CC3201-1

Bases de Datos

Otoño 2023

Clase 7: Actualizaciones, Restricciones, Formas Normales

Aidan Hogan

aidhog@gmail.com

Las preguntas de hoy

Planeta

nombre	dist	radio	grav	días	años	temp	anillo
Mercurio	0,39	0,38	2,8	58,646	0,241	440	false
Venus	0,72	0,95	8,9	-243,019	0,615	730	false
Tierra	1,00	1,00	9,8	0,997	1,000	288	false
Marte	1,52	0,53	3,7	1,026	1,880	186	false
Júpiter	5,20	10,97	22,9	0,414	11,862	152	true
Saturno	9,54	9,14	9,1	0,444	29,447	134	true
Urano	19,19	3,98	7,8	-0,719	84,017	76	true
Neptuno	30,07	3,86	11,0	0,671	164,791	53	true

Satélite

nombre	planeta	descubridor	año
Luna	Tierra	⊥	⊥
Ganímedes	Júpiter	Galileo Galilei	1610
Calisto	Júpiter	Galileo Galilei	1610
Europa	Júpiter	Galileo Galilei	1610
Ío	Júpiter	Galileo Galilei	1610
Titán	Saturno	Christiaan Huygens	1655
Tritón	Neptuno	William Lassell	1846

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	URRS	1971
Viking 1	Marte	EEUU	1976
Beagle 2	Marte	ESA	2003
Galileo	Júpiter	EEUU	2003

Pero ¿cómo se pueden crear y actualizar las tablas?

Y ¿cómo se puede saber si es un buen diseño relacional o no?

...

SQL: Gestionar y crear tablas

Capítulo 3.1.1 | Ramakrishnan / Gehrke

SQL: Esquema

SistemaSolar

```
CREATE SCHEMA SistemaSolar; -- crear una agrupación de tablas
```

¿Para qué sirven los esquemas?

Podemos configurar agrupaciones de tablas usando esquemas …

SQL: Privilegios de Esquema

SistemaSolar

```
CREATE SCHEMA SistemaSolar; -- crear una agrupación de tablas
GRANT USAGE ON SCHEMA SistemaSolar TO narmstrong; -- sólo lectura
GRANT ALL PRIVILEGES ON SCHEMA SistemaSolar TO csagan; -- todo
```

SQL: Crear tablas

SistemaSolar

Aterrizaje

nave	planeta	país	año

```
CREATE SCHEMA SistemaSolar; -- crear una agrupación de tablas
CREATE TABLE SistemaSolar.Aterrizaje (
  nave VARCHAR (255),
  planeta VARCHAR (255),
  país VARCHAR (255),
  año SMALLINT
);
```

SQL: Borrar tablas

SistemaSolar

```
CREATE SCHEMA SistemaSolar; -- crear una agrupación de tablas
CREATE TABLE SistemaSolar.Aterrizaje (
  nave VARCHAR (255),
  planeta VARCHAR (255),
  país VARCHAR (255),
  año SMALLINT
);
DROP TABLE SistemaSolar.Aterrizaje;
```

¿Hay que poner el esquema cada vez?

...

SQL: Camino de esquema

SistemaSolar

```
CREATE SCHEMA SistemaSolar; -- crear una agrupación de tablas
SHOW search_path; -- public
ALTER USER csagan SET search_path TO SistemaSolar,public;
CREATE TABLE Aterrizaje ( ... );
DROP TABLE Aterrizaje;
```

Seleccionará el primer esquema relevante en el camino.
P. ej., si hay `SistemaSolar.Aterrizaje` *y* `public.Aterrizaje`*, leerá de la primera tabla.*

SQL: Actualizar tablas

Capítulo 3.1.1 | Ramakrishnan / Gehrke

SQL: Insertar tuplas

SistemaSolar

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015

```
...
CREATE TABLE Aterrizaje ( ... );
INSERT INTO Aterrizaje VALUES ('Messenger','Mercurio','EEUU',2015);
```

SQL: Insertar tuplas

SistemaSolar

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966

```
...
CREATE TABLE Aterrizaje ( ... );
INSERT INTO Aterrizaje VALUES ('Messenger','Mercurio','EEUU',2015);
INSERT INTO Aterrizaje VALUES ('Venera 3','Venus','URRS',1966);
```

SQL: Insertar tuplas

SistemaSolar

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978

```
...
CREATE TABLE Aterrizaje ( ... );
INSERT INTO Aterrizaje VALUES ('Messenger','Mercurio','EEUU',2015);
INSERT INTO Aterrizaje VALUES ('Venera 3','Venus','URRS',1966);
INSERT INTO Aterrizaje VALUES ('Pioneer','Venus','EEUU',1978);
```

SQL: Insertar tuplas

SistemaSolar

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978

```
...
CREATE TABLE Aterrizaje ( ... );
INSERT INTO Aterrizaje VALUES ('Messenger','Mercurio','EEUU',2015);
INSERT INTO Aterrizaje VALUES ('Venera 3','Venus','URRS',1966);
INSERT INTO Aterrizaje VALUES ('Pioneer','Venus','EEUU',1978);
INSERT INTO Aterrizaje VALUES ('Mars 2 lander','Marte','URRS','1971'); -- error
```

SQL: Insertar tuplas

SistemaSolar

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	⊥

```
...
CREATE TABLE Aterrizaje ( ... );
INSERT INTO Aterrizaje VALUES ('Messenger','Mercurio','EEUU',2015);
INSERT INTO Aterrizaje VALUES ('Venera 3','Venus','URRS',1966);
INSERT INTO Aterrizaje VALUES ('Pioneer','Venus','EEUU',1978);
INSERT INTO Aterrizaje VALUES ('Mars 2 lander','Marte','URRS','1971'); -- error
INSERT INTO Aterrizaje (país,nave,planeta) VALUES ('EEUU','Mars 2 lander','Marte');
```

SQL: Insertar tuplas

SistemaSolar

AterrizajeEEUU

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	⊥

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	⊥

```
...
CREATE TABLE Aterrizaje ( ... );
INSERT INTO Aterrizaje VALUES ('Messenger','Mercurio','EEUU',2015);
INSERT INTO Aterrizaje VALUES ('Venera 3','Venus','URRS',1966);
INSERT INTO Aterrizaje VALUES ('Pioneer','Venus','EEUU',1978);
INSERT INTO Aterrizaje VALUES ('Mars 2 lander','Marte','URRS','1971'); -- error
INSERT INTO Aterrizaje (país,nave,planeta) VALUES ('EEUU','Mars 2 lander','Marte');
CREATE TABLE AterrizajeEEUU ( ... ); -- misma definición que Aterrizaje
INSERT INTO AterrizajeEEUU ( SELECT * FROM Aterrizaje WHERE país='EEUU' );
```

