,A2,...,An} son superclave de una relación R si cumplen:

- {A1,A2,...,An} determinan funcionalmente los restantes atributos de la relación R
- Notar que:
 - Una clave está contenida en una superclave
 - Una superclave no necesariamente es minimal (como lo es la clave por la segunda condición de su definición)

Ejemplo:

- PERSONA(dni, nombre, edad, fechaNacimiento)
 - df1: dni->nombre,edad,fechaNac
- superclave: {dni, nombre}

Axiomas de Armstrong

- Permiten inferir nuevas dependencias funcionales dado un conjunto base que resultó evidente
- Aplicándolos halla un conjunto completo y seguro donde todas las dependencias funcionales halladas son correctas
- Al generar todas las dependencias funcionales algunas son triviales

Axiomas Básicos

- Reflexión
- Aumento

- Transitividad
Axiomas que se deducen a partir de los básicos
- Unión
- Descomposición
- Pseudotransitividad

Reflexión

X es un conjunto de atributos e $Y \subseteq X$ entonces $X \rightarrow Y$

Demostración:

Si $Y \subseteq X$ y existen dos tuplas diferentes de R tales que $t_1[x] = t_2[x]$ por definición de dependencia funcional $t_1[y] = t_2[y]$

Aumento

Si $X \rightarrow Y$;

Z es un conjunto de atributos,
entonces

$Z, X \rightarrow Z, Y$

Demostración:

Asumamos que $X \rightarrow Y$ vale pero $X, Z \rightarrow Y, Z$ no vale

Si $X \rightarrow Y$ entonces cada vez que

1. $t_1[x] = t_2[x]$ implica

2. $t_1[y] = t_2[y]$

Por otro lado, cada vez que

3. $t_1[x, z] = t_2[x, z]$ implica

4. $t_1[y, z] \neq t_2[y, z]$

De 1) y 3) se deduce $t_1[z] = t_2[z]$

De 2) y 4) se deduce que $t_1[y, z] = t_2[y, z]$

Transitividad

Si $X \rightarrow Y$;

$Y \rightarrow Z$,

entonces $X \rightarrow Z$

Demostración:

1. $X \rightarrow Y$

2. $Y \rightarrow Z$

$t_1[x] = t_2[x]$ implica por 1)

$t_1[y] = t_2[y]$ implica por 2)

$t_1[z] = t_2[z]$ entonces

$X \rightarrow Z$

Unión

Si $X \rightarrow Y$;

$X \rightarrow Z$,

entonces $X \rightarrow Y, Z$

Demostración:

1. $X \rightarrow Y$

2. $X \rightarrow Z$

Si $X \rightarrow Y$, por aumentación vale que $X \rightarrow XY$

Si $X \rightarrow Z$, por aumentación vale que $X, Y \rightarrow Y, Z$

Luego por transitividad, $X \rightarrow Y, Z$

Descomposición

Si $X \rightarrow Y, Z$

entonces $X \rightarrow Y$, $X \rightarrow Z$

Demostración:

$X \rightarrow Y, Z$

por reflexividad vale que $Y, Z \rightarrow Y$

Luego, por transitividad $X \rightarrow Y$

Por reflexividad también vale que $Y, Z \rightarrow Z$

Luego por transitividad, también vale que $X \rightarrow Z$

Pseudotransitividad

Si $X \rightarrow Y$; $Y, Z \rightarrow W$ entonces $X, Z \rightarrow W$

Demostración:

$X \rightarrow Y$

por aumento vale que $X, Z \rightarrow Y, Z$

Por otro lado, se sabe que $Y, Z \rightarrow W$

Luego por transitividad, vale que $X, Z \rightarrow W$

Clausura de un conjunto de atributos

X^+

- Sea F un conjunto de dependencias funcionales sobre un esquema R y,
- Sea X un subconjunto de R

La clausura del conjunto de atributos X respecto de F , se denota X^+ y es el conjunto de atributos de A tal que la dependencia $X \rightarrow A$ puede deducirse a partir de F , por los axiomas de Armstrong.

Es decir, X^+ , son todos los atributos determinados por X en R .

Algoritmo para hallar X^+


```

Result:= X
While (hay cambios en result)
do
For (cada dependencia
funcional Y->Z en F ) do
if (Y  $\subseteq$  result) then
result := result u Z

```

¿Como funciona?

Ejemplo:

Dada la relación: PERSONA(dni, nombre, edad, fechaNacimiento, nroLegajo, carrera)

Donde

- Una persona puede cursar diversas carreras
- Nombre indica como se llama la persona
- Una persona posee un único número de legajo asignado para cada carrera que cursa
- Un número de legajo pertenece a una sola persona de una carrera
 - df1) dni -> nombre, edad, fechaNac
 - df2) nroLegajo, carrera -> dni
 - df3) dni, carrera -> nroLegajo

Clave candidata 1 (cc1): {nroLegajo, carrera }

Clave candidata 2 (cc2): {dni, carrera}

Por ejemplo, podríamos preguntarnos: ¿Es cierto que a partir de los atributos de cc1, puedo recuperar los atributos restantes de PERSONA?

- Para ello podemos ejecutar el algoritmo de X+ instanciándolo con la información de PERSONA

Hallar (nroLegajo, carrera)⁺

Result= (nroLegajo, carrera)

Paso 1) Tomamos la dep. fun: dni -> nombre, edad, fechaNac, {dni} no está incluido en result, no

agrego nada a result => (nroLegajo, carrera)

Paso 2) Tomamos la dep. fun. nroLegajo, carrera -> dni , {nroLegajo, carrera} está incluido en result,

agrego {dni} a result => (nroLegajo, carrera, dni)

Paso 3) Tomamos la dep. fun. dni, carrera -> nroLegajo, {dni, carrera} está incluido en result, agrego

{nroLegajo} a result => (nroLegajo, carrera, dni)

Como ya recorrí todas las dependencias funcionales y result cambió vuelvo a iterar

Paso 1) Tomamos la dep. fun. dni -> nombre, edad, fechaNac, {dni} está incluido en result, agrego

{nombre, edad, fechaNac} a result => (nroLegajo, carrera, dni , nombre, edad, fechaNac)

Important

- De la misma manera podríamos haber probado con la otra clave candidata
- Recordar que este algoritmo NO asegura que el conjunto de atributos de partida sea mínimo

Hasta ahora vimos, para una relación R

- Cómo hallar la o las claves candidatas
 - Y como usar el algoritmo de la clausura de atributos para corroborar que a partir de un subconjunto de atributos de R se puede recuperar al resto de los atributos de la relación (aunque éste no asegura que dicho subconjunto sea mínimo)

- Cómo hallar dependencias funcionales

- Y su conjunto completo mediante los Axiomas de Armstrong

A continuación:

- Considerando las dependencias funcionales y las claves candidatas, veremos conceptos necesarios para realizar un proceso que permite generar relaciones que cumplan ciertas condiciones de un buen diseño (quitando anomalías) con el fin de normalizar esquemas

Normalización de esquemas de relación.

Algunos conceptos

¿Cómo generar relaciones que cumplan ciertas condiciones de un buen diseño?

Descomposición o particionamiento de un esquema

- Es una forma aceptada de eliminar las anomalías de una relación
- Consiste en separar los atributos de una relación en dos nuevas relaciones (bajo ciertos criterios)
- Al particionar, no se debe perder
 - Información
 - Dependencias funcionales

Descomposición de un esquema

Dado un esquema R donde vale una dependencia funcional $X \rightarrow Y$ R se descompone/particiona en

- $R_1(X, Y)$

- $R^2(R-Y)$
Al descomponer, no se debe perder:
- Información
- Dependencias funcionales
 - Validación simple
 - Validación formal mediante un algoritmo

Pérdida de información

Si a un esquema R , se lo particiona en dos subesquemas R_1 y R_2 entonces, se debe cumplir alguna de las siguientes condiciones:

- $R_1 \cap R_2$ clave en el esquema R_1
o bien,
- $R_1 \cap R_2$ clave en el esquema R_2

Pérdida de dependencias funcionales

Al particionar, se debe verificar que cada una de las dependencias funcionales que valían en el esquema R , sigan valiendo en alguna de las particiones R_i .

Cuando se chequean las dependencias funcionales pueden ocurrir dos cosas:

- los atributos de la dependencia funcional original quedaron todos incluidos en alguna de las particiones generadas.
=> Validación simple
- Los atributos de la dependencia funcional original quedaron distribuidos en más de una partición
=> Validación formal mediante un algoritmo

Formas normales

- Criterio para determinar grado de vulnerabilidad a inconsistencias y anomalías
- Existen diferentes formas normales
- Al lograr aplicar una mayor forma normal se logrará menor vulnerabilidad

Algunas formas normales

- ^[1]Primera Forma Normal (1FN)
- ^[2]Segunda Forma Normal (2FN)
- ^[2-1]Tercera Forma Normal (3FN)
- ^[2-2]Forma Normal de Boyce y Codd

Forma Normal de Boyce y Codd (BCNF)

- Conocida por su acrónimo en inglés de BCNF

- Particionar para llevar un esquema a esta FN, asegura que:
 - las anomalías dejan de estar (sólo puede quedar redundancia),
 - que no se pierda información y,
 - en algunos casos, asegura que no se pierdan dependencias funcionales

Un esquema de relación está en BCNF sí, siempre que una dependencia funcional de la forma $X \rightarrow A$ es válida en R, entonces se cumple que:

- X es superclave de R (mínima)
o bien
- $X \rightarrow A$ es una dependencia funcional trivial

- Hasta ahora vimos que
 - Las formas normales se usan para sacar anomalías
 - A mayor forma normal, menos vulnerabilidad
- A continuación
 - Intentaremos llevar, en principio, un esquema a la Forma Normal de Boyce y Codd para normalizarlo

Para normalizar un esquema (en principio) y propiciar un buen diseño, vamos a valernos de:

- las dependencias funcionales del esquema
- las claves candidatas
- la definición de BCNF
- la descomposición o particionamiento del esquema

Proceso de normalización

Cómo llevar un esquema R a BCNF (cuando se puede)

Hallar dependencias funcionales y claves candidatas

1. Analizar si en el esquema R existe alguna dependencia funcional que lleva al esquema a no cumplir con la definición de BCNF
 1. Si existe tal dependencia funcional, particionar el esquema en dos nuevos esquemas R_i , R_{i+1} , contemplando la dependencia funcional en cuestión.
Analizar las 2 particiones generadas preguntándose:
 - 1) Se pierde información?
 - 1) NO, entonces sigo a 1.1.2
 - 2) SI. La partición es errónea. Re analizar
 - 2) Se pierden Dependencias funcionales?
 - 1) NO, entonces sigo a 1.1.3
 - 2) Si. Veremos este caso en breve

- 3) Determinar en qué forma normal esta R_i , R_{i+1} , si no están en BCNF, reiniciar desde 1, sino pasar a 1.2
2. Si NO existe, el esquema está en BCNF

Clase 4

Forma Normal de Boyce y Codd (BCNF)

Un esquema de relación está en BCNF sí, siempre que una dependencia funcional de la forma $X \rightarrow A$ es válida en R , entonces se cumple que:

- X es superclave de R
o bien
- $X \rightarrow A$ es una dependencia funcional trivial

En general, al llevar un esquema a BCNF

Particionar llevando a un esquema en BCNF

- Las anomalías dejan de estar (sólo puede quedar redundancia)
- Que no se pierda información
- En algunos casos, asegura que no se pierdan dependencias funcionales

Para normalizar un esquema (en principio) y propiciar un buen diseño, vamos a valernos de:

- las dependencias funcionales del esquema
- las claves candidatas • la definición de BCNF
- la descomposición o particionamiento del esquema

Cuando no se puede llevar a BCNF porque se pierden dependencias funcionales, entonces, se lleva el esquema a 3FN

En este escenario, la 3FN, lo que asegura es que:

- no se pierde información,
- no se pierdan dependencias funcionales, pero no siempre se quitan las anomalías.

Tercera forma Normal 3FN

Un esquema de relación R está en 3FN si para toda dependencia de la forma $X \rightarrow A$, donde A no está incluida en X , se cumple que:

- X es superclave
O bien
- A es primo

Atributo primo:

atributo que forma parte de alguna clave candidata

¿Cómo se lleva un esquema a 3FN cuándo no se puede llevar a BCNF porque se pierden dependencias funcionales?

- Se construye una tabla por cada dependencia funcional
- Si la clave de la tabla original no está incluida en ninguna de las tablas del punto anterior, se construye una tabla con la clave

Ejemplo:

LIBROS (titulo, teatro, ciudad)

Valen las siguientes dependencias funcionales

df1) teatro → ciudad

df2) titulo, ciudad → teatro

Claves candidatas:

cc1: {titulo, ciudad}

cc2: {teatro, titulo}

Como no es posible llevar el esquema a BCNF sin perder dependencias funcionales, entonces, aplico el proceso para dejar el esquema en 3FN.

A (**teatro**, ciudad)

B (**titulo, ciudad**, teatro)

(en este caso, como una de las claves quedó en una de las particiones, no se agrega una nueva partición con ella)

Las particiones A y B están en 3FN

Síntesis

Llamamos normalizar hasta BCNF o 3FN, al proceso que involucra, hasta ahora, los siguientes pasos:

- Encontrar las dependencias funcionales y las claves candidatas
- Llevar a BCNF aplicando el proceso de descomposición/particionamiento sin pérdida de información
- Comprobar que no se pierden dependencias funcionales en la descomposición
- No, continuo el proceso a BCNF
- Si, se lleva el esquema correspondiente a 3NF

Dejar un esquema en la 3FN o en BCNF no asegura eliminar la redundancia

Otras formas normales

1FN

- Los atributos de la relación son simples y atómicos
- 2FN:
- Un esquema de relación R está en 2FN si para toda dependencia de la forma $X \rightarrow A$, se cumple que: A depende de manera total de la clave.

Dependencia Multivaluada (DM)

Se puede decir que:

$X \twoheadrightarrow Y$

si:

dado un valor de X, hay un conjunto de valores de Y asociados y este conjunto de valores de Y NO está relacionado (ni funcional ni multifuncionalmente) con los valores de R - X - Y (donde R es el esquema), es decir, Y es independiente de los atributos de R-X-Y.

Dependencia Multivaluada Trivial (DM Trivial)

Sea R un esquema de relación

Una dependencia multivaluada de la forma

$X \twoheadrightarrow Y$

que vale en R es trivial si:

- el conjunto de atributos X,Y (X unión Y) son todos los atributos del esquema R

Caso especial DM

Sea R un esquema de relación

Es posible definir una

dependencia multivaluada de la forma

$\emptyset \twoheadrightarrow Y$

Y se lee:

vacío multidetermina a Y

Cuarta Forma Normal 4FN

Un esquema R está en 4NF con respecto a un conjunto de dependencias multivaluadas D, si \forall dependencia multivaluada (DM) de la forma

$X \twoheadrightarrow Y$

se cumple que:

- $X \twoheadrightarrow Y$ es una DM trivial

En otras palabras:

Un esquema R está en 4FN cuando:

- No tiene dependencias multivaluadas o bien,

- Las dependencias multivaluadas que en él valen, son triviales.

Resumiendo - Proceso de Normalización hasta 4FN

1. Encontrar las dependencias funcionales
2. Encontrar las claves candidatas
3. Verificar si el esquema cumple con la definición de BCNF, si no se cumple descomponer la relación sin perder información ni dependencias funcionales
 - Si se pierden dependencias funcionales, llevar a 3NF
 - Esto hasta dejar las particiones en BCNF o 3FN
4. Expresar en este punto, cual es la clave primaria y que particiones quedaron en BCNF o 3FN (según corresponda)
5. Encontrar las dependencias multivaluadas sobre la última partición realizada (aquella que tiene la clave primaria del esquema) y verificar 4NF, si no se cumple dividir la relación
 - Esto se hace hasta dejar las particiones del esquema en 4FN
6. Expresar las particiones resultantes que quedaron en 4FN o Explicar porque las particiones descritas en el ítem 4 (excepto la analizada en el punto 5) quedaron en 4FN
7. Indicar que particiones en 4FN quedan en el esquema final (que no sean proyecciones de atributos claves de otras particiones en 4FN)

Clase 5

Álgebra Relacional

- Lenguaje algebraico donde:
 - Los cambios de estado se especifican mediante operaciones.
 - Los operandos son relaciones y el resultado es una nueva relación
- Proporciona una base teórica
- Lenguaje de consulta
 - Operaciones fundamentales
 - Operaciones adicionales
- Lenguaje de manipulación
 - Operaciones de manipulación

Lenguaje de consultas

Operaciones fundamentales

- Son suficientes para expresar cualquier consulta en álgebra relacional
 - Selección (σ)
 - Proyección (π)
 - Producto Cartesiano (\times)

- Renombre (ρ)
 - De una relación
 - De atributos de una relación
- Unión (\cup)
- Diferencia ($-$)

Selección σ

- Operación unaria ($\sigma_{condición} R$)
- Requiere una condición booleana
 - Operaciones: and, or y not
- El resultado es una relación con un subconjunto "horizontal" de la relación dada
- Ejemplo:

Ingenieros

E#	Nombre	Edad
320	José	34
322	Rosa	37
323	María	25

$\sigma_{edad \geq 30} (Ingenieros)$

E#	Nombre	Edad
320	José	34
322	Rosa	37

•

Proyección Π

- Operación unaria ($\Pi_{lista_de_atributos} R$)
- Dada una lista de atributos produce un corte "vertical" de la relación
 - Los atributos de la lista se toman de izquierda a derecha.
- Ejemplo:

Ingenieros

E#	Nombre	Edad
320	José	34
322	Rosa	37
323	María	25

$\Pi_{nombre, edad} (Ingenieros)$

Nombre	Edad
José	34
Rosa	37
María	25

•

Producto Cartesiano \times

- Operación binaria ($A \times B$)
- El resultado es una relación que incluye todas las tuplas posibles que se obtienen concatenando cada tupla de A con cada una de las tuplas de B
- Ejemplo

Ingenieros			Proyectos	
E#	Nombre	D#	Proyecto	Tiempo
320	José	D1	RX338A	21
322	Rosa	D3	PY254Z	32

Ingenieros X Proyectos

E#	Nombre	D#	Proyecto	Tiempo
320	José	D1	RX338A	21
320	José	D1	PY254Z	32
322	Rosa	D3	RX338A	21
322	Rosa	D3	PY254Z	32

•

Renombre de una relación ρ

- Operación unaria ($\rho_x R$)
- El resultado es la relación R con nombre X y atributos nombrados como se expresa en lista_de_atributos
- Ejemplo:

Ingenieros		
E#	Nombre	Edad
320	José	34
322	Rosa	37
323	María	25

ρ

PROFESIONALES(E#, NombreProfesional, Edad)

Ingenieros

PROFESIONALES

E#	NombreProfesional	Edad
320	José	34
322	Rosa	37
323	María	25

•

Unión U

- Operación binaria ($A \cup B$)
- El resultado es una relación en la que se agrega a la relación A los elementos (no repetidos) de la relación B
- Es necesario que las operaciones A y B sean de **unión compatible**
 - Relaciones con igual aridad (igual número de atributos)
 - El dominio del i-ésimo atributo de ambas relaciones debe ser el mismo ($\forall i$)
- Ejemplo

Ingenieros

E#	Nombre	Edad
320	José	34
322	Rosa	37
323	María	25

Jefes

E#	Nombre	Edad
320	José	34
421	Jorge	48

Ingenieros \cup Jefes

E#	Nombre	Edad
320	José	34
322	Rosa	37
323	María	25
421	Jorge	48

•

Diferencia -

- Operación binaria (A B).
- El resultado es una relación donde están los elementos que pertenecen a A y no pertenecen a B.
- Es necesario que las relaciones A y B sean de «unión compatible»
 - Relaciones con igual aridad (igual número de atributos)
 - El dominio del i-ésimo atributo de ambas relaciones debe ser el mismo ($\forall i$)
- Ejemplo

Ingenieros

E#	Nombre	Edad
320	José	34
322	Rosa	37
323	María	25

Jefes

E#	Nombre	Edad
320	José	34
421	Jorge	48

Ingenieros - Jefes

E#	Nombre	Edad
322	Rosa	37
323	María	25

•

Operaciones adicionales

- No agregan potencia al álgebra, simplifican consultas.
- Son reescribibles en término de operaciones fundamentales
 - Intersección (\cap)
 - Producto Theta ($|X|_{\theta}$)
 - Producto Natural ($|X|$)
 - División ($\%$)

- Operación especial de Asignación (\leftarrow)

Intersección

- Operación binaria ($A \cap B$)
- El resultado es una relación con aquellas tuplas que pertenecen a ambas relaciones (al mismo tiempo)
- Es necesario que las relaciones A y B sean unión compatible
- $R \cap S$ es equivalente a $R - (R - S)$
- Ejemplo

Ingenieros			Jefes		
E#	Nombre	Edad	E#	Nombre	Edad
320	José	34	320	José	34
322	Rosa	37	421	Jorge	48
323	María	25			

Ingenieros \cap Jefes

E#	Nombre	Edad
320	José	34

•

Producto Theta $|X|_{\theta}$

- Operación binaria ($R |X|_{\theta} S$)
- Genera una nueva relación con las tuplas resultantes de aplicar una operación de selección con la condición indicada por θ sobre el resultado de un producto cartesiano
- La condición (θ) se indica como una expresión booleana de términos (se pueden usar conectores lógicos entre las condiciones)
- $R |X|_{\theta} S$ es equivalente a $\sigma_{\theta}(R \times S)$
- Ejemplo

Ingenieros			Jefes		
E#	Nombre	Edad	E#	Nombre	Edad
320	José	34	320	José	34
322	Rosa	37	421	Jorge	48
323	María	25			

Ingenieros $|X|_{\text{Ingenieros.edad = Jefes.edad}}$ Jefes

Ingenieros.E#	Ingenieros.Nombre	Ingenieros.Edad	Jefes.E#	Jefes.Nombre	Jefes.Edad
320	José	34	320	José	34

•

Producto Natural $|X|$

- Operación binaria ($R |X| S$)

- Genera una nueva relación con las tuplas resultantes de aplicar una operación de selección con la condición indicada sobre el resultado de un producto cartesiano
- La condición se indica como una expresión booleana de términos (se pueden usar conectores lógicos entre las condiciones)
- $R \mid X \mid S$ es equivalente a
- $\Pi_{lista}(\sigma_{condición}(R \times S))$
 - Donde:
 - condición implica a todos los atributos de R que están en S y son iguales
 - lista elimina columnas repetidas (dejando una sola en el conjunto) y los atributos que no tienen en común R y S
- Ejemplo:

Postulantes			Administrativos			
Nombre	Edad	DNI	Nombre	Edad	Domicilio	DNI
Paula	19	29235142	Martina	22	1 y 50	35215415
Martina	22	35215415	Paula	19	8 y 49	29899632
Joaquín	28	28152478	Pablo	32	26 y 50	20125789

Postulantes | X | Administrativos

Nombre	Edad	DNI	Domicilio
Martina	22	35215415	1 y 50

•

División %

- Operación binaria ($R \% S$)
 - R dividendo
 - S divisor
- Los atributos del divisor S deben ser un subconjunto de los atributos de la relación R con igual dominio
- La relación resultante de la división llamemosla T, posee tuplas t tal que:
 - Los valores de t deben aparecer en R en combinación con todas las tuplas de S
- $R \% S$ es equivalente a:
 - $\Pi_{att(R)-att(S)}R - \Pi_{att(R)-att(S)}((\Pi_{att(R)-att(S)}(R) \times S) - R)$
 - Donde:
 - $att(R) - att(S)$ significan los atributos de la relación R menos los atributos de la relación S
- Ejemplos

R1

E#	Proyecto
320	RX338A
320	PY254Z
323	RX338A
323	PY254Z
323	NC168T
324	NC168T
324	KT556B

R2

Proyecto
RX338A
PY254Z

R1 % R2

E#
320
323

•

Lugar_Trabajo

Nombre	Sucursal	Color
Juan	Sucursal1	Rojo
Pedro	Sucursal1	Verde
Juan	Sucursal2	Azul
María	Sucursal 1	Rojo
Juan	Sucursal3	Violeta
Pedro	Sucursal1	Rojo
Pedro	Sucursal2	Azul
Juan	Sucursal1	Verde

Sucursales_Vip

Sucursal	Color
Sucursal1	Rojo
Sucursal2	Azul
Sucursal1	Verde

Lugar_Trabajo % Sucursales_Vip

Nombre
Juan
Pedro

•

Asignación ←

- Es una forma conveniente de expresar operaciones complejas
 - Modularidad
- El resultado de una operación se asigna temporalmente a una variable
 - La variable a la cual se asigna el resultado de una operación se puede usar en otras operaciones
- Ejemplo

Lugar_Trabajo

Nombre	Sucursal
Juan	Sucursal1
Pedro	Sucursal1
Juan	Sucursal2
María	Sucursal 1
Juan	Sucursal3

Sucursales_Vip

Sucursal
Sucursal1
Sucursal2

Empleado

Nombre	Domicilio
Juan	1 y 50
Pedro	120 y 43
María	150 y 62

Empleados_Vip \leftarrow Lugar_Trabajo % Sucursales_Vip

Empleados_Vip |X| Empleado

Nombre
Juan

Nombre	Domicilio
Juan	1 y 50

•

En general

- Combinación de operaciones para formar consultas
 - Las operaciones se pueden usar
 - Aisladas o
 - Combinadas (expresiones)
 - Permiten resolver consultas complejas
 - Se usan paréntesis cuando es necesario agrupar operaciones
 - Notación lineal

Lenguaje de Manipulación de Datos

Operaciones de manipulación

- Requieren de la operación de asignación
- Modifican la cantidad o los valores de las tuplas de una relación

Inserción \cup

- La o las tuplas a insertar deben ser compatibles con la relación
- $R \leftarrow R \cup E$
 - Donde
 - R es la relación en la que se insertarán los resultados de la expresión E
- Ejemplo

Empleado	Nombre	Domicilio
	Juan	1 y 50
	Pedro	120 y 43
	María	150 y 62

$\text{Empleado} \leftarrow \text{Empleado} \cup \{("Joaquín", "4 y 497")\}$

Empleado	Nombre	Domicilio
	Juan	1 y 50
	Pedro	120 y 43
	María	150 y 62
	Joaquín	4 y 497

•

Empleado

Nombre	Domicilio
Juan	1 y 50
Pedro	120 y 43
María	150 y 62

Asistentes

Nombre	Domicilio	DNI
Joaquín	4 y 497	1234536
Martina	1 y 32	2541258

$\text{Empleado} \leftarrow$
 $\text{Empleado} \cup$
 $(\Pi_{\text{nombre, domicilio}} \text{Asistentes})$

Empleado

Nombre	Domicilio
Juan	1 y 50
Pedro	120 y 43
María	150 y 62
Joaquín	4 y 497
Martina	1 y 32

•

Eliminación –

- La o las tuplas a insertar deben ser compatibles con la relación
- $R \leftarrow R - E$
 - Donde
 - R es la relación en la que se insertarán los resultados de la expresión E
- Ejemplo

Empleado

Nombre	Domicilio
Juan	1 y 50
Pedro	120 y 43
María	150 y 62
Joaquín	4 y 497
Martina	1 y 32

$\text{Empleado} \leftarrow \text{Empleado} -$
 $\{("Joaquín", "4 y 497"), ("Martina", "1 y 32")\}$

Empleado

Nombre	Domicilio
Juan	1 y 50
Pedro	120 y 43
María	150 y 62

•

Empleado

Nombre	Domicilio
Juan	1 y 50
Pedro	120 y 43
María	150 y 62
Joaquín	4 y 497
Martina	1 y 32

Algunos_empleados ←

$\sigma_{\text{nombre}=\text{"Martina"} \text{ o nombre}=\text{"Joaquín"}}(\text{Empleado})$

Empleado ← Empleado – Algunos_empleados

Empleado

Nombre	Domicilio
Juan	1 y 50
Pedro	120 y 43
María	150 y 62

•

Actualización δ

- Permite actualizar el valor particular de una tupla
- $\delta_{att(R) \leftarrow E}(R)$
 - Donde
 - R es la relación a la que se le modificará el atributo mencionado en att(R), como resultado de la expresión E
- Ejemplo

Empleado

Nombre	Domicilio	Salario
Juan	1 y 50	10200
Pedro	120 y 43	15000
María	150 y 62	22000

$\delta_{\text{salario} \leftarrow \text{salario} * 1.2}(\text{Empleado})$

Empleado

Nombre	Domicilio	Salario
Juan	1 y 50	12240
Pedro	120 y 43	18000
María	150 y 62	26400

•

Empleado

Nombre	Domicilio	Salario
Juan	1 y 50	10200
Pedro	120 y 43	15000
María	150 y 62	22000

δ salario \leftarrow salario * 1.2 ($\sigma_{\text{nombre}=\text{"Juan"}}$ (Empleado))

Empleado

Nombre	Domicilio	Salario
Juan	1 y 50	12240
Pedro	120 y 43	15000
María	150 y 62	22000

•

-
1. El esquema no debe tener atributos polivalentes o compuestos \leftrightarrow
 2. Se determinan a partir de las dependencias funcionales $\leftrightarrow \leftrightarrow \leftrightarrow$