Bases de Datos

Clase 09: Foreign Keys - Assertions

Construir una aplicación con un DBMS

1. Modelar los requerimientos (trabajo conceptual, con diagramas)

2. Diseño del esquema e implementación (tablas, atributos, llenar la base de datos)

3. Programar aplicación usando el DMBS (mucho más fácil si el trabajo anterior fue bien hecho)

Construir una aplicación con un DBMS

1. Modelar los requerimientos (trabajo conceptual, con diagramas)
2. Diseño del esquema e implementación (tablas, atributos, llenar la base de datos)
3. **Programar aplicación usando el DMBS (mucho más fácil si el trabajo anterior fue bien hecho)**

BCNF al crear la base de datos

SQL está diseñado para soportar BCNF. Las dependencias son difíciles de expresar, pero no las llaves

Restricciones de Inclusión

Son de la forma:

$$R[A_1, \ldots, A_n] \subseteq S[B_1, \ldots, B_n]$$

Para ***R*** y ***S*** relaciones y $A_1,\ldots,A_n$ y $B_1,\ldots,B_n$ atributos de ***R*** y ***S*** respectivamente

En álgebra:

$$\pi_{A_1,\ldots,A_n}(R) \subset \pi_{B_1,\ldots,B_n}(S)$$

Por ejemplo: Fabrica[idCompañía] ⊆ Compañía[id]

Llaves Foráneas

Cuando la referencia a la tabla es una llave:

$$R[A_1, \ldots, A_n] \subseteq S[B_1, \ldots, B_n]$$

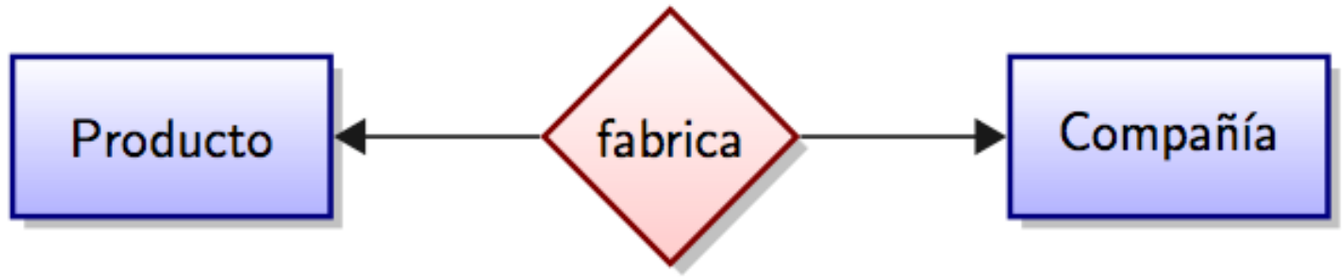
Y $B_1,\ldots,B_n$ son llave para ***S***

La relación R contiene la llave de la relación S

Llaves Foráneas

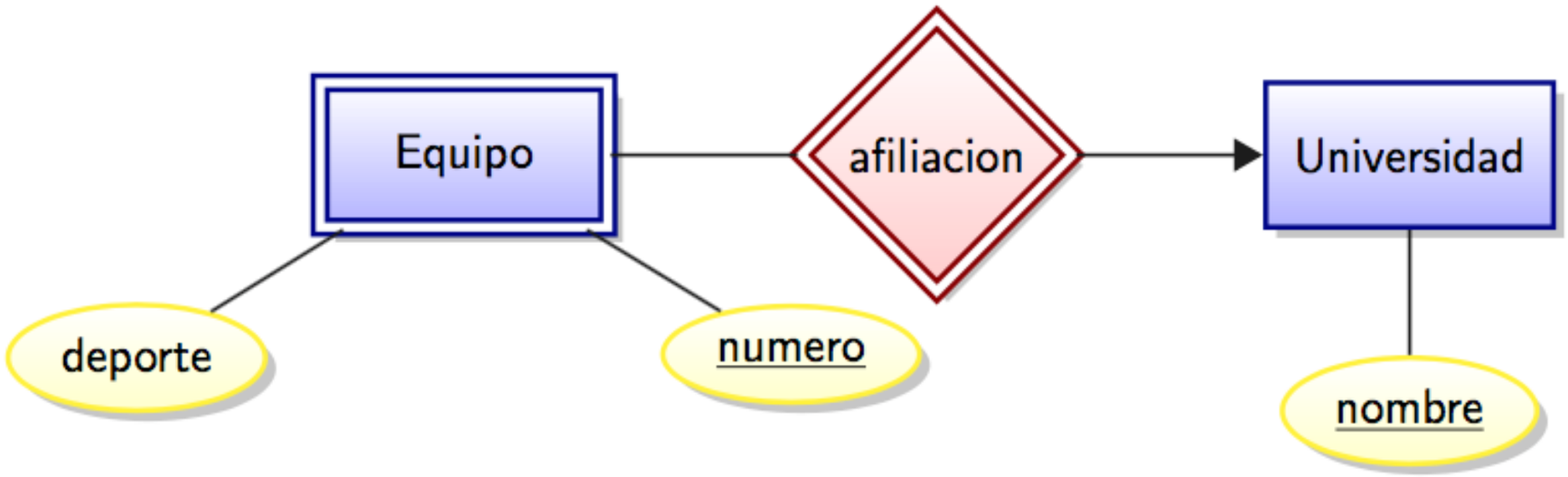
Por ejemplo: Fabrica[idCompañía] ⊆ Compañía[id], asumiendo que id es llave de Compañía