SQL: Editar tuplas

SistemaSolar

AterrizajeEEUU

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	⊥

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	URRS	1971

```
...
UPDATE Aterrizaje
  SET año=1971,país='URRS'
  WHERE nave='Mars 2 lander';
```

SQL: Actualizar tuplas de otra tabla

SistemaSolar

AterrizajeEEUU

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	⊥

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	1971

```
...
UPDATE Aterrizaje A
  SET país=AE.país
  FROM AterrizajeEEUU AE
  WHERE A.nave=AE.nave AND A.planeta=AE.planeta;
```

SQL: Borrar tuplas

SistemaSolar

AterrizajeEEUU

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Pioneer	Venus	EEUU	1978

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	1971

```
...
DELETE FROM AterrizajeEEUU WHERE año IS NULL;
```

SQL: Borrar columnas

SistemaSolar

AterrizajeEEUU

nave	planeta	año
Messenger	Mercurio	2015
Pioneer	Venus	1978

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	1971

```
...
ALTER TABLE AterrizajeEEUU DROP COLUMN país;
```

SQL: Crear columnas

SistemaSolar

AterrizajeEEUU

nave	planeta	año	despegue
Messenger	Mercurio	2015	⊥
Pioneer	Venus	1978	⊥

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	1971

```
...
ALTER TABLE AterrizajeEEUU ADD COLUMN despegue DATE;
```

SQL: Modificar columnas

SistemaSolar

AterrizajeEEUU

nave	planeta	año	despegue
Messenger	Mercurio	2015	⊥
Pioneer	Venus	1978	⊥

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	1971

```
...
ALTER TABLE AterrizajeEEUU ALTER COLUMN despegue VARCHAR(255);
```

Postgres: Cargar datos

SistemaSolar

Aterrizaje

nave	planeta	país	año
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Messenger	Mercurio	EEUU	2015
Venera 3	Venus	URRS	1966
Pioneer	Venus	EEUU	1978
Mars 2 lander	Marte	EEUU	1971
Viking 1	Marte	EEUU	1976
Beagle 2	Marte	ESA	2003
Galileo	Júpiter	EEUU	2003

AterrizajeEEUU

nave	planeta	año	despegue
Messenger	Mercurio	2015	⊥
Pioneer	Venus	1978	⊥

```
...
COPY Aterrizaje FROM '/home/csagan/aterrizaje.tsv' DELIMITER E'\t';
```

Específico de Postgres

Concatena los datos

(*Integrity Constraints*)

SQL: RESTRICCIONES

Capítulo 5.7 | Ramakrishnan / Gehrke

Abre una cuenta

Y (por supuesto) hay una base de datos

Ingreso

cuenta	comentario	fecha	hora	monto	saldo	id
7873698669	Deposito iniciál	2020-21-01	20:02:02	300000	300000	TRCXGU8JSHD
7873698669	C0°0°L Designs	2020-02-06	09:15:33	50000	325000	TRCCIA2J8A0

Gasto

cuenta	comentario	fecha	hora	monto	saldo	id
7873698669	Electricidad	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Calefacción	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	Cajero	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD

Cuenta

número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

Divisa

d1	d2	valor
CLP	USD	0,0001533
USD	CLP	652,2750000

Modelo Relacional: Restricciones

Restricciones (*de integridad*):

son restricciones formales

que imponemos a un esquema

que todas sus instancias

deben satisfacer

Restricciones básicas: llaves, nulos, domino

Cuenta

número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94

```
CREATE TABLE Cuenta (
  número BIGINT PRIMARY KEY,
  rut VARCHAR (12) NOT NULL,
  tipo VARCHAR (12) NOT NULL,
  saldo_clp BIGINT NOT NULL,
  saldo_usd DOUBLE PRECISION
     NOT NULL
)
```

```
INSERT INTO Cuenta VALUES
 (7873698669, '28.923.123-7', 'Estacional', 1000, 1.53)
```

```
UPDATE Cuenta SET tipo=NULL WHERE número=7873698669
```

```
INSERT INTO Cuenta VALUES
 (7273697679, '28.923.0123-7', 'Estacional', 1000, 1.53)
```

Restricciones básicas: valores por defecto

Cuenta

número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94
7273697679	28.923.123-7	Estacional	0	0,00

```
CREATE TABLE Cuenta (
  número BIGINT PRIMARY KEY,
  rut VARCHAR (12) NOT NULL,
  tipo VARCHAR (12) NOT NULL,
  saldo_clp BIGINT NOT NULL
    DEFAULT 0,
  saldo_usd DOUBLE PRECISION
    NOT NULL DEFAULT 0
)
```

```
INSERT INTO Cuenta (número, rut, tipo)
VALUES (7273697679, '28.923.123-7', 'Estacional')
```

Restricciones de unicidad

Cuenta

número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94
7273697679	28.923.123-7	Estacional	0	0,00

```
CREATE TABLE Cuenta (
  número INTEGER PRIMARY KEY,
  rut VARCHAR (12) NOT NULL,
  tipo VARCHAR (12) NOT NULL,
  saldo_clp BIGINT NOT NULL
    DEFAULT 0,
  saldo_usd FLOAT NOT NULL
    DEFAULT 0,
  UNIQUE (rut,tipo)
)
```

```
INSERT INTO Cuenta (número, rut, tipo)
VALUES (8079766582, '28.923.123-7', 'Estacional')
```

La llave primaria implica una restricción de unicidad. La unicidad representa una llave candidata: se pueden tener varias llaves candidatas pero una sola llave primaria.

Nombrar (y borrar) restricciones

Cuenta

número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94
7273697679	28.923.123-7	Estacional	0	0,00

```
CREATE TABLE Cuenta (
  número INTEGER,
  rut VARCHAR (12) NOT NULL,
  tipo VARCHAR (12) NOT NULL,
  saldo_clp BIGINT NOT NULL
    DEFAULT 0,
  saldo_usd FLOAT NOT NULL
    DEFAULT 0,
  CONSTRAINT Cuenta_uni_rt UNIQUE (rut,tipo),
  CONSTRAINT Cuenta_pk PRIMARY KEY (número) )
```

```
ALTER TABLE Cuenta
DROP CONSTRAINT Cuenta_uni_rt
```

Más fácil cambiar restricciones posteriormente.

Si hay una violación, el mensaje de error será más intuitivo si las restricciones tienen nombres intuitivos.

Restricciones de llaves foráneas

Ingreso

cuenta	comentario	fecha	hora	monto	saldo	id
7873698669	Deposito iniciál	2020-01-21	20:02:02	300000	300000	TRCXGU8JSHD
7873698669	C0°0°L Designs	2020-02-06	09:15:33	50000	325000	TRCCIA2J8A0

Cuenta

número	rut	tipo	saldo_clp	saldo_usd
7873698669	32.000.273-K	Estacional	225000	344,94

```
CREATE TABLE Ingreso (
 cuenta BIGINT REFERENCES Cuenta(número),
 comentario VARCHAR (255),
 fecha DATE NOT NULL,
 hora TIME NOT NULL,
 monto BIGINT NOT NULL,
 saldo INT NOT NULL,
 id VARCHAR (12) PRIMARY KEY
)
```

```
INSERT INTO Ingreso VALUES
 (7273697679, ... , ...)
```

Cada cuenta en Ingreso tiene que estar en Cuenta.número.

Restricciones de llaves compuestas

Divisa

d1	d2	valor
CLP	USD	0,0001533
USD	CLP	652,2750000
CLP	EUR	0,0001498
EUR	CLP	735,9700000

Cambio

id	D.venta	D.compa	monto
CA0121312393	CLP	USD	100000,00
CA0121312393	CLP	EUR	20000,00
CA2134812341	EUR	CLP	4815,16

```
CREATE TABLE Divisa (
 d1 VARCHAR (3),
 d2 VARCHAR (3),
 valor DOUBLE PRECISION,
 PRIMARY KEY (d1, d2)
)
```

```
CREATE TABLE Cambio (
 id VARCHAR (12),
 venta VARCHAR (3),
 compra VARCHAR (3),
 monto DOUBLE PRECISION,
 FOREIGN KEY (venta, compra) REFERENCES Divisa (d1, d2),
 PRIMARY KEY (id, venta, compra)
)
```

FORMAS NORMALES

Capítulo 12 | Ramakrishnan / Gehrke

Todo bien

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

¿Y si un cliente tiene varios números de teléfono?

...

UNF: Forma No Normalizada (*UnNormalised Form*)

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842,+56223491234	Campo de Hielo Norte, Depto 502

UNF (o 0NF):

Varias multiplicidades de valores en una columna de la tabla

UNF: Forma No Normalizada (*UnNormalised Form*)

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842,+56223491234	Campo de Hielo Norte, Depto 502

1NF:

Un valor en cada celda de la tabla

1NF: Primera Forma Normal (*First Normal Form*)

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

1NF:

Un valor en cada celda de la tabla

1NF: Primera Forma Normal (*First Normal Form*)

Cliente

rut	**nombre**	**fono**	**dirección**
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

¿Algún problema aquí? ...

1NF: Primera Forma Normal (*First Normal Form*)

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

Soluciones con nulos no cuentan aquí.

¿Algún problema aquí? …

Redundancia

Pero ¿por qué es un problema? ¿Solo por el espacio? …

Anomalía de actualización:
Por ejemplo, se puede actualizar la dirección de Rankine en un lugar sin actualizar todos los valores

Anomalía de inserción:
No podemos insertar un nuevo cliente a la tabla si no tenemos un número de teléfono

Anomalía de borrado:
Si el número de teléfono ahora está invalido, tendremos que borrar la fila entera con la dirección, etc.

1NF: Primera Forma Normal (*First Normal Form*)

Cliente

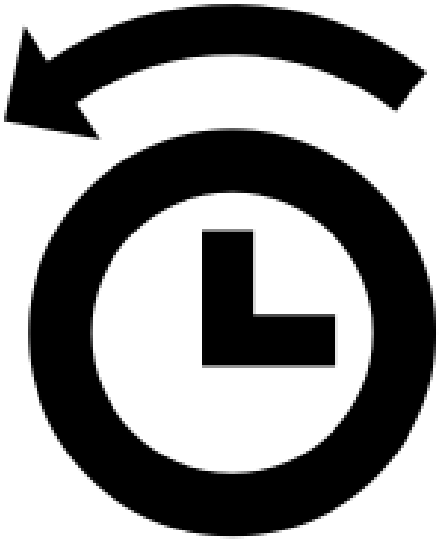
rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

¿La solución? …

Crear otra tabla con rut y fono

Pero ¿cómo podemos definir el problema aquí? …

…



Modelo Relacional: Restricciones (Llaves)

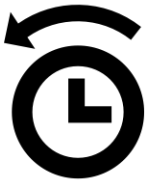
Un conjunto de atributos de una relación

forma una llave **candidata**

si es una súper llave

y *no hay un subconjunto propio de esos atributos*

que es una súper llave



Modelo Relacional: Restricciones (Llaves)

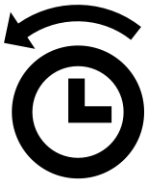
Persona(**rut**,**nombres**,**apellidos**,**fecha-de-nacimiento**,**celular**,**correo-elect**)

¿Intuitivamente, hay otra llave candidata?

Asumiendo que son celulares personales y correos electrónicos personales …

Persona(**rut**,**nombres**,**apellidos**,**fecha-de-nacimiento**,**celular**,**correo-elect**)

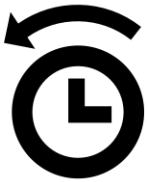
Persona(**rut**,**nombres**,**apellidos**,**fecha-de-nacimiento**,**celular**,**correo-elect**)



Modelo Relacional: Restricciones (Llaves)

Un atributo es primo

si está en alguna llave **candidata**



Modelo Relacional:
Restricciones (Dependencias funcionales)

Dada una relación

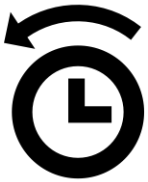
y dos conjuntos de atributos *X*, *Y*

X determina funcionalmente *Y*

si y solo si

cada valor de *X* en la relación

tiene asociado un solo valor de *Y*



Modelo Relacional: Restricciones (Dependencias funcionales)

Cervezas

nombre	tipo	grados	ciudad-origen
Austral Lager	Lager	4,6	Punta Arenas
Austral Yagan	Ale	5,0	Punta Arenas
Austral Pale Ale	Ale	5,0	Punta Arenas
Kuntsmann Torobayo	Ale	5,1	Valdivia
Kross 5	Ale	7,2	Curacaví
Kross Golden	Ale	5,3	Curacaví
Kross Pilsner	Pilsner	4,9	Curacaví

¿Hay una dependencia funcional aquí?

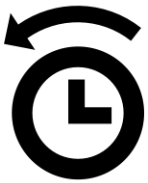
$\{\underline{\textbf{nombre}}\} \rightarrow \{\textbf{tipo}, \textbf{grados}, \textbf{ciudad-origen}\}$

$\{\underline{\textbf{nombre}}\} \rightarrow \{\textbf{tipo}, \underline{\textbf{nombre}}\}$

$\{\textbf{grados}\} \rightarrow \{\textbf{grados}\}$

(al menos aquí) $\{\textbf{grados}\} \rightarrow \{\textbf{tipo}, \textbf{ciudad-origen}\}$

...



1NF: Primera Forma Normal

Asumiendo: $\{\underline{\textbf{rut}}\} \rightarrow \{\textbf{nombre}, \textbf{dirección}\}$

Cliente

<u>rut</u>	nombre	<u>fono</u>	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

¿La solución? ...

Crear otra tabla con rut *y* fono

Pero ¿cómo podemos <u>definir</u> el problema aquí? ...

$\{\underline{\textbf{rut}}\} \rightarrow \{\textbf{nombre}, \textbf{dirección}\}$

... pero **rut** es solo <u>parte</u> de una llave candidata

2NF: Segunda Forma Normal

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

2NF: Satisface la 1NF y …

No existe:

$$A \rightarrow \{b\}$$

tal que:

A *sea un subconjunto propio de una llave candidata*

y b *sea un atributo no primo.*

¿El ejemplo está en la 2NF? …

Depende …

2NF: Segunda Forma Normal

Asumiendo:

$$\{\textbf{nombre},\textbf{fono}\} \rightarrow \{\underline{\textbf{rut}}\}$$
$$\{\textbf{dirección},\textbf{fono}\} \rightarrow \{\underline{\textbf{rut}}\}$$
$$\{\underline{\textbf{rut}}\} \rightarrow \{\textbf{nombre}, \textbf{dirección}\}$$

Cliente

<u>rut</u>	nombre	<u>fono</u>	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

2NF: Satisface la 1NF y ...

No existe:

$$A \rightarrow \{b\}$$

tal que:

A *sea un subconjunto propio de una llave candidata*

y b *sea un atributo no primo.*

¿El ejemplo está en la 2NF? ...

¡Sí! Todos los atributos son primos. Por ejemplo:

$$\{\underline{\textbf{rut}}\} \rightarrow \{\textbf{dirección}\}$$

... pero el atributo **dirección** es primo

2NF: Segunda Forma Normal

Asumiendo: $\{\text{nombre}, \text{fono}\} \rightarrow \{\underline{\text{rut}}\}$

$\{\underline{\text{rut}}\} \rightarrow \{\text{nombre}, \text{dirección}\}$

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

2NF: Satisface la 1NF y ...

No existe:

$$A \rightarrow \{b\}$$

tal que:

A *sea un subconjunto propio de una llave candidata*

y b *sea un atributo no primo.*

¿El ejemplo está en la 2NF? ...

¡No! *Por ejemplo:*

$\{\underline{\text{rut}}\} \rightarrow \{\text{dirección}\}$

2NF: Segunda Forma Normal

Asumiendo: $\{\text{nombre}, \text{fono}\} \rightarrow \{\underline{\text{rut}}\}$

$\{\underline{\text{rut}}\} \rightarrow \{\text{nombre}, \text{dirección}\}$

Cliente

rut	nombre	fono	dirección
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	+56991324842	Campo de Hielo Norte, Depto 502
12.491.671-K	Rankine	+56223491234	Campo de Hielo Norte, Depto 502

2NF: Satisface la 1NF y ...

No existe:

$$A \rightarrow \{b\}$$

tal que:

A *sea un subconjunto propio de una llave candidata*

y b *sea un atributo no primo.*

¿La solución? ...

2NF: Segunda Forma Normal

Cliente

rut	nombre	dirección
32.000.273-K	Kelvin	Campo de Hielo Sur, Depto 273
12.491.671-K	Rankine	Campo de Hielo Norte, Depto 502

Contacto

rut	fono
32.000.273-K	+56976698463
12.491.671-K	+56991324842
12.491.671-K	+56223491234

2NF: Satisface la 1NF y ...

No existe:

$$A \rightarrow \{b\}$$

tal que:

A *sea un subconjunto propio de una llave candidata*

y b *sea un atributo no primo.*

¿El ejemplo está en la 2NF? ...

¡Sí! Contacto *no puede tener atributos no primos y ...*
Si {**nombre**,**dirección**} → {**rut**} *no hay atributos no primos*
Si no, no hay subconjuntos propios de una llave candidata

2NF: Segunda Forma Normal

Asumiendo: $\{\text{cuenta\_destino}\} \rightarrow \{\text{rut\_destino}\}$
$\{\text{rut\_destino}\} \rightarrow \{\text{nombre\_destino}\}$

Transferencia

cuenta_origen	número	cuenta_destino	rut_destino	nombre_destino	monto
7873698669	0000047	1849123812	12.491.671-K	Rankine	41920
7873698669	0000048	1849123812	12.491.671-K	Rankine	10000
8273697679	0000047	7873698669	32.000.273-K	Kelvin	10000
8273697679	0000048	1849123812	12.491.671-K	Rankine	41920

2NF: Satisface la 1NF y ...

No existe:

$$A \rightarrow \{b\}$$

tal que:

A *sea un subconjunto propio de una llave candidata*

y b *sea un atributo no primo.*

¿El ejemplo está en la 2NF? ...

¡Sí!

2NF: Segunda Forma Normal

Asumiendo: $\{\textbf{cuenta\_destino}\} \rightarrow \{\textbf{rut\_destino}\}$
$\{\textbf{rut\_destino}\} \rightarrow \{\textbf{nombre\_destino}\}$

Transferencia

cuenta_origen	número	cuenta_destino	rut_destino	nombre_destino	monto
7873698669	0000047	1849123812	12.491.671-K	Rankine	41920
7873698669	0000048	1849123812	12.491.671-K	Rankine	10000
8273697679	0000047	7873698669	32.000.273-K	Kelvin	10000
8273697679	0000048	1849123812	12.491.671-K	Rankine	41920

¿Se pueden tener anomalías? ...

¡Sí! Por ejemplo ...

Transferencia

...	cuenta_destino	...	nombre_destino	monto
...	1849123812	...	Rankine	41920
...	...	...	...	...
...	1849123812	...	Kelvin	10000

¿Dónde está el problema? ...

$\{\textbf{cuenta\_destino}\} \rightarrow \{\textbf{rut\_destino}, \textbf{nombre\_destino}\}$

(pero cuenta_destino no forma parte de una llave candidata)

3NF: Tercera Forma Normal

Asumiendo: $\{\text{cuenta\_destino}\} \rightarrow \{\text{rut\_destino}\}$
$\{\text{rut\_destino}\} \rightarrow \{\text{nombre\_destino}\}$

Transferencia

cuenta_origen	número	cuenta_destino	rut_destino	nombre_destino	monto
7873698669	0000047	1849123812	12.491.671-K	Rankine	41920
7873698669	0000048	1849123812	12.491.671-K	Rankine	10000
8273697679	0000047	7873698669	32.000.273-K	Kelvin	10000
8273697679	0000048	1849123812	12.491.671-K	Rankine	41920

3NF: Satisface la 2NF y ...

No existe:

$$A \rightarrow B \rightarrow \{c\}$$

tal que :

c *sea no primo*

y $B \not\rightarrow A$ y $c \notin B$

¿El ejemplo está en la 3NF? ...

¡No! *Por ejemplo:*

$\{\underline{\text{cuenta\_origen}}, \underline{\text{número}}\} \rightarrow \{\text{cuenta\_destino}\} \rightarrow \{\text{rut\_destino}\}$

3NF: Tercera Forma Normal

Asumiendo: $\{\text{cuenta\_destino}\} \rightarrow \{\text{rut\_destino}\}$
$\{\text{rut\_destino}\} \rightarrow \{\text{nombre\_destino}\}$

Transferencia

cuenta_origen	número	cuenta_destino	rut_destino	nombre_destino	monto
7873698669	0000047	1849123812	12.491.671-K	Rankine	41920
7873698669	0000048	1849123812	12.491.671-K	Rankine	10000
8273697679	0000047	7873698669	32.000.273-K	Kelvin	10000
8273697679	0000048	1849123812	12.491.671-K	Rankine	41920

3NF: Satisface la 2NF y ...

No existe:

$$A \rightarrow B \rightarrow \{c\}$$

tal que :

c *sea no primo*

y $B \not\rightarrow A$ y $c \notin B$

¿La solución? ...

3NF: Tercera Forma Normal

Transferencia

cuenta_origen	número	cuenta_destino	monto
7873698669	0000047	1849123812	41920
7873698669	0000048	1849123812	10000
8273697679	0000047	7873698669	10000
8273697679	0000048	1849123812	41920

Cuenta

cuenta	rut	nombre
1849123812	12.491.671-K	Rankine
7873698669	32.000.273-K	Kelvin
8273697679	32.000.273-K	Kelvin

3NF: Satisface la 2NF y ...

No existe:

$$A \rightarrow B \rightarrow \{c\}$$

tal que :

c *sea no primo*

y $B \not\rightarrow A$ y $c \notin B$

¿El ejemplo está en la 3NF? ...

Depende ...

3NF: Tercera Forma Normal

Asumiendo

$$\{\textbf{rut}\} \rightarrow \{\textbf{nombre}\}$$

Transferencia

cuenta_origen	número	cuenta_destino	monto
7873698669	0000047	1849123812	41920
7873698669	0000048	1849123812	10000
8273697679	0000047	7873698669	10000
8273697679	0000048	1849123812	41920

Cuenta

cuenta	rut	nombre
1849123812	12.491.671-K	Rankine
7873698669	32.000.273-K	Kelvin
8273697679	32.000.273-K	Kelvin

3NF: Satisface la 2NF y ...

No existe:

$$A \rightarrow B \rightarrow \{c\}$$

tal que :

c *sea no primo*

y $B \not\rightarrow A$ y $c \notin B$

¿El ejemplo está en la 3NF? ...

¡No!

$$\{\underline{\textbf{cuenta}}\} \rightarrow \{\textbf{rut}\} \rightarrow \{\textbf{nombre}\}$$

3NF: Tercera Forma Normal

Asumiendo

$$\{\textbf{rut}\} \rightarrow \{\textbf{nombre}, \underline{\textbf{cuenta}}\}$$

Transferencia

cuenta_origen	número	cuenta_destino	monto
7873698669	0000047	1849123812	41920
7873698669	0000048	1849123812	10000
8273697679	0000047	7873698669	10000
8273697679	0000048	1849123812	41920

Cuenta

cuenta	rut	nombre
1849123812	12.491.671-K	Rankine
7873698669	32.000.273-K	Kelvin

3NF: Satisface la 2NF y ...

No existe:

$$A \rightarrow B \rightarrow \{c\}$$

tal que :

c *sea no primo*

y $B \not\rightarrow A$ y $c \notin B$

¿El ejemplo está en la 3NF? ...

¡Sí!

$\{\underline{\textbf{cuenta}}\} \rightarrow \{\textbf{rut}\} \rightarrow \{\textbf{nombre}\}$ *no cuenta si* $\{\textbf{rut}\} \rightarrow \{\underline{\textbf{cuenta}}\}$

3NF: Tercera Forma Normal

Asumiendo

$\{\underline{\text{cuenta\_destino}}\} \rightarrow \{\text{rut\_destino}\}$

TransferenciaTotal

cuenta_origen	cuenta_destino	rut_destino	total
7873698669	1849123812	12.491.671-K	10000
8273697679	7873698669	32.000.273-K	10000
8273697679	1849123812	12.491.671-K	51920

3NF: Satisface la 2NF y ...

No existe:

$$A \rightarrow B \rightarrow \{c\}$$

tal que :

c sea no primo

y $B \not\rightarrow A$ y $c \notin B$

¿El ejemplo está en la 3NF? ...

¡No! (Ni en la 2NF)

$\{\underline{\text{cuenta\_origen}}, \underline{\text{cuenta\_destino}}\} \rightarrow \{\underline{\text{cuenta\_destino}}\} \rightarrow \{\text{rut\_destino}\}$

3NF: Tercera Forma Normal

Asumiendo

$\{\underline{\text{cuenta\_destino}}\} \rightarrow \{\text{rut\_destino}\}$

$\{\text{rut\_destino}\} \rightarrow \{\underline{\text{cuenta\_destino}}\}$

TransferenciaTotal

cuenta_origen	cuenta_destino	rut_destino	total
7873698669	1849123812	12.491.671-K	10000
8273697679	7873698669	32.000.273-K	10000
8273697679	1849123812	12.491.671-K	51920

3NF: Satisface la 2NF y ...

No existe:

$$A \rightarrow B \rightarrow \{c\}$$

tal que :

c *sea no primo*

y $B \not\rightarrow A$ y $c \notin B$

¿El ejemplo está en la 3NF? ...

¡Sí! El único atributo no primo es total *y no está en una dependencia problemática según la 2NF ni la 3NF.*

3NF: Tercera Forma Normal

Asumiendo

$\{\underline{\text{cuenta\_destino}}\} \rightarrow \{\text{rut\_destino}\}$

$\{\text{rut\_destino}\} \rightarrow \{\underline{\text{cuenta\_destino}}\}$

TransferenciaTotal

cuenta_origen	cuenta_destino	rut_destino	total
7873698669	1849123812	12.491.671-K	10000
8273697679	7873698669	32.000.273-K	10000
8273697679	1849123812	12.491.671-K	51920

¿Hay una posibilidad de anomalías? ...

¡Sí! Por ejemplo ...

TransferenciaTotal

cuenta_origen	cuenta_destino	rut_destino	total
7873698669	1849123812	12.491.671-K	51920
...	...	...	...
8273697679	1849123812	24.482.054-9	51920

BCNF: Boyce–Codd

Asumiendo

$\{\underline{\text{cuenta\_destino}}\} \rightarrow \{\text{rut\_destino}\}$

$\{\text{rut\_destino}\} \rightarrow \{\underline{\text{cuenta\_destino}}\}$

TransferenciaTotal

cuenta_origen	cuenta_destino	rut_destino	total
7873698669	1849123812	12.491.671-K	10000
8273697679	7873698669	32.000.273-K	10000
8273697679	1849123812	12.491.671-K	51920

BCNF: Satisface la 1NF y ...

Para cada:

$X \rightarrow Y$

X es una súper llave

o $Y \subseteq X$

¿El ejemplo está en la BCNF? ...

¡No!

$\{\text{rut\_destino}\} \rightarrow \{\underline{\text{cuenta\_destino}}\}$

BCNF: Boyce–Codd

Asumiendo

$\{\underline{\text{cuenta\_destino}}\} \rightarrow \{\text{rut\_destino}\}$

$\{\text{rut\_destino}\} \rightarrow \{\underline{\text{cuenta\_destino}}\}$

TransferenciaTotal

cuenta_origen	cuenta_destino	rut_destino	total
7873698669	1849123812	12.491.671-K	10000
8273697679	7873698669	32.000.273-K	10000
8273697679	1849123812	12.491.671-K	51920

BCNF: Satisface la 1NF y ...

Para cada:

$X \rightarrow Y$

X es una súper llave

o $Y \subseteq X$

¿La solución? ...

BCNF: Boyce–Codd

TransferenciaTotal

cuenta_origen	cuenta_destino	total
7873698669	1849123812	10000
8273697679	7873698669	10000
8273697679	1849123812	51920

Cuenta

cuenta	rut
1849123812	12.491.671-K
7873698669	32.000.273-K

BCNF: Satisface la 1NF y ...

Para cada:

$X \rightarrow Y$

X *es una súper llave*

o $Y \subseteq X$

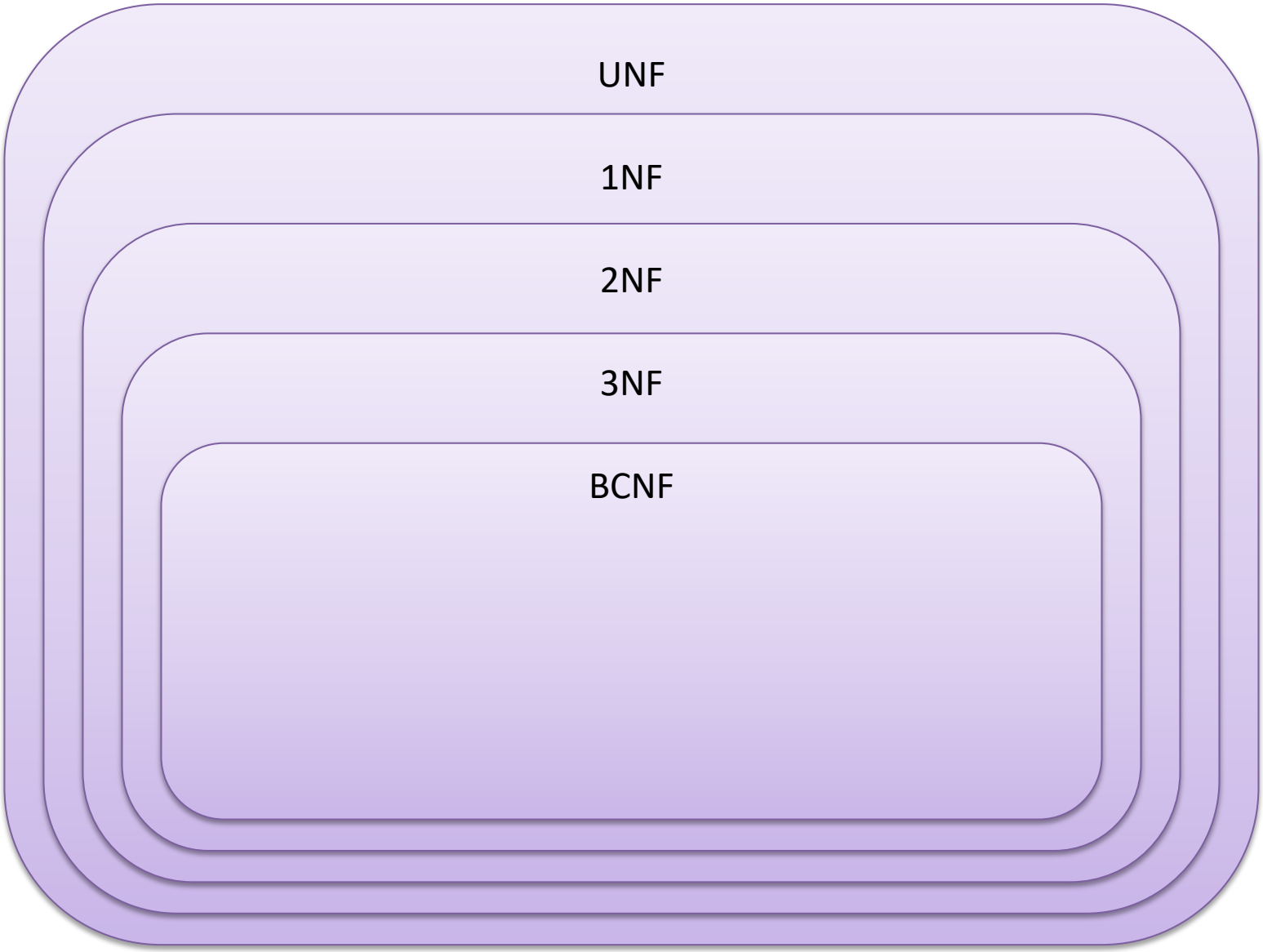
¿El ejemplo está en la BCNF? ...

¡Sí! Por ejemplo ...

Si {**rut**} → {**cuenta**} *entonces* **rut** *es una súper llave*

Si no, entonces no hay problema.

UNF ⊇ 1NF ⊇ 2NF ⊇ 3NF ⊇ BCNF



UNF ⊇ 1NF ⊇ 2NF ⊇ 3NF ⊇ BCNF

1NF ⊇ 2NF

2NF: ***Satisface la 1NF*** y ...

No existe:

$$A \rightarrow \{b\}$$

tal que:

A *sea un subconjunto propio de una llave candidata*

y b *sea un atributo no primo.*

2NF $\supseteq$ 3NF

3NF: *Satisface la 2NF* y ...

No existe:

$$A \rightarrow B \rightarrow \{c\}$$

tal que :

c *sea no primo*

y $B \not\rightarrow A$ y $c \notin B$

2NF $\supseteq$ BCNF

2NF: Satisface la 1NF y ...

No existe:

$$A \rightarrow \{b\}$$

tal que:

A *sea un subconjunto propio de una llave candidata y* b *sea no primo.*

$\supseteq$

BCNF: Satisface la 1NF y ...

Para cada:

$$X \rightarrow Y$$

X *es una súper llave*

o $Y \subseteq X$

Conyectura: Si una tabla satisface la BNCF, satisface la 2NF.

Argumento: Cualquier contraejemplo $A \rightarrow \{b\}$ para la 2NF es un contraejemplo para la BCNF: dado que A es un subconjunto propio de una llave candidata no puede ser una súper llave, y dado que b no es primo, $\{b\}$ no puede estar contenido en A (porque A es un subconjunto de una llave candidata).

Así que si una tabla no satisface la 2NF, no puede satisfacer la BCNF.
Equivalentemente (por contraposición), si satisface la BCNF, satisface la 2NF.

3NF $\supseteq$ BCNF

3NF: Satisface la 2NF y ...

No existe:

$$A \to B \to \{c\}$$

tal que :

c *sea no primo*

y $B \not\to A$ y $c \notin B$

$\supseteq$

BCNF: Satisface la 1NF y ...

Para cada:

$$X \to Y$$

X *es una súper llave*

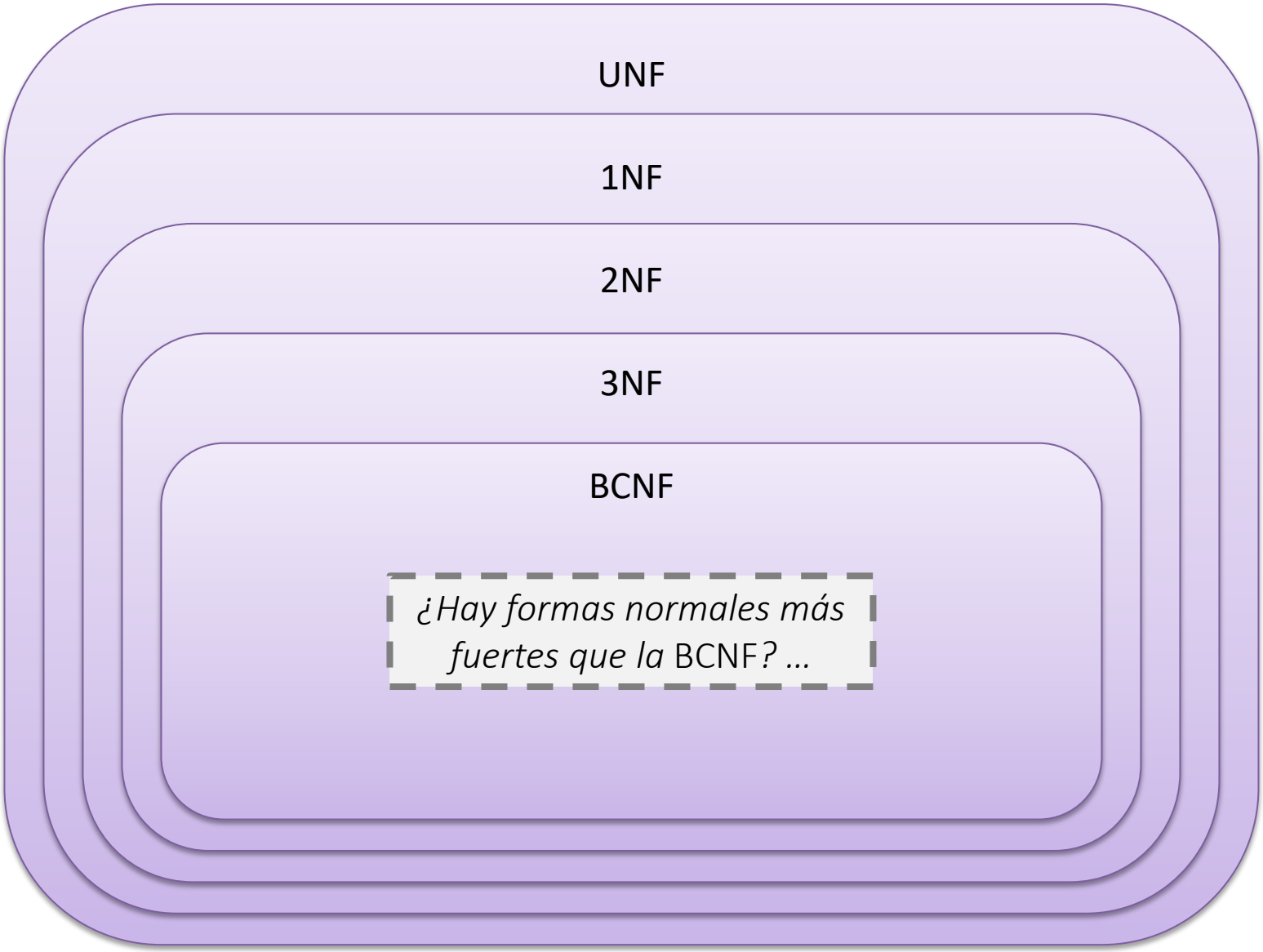
o $Y \subseteq X$

Conyectura: Si una tabla está en la BNCF, está en la 3NF.

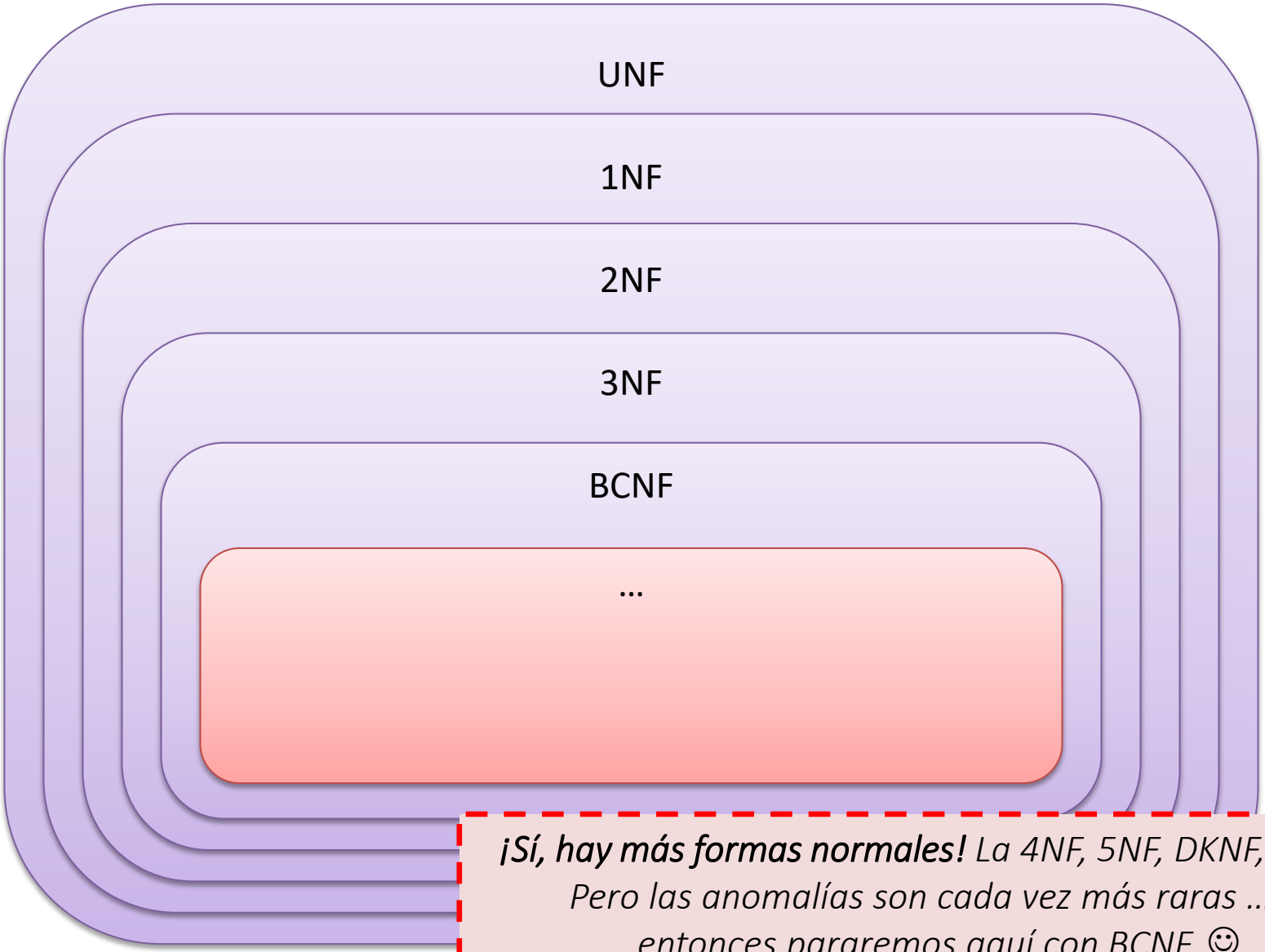
Argumento: Si la tabla no está en la 2NF, no está en la BCNF (demonstrado anteriormente). Cualquier contraejemplo $A \to B \to \{c\}$ para la 3NF da el contraejemplo $B \to \{c\}$ para la BCNF: $B \not\to A$ implica que B no puede ser una súper llave, y $c \notin B$ implica que $\{c\} \not\subseteq B$.

Así que si una tabla no satisface la 3NF, no puede satisfacer la BCNF.
Equivalentemente (por contraposición), si satisface la BCNF, satisface la 3NF.

UNF ⊇ 1NF ⊇ 2NF ⊇ 3NF ⊇ BCNF



UNF ⊇ 1NF ⊇ 2NF ⊇ 3NF ⊇ BCNF ⊇ …



¡Sí, hay más formas normales! *La 4NF, 5NF, DKNF, 6NF,*
Pero las anomalías son cada vez más raras …
… entonces pararemos aquí con BCNF. ☺

Formas Normales

¿Qué piensan ustedes?

¿Son útiles las formas normales?

BCNF: Satisface la 1NF y …

Para cada:

$X \rightarrow Y$

X es una súper llave

o $Y \subseteq X$

¿Los nulos pueden ser una solución?

Satélite

nombre	planeta	descubridor	año
Luna	Tierra	⊥	⊥
Ganímedes	Júpiter	Galileo Galilei	1610
Calisto	Júpiter	Galileo Galilei	1610
Europa	Júpiter	Galileo Galilei	1610
Ío	Júpiter	Galileo Galilei	1610
Titán	Saturno	Christiaan Huygens	1655
Tritón	Neptuno	William Lassell	1846

¿Qué piensan ustedes?

¿Más fácil permitir algunos nulos?

Preguntas?