Llaves Foráneas



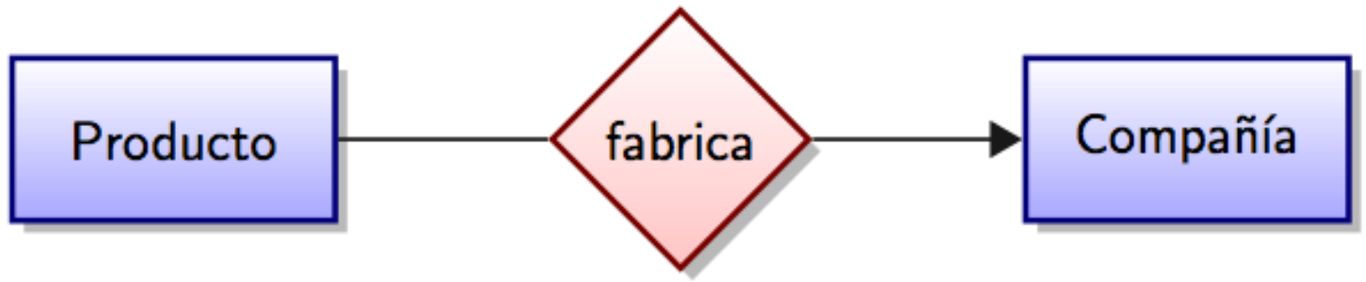
¿Hay que aplicar llaves foráneas?

Llaves Foráneas



¿Hay que aplicar llaves foráneas?

Llaves Foráneas



¿Hay que aplicar llaves foráneas?

Llaves Foráneas

Las llaves foráneas (junto a las llaves) son las restricciones de integridad que se usan día a día en los DBMS relacionales

SQL

Llaves

```
CREATE TABLE Películas(
  título varchar(30),
  año int,
  género char(10),
  PRIMARY KEY (título, año))
```

SQL

Llaves

```
CREATE TABLE Películas(
  título varchar(30),
  año int,
  género char(10),
  PRIMARY KEY (título, año))
```

```
INSERT INTO Películas VALUES (‘El Gran Gatsby’, 2013, ‘Drama’)
```

SQL

Llaves

```
CREATE TABLE Películas(
  título varchar(30),
  año int,
  género char(10),
  PRIMARY KEY (título, año))
```

```
INSERT INTO Películas VALUES (‘El Gran Gatsby’, 2013, ‘Drama’)
```

```
INSERT INTO Películas VALUES (‘El Gran Gatsby’, 2013, ‘Drama’)
```

SQL

Llaves

```
CREATE TABLE Películas(
  título varchar(30),
  año int,
  género char(10),
  PRIMARY KEY (título, año))
```

```
INSERT INTO Películas VALUES (‘El Gran Gatsby’, 2013, ‘Drama’)
```

```
INSERT INTO Películas VALUES an Gatsby’, 2013, ‘Drama’)
```

SQL

Llaves

```
ERROR: duplicate key value violates
unique constraint "t_peliculas_pkey"
```

Llaves Foráneas en SQL

```
CREATE TABLE Películas(
  título varchar(30), año int, género char(10)
)

CREATE TABLE Programa(
  título varchar(30), cine varchar(20),
  FOREIGN KEY(título) REFERENCES Películas(título)
)
```

Llaves Foráneas en SQL

```
CREATE TABLE Películas(
  título varchar(30), año int, género char(10)
)

CREATE TABLE Programa(
  título varchar(30), cine varchar(20),
  FOREIGN KEY(título) REFERENCES Películas(título)
)
```

Semántica: Programa[título] ⊆ Películas[título]

Llaves Foráneas en SQL

```
CREATE TABLE Películas(
  título varchar(30), año int, género char(10)
)

CREATE TABLE Programa(
  título varchar(30), cine varchar(20),
  FOREIGN KEY(título) REFERENCES Películas(título)
)
```

Semántica: Programa[título] ⊆ Películas[título]

Está en el estándar, pero no siempre se implementa!

Llaves Foráneas en SQL

Cuando atributos se llaman igual

```
CREATE TABLE Películas(
  título varchar(30), año int, género char(10)
)

CREATE TABLE Programa(
  título varchar(30), cine varchar(20),
  FOREIGN KEY(título) REFERENCES Películas
)
```

Llaves Foráneas en SQL

Inserciones con llaves foráneas

¿Qué pasa en este caso?

```
CREATE TABLE R(a int, b int, PRIMARY KEY(a))

CREATE TABLE S(a int, FOREIGN KEY(a) REFERENCES R)

INSERT INTO R VALUES(1, 1)

INSERT INTO S VALUES(1)
```

Llaves Foráneas en SQL

Inserciones con llaves foráneas

¿Qué pasa en este caso?

```
CREATE TABLE R(a int, b int, PRIMARY KEY(a))

CREATE TABLE S(a int, FOREIGN KEY(a) REFERENCES R)

INSERT INTO R VALUES(1, 1)

INSERT INTO S VALUES(1)
```

Todo bien hasta ahora…

Llaves Foráneas en SQL

Inserciones con llaves foráneas

```
CREATE TABLE R(a int, b int, PRIMARY KEY(a))

CREATE TABLE S(a int, FOREIGN KEY(a) REFERENCES R)
```

```
INSERT INTO R VALUES(1, 1)

INSERT INTO S VALUES(1)

INSERT INTO S VALUES(2)
```

Llaves Foráneas en SQL

Inserciones con llaves foráneas

```
CREATE TABLE R(a int, b int, PRIMARY KEY(a))

CREATE TABLE S(a int, FOREIGN KEY(a) REFERENCES R)
```

```
INSERT INTO R VALUES(1, 1)

INSERT INTO S VALUES(1)

INSERT INTO S VALUES(2)
```

ERROR

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Tenemos **S**[a]⊆**R**[a]

S	A	C
	1	3
	2	2

R	A	B
	1	2
	2	3

Qué ocurre al eliminar (1, 2) en **R**?

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Qué ocurre al eliminar (1, 2) en **R**?

Tenemos las siguientes opciones:

- No permitir eliminación
- Propagar la eliminación y también borrar (1,3) de S
- Enfoque “no sabemos”: mantener la tupla pero dejar el valor en nulo

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 1: no permitir la eliminación es el default en SQL

```
CREATE TABLE R(a int, b int, PRIMARY KEY(a))

CREATE TABLE S(a int, FOREIGN KEY(a) REFERENCES R)
```

Qué ocurre al eliminar (1, 2) en **R**?

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 1: no permitir la eliminación es el default en SQL

S	A	C
	1	3
	2	2

R	A	B
	1	2
	2	3

Qué ocurre al eliminar (1, 2) en **R**?

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 1: no permitir la eliminación es el default en SQL

S	A	C
	1	3
	2	2

R	A	B
	1	2
	2	3

Qué ocurre al eliminar (1, 2) en **R**?

Respuesta: obtenemos error

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 2: Propagar la eliminación

```
CREATE TABLE R(a int, b int, PRIMARY KEY(a))

CREATE TABLE S(a int, FOREIGN KEY(a) REFERENCES R ON
                DELETE CASCADE)
```

Qué ocurre al eliminar (1, 2) en **R**?

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 2: Propagar la eliminación

S	A	C
	1	3
	2	2

R	A	B
	1	2
	2	3

Qué ocurre al eliminar (1, 2) en **R**?

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 2: Propagar la eliminación

S	A	C
	1	3
	2	2

R	A	B
	1	2
	2	3

Qué ocurre al eliminar (1, 2) en **R**?

Respuesta: se elimina también (1, 3) en S

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 3: dejar en nulo

```
CREATE TABLE R(a int, b int, PRIMARY KEY(a))
CREATE TABLE S(a int, FOREIGN KEY(a) REFERENCES R ON
                DELETE SET NULL)
```

Qué ocurre al eliminar (1, 2) en **R**?

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 3: dejar en nulo

S	A	C
	1	3
	2	2

R	A	B
	1	2
	2	3

Qué ocurre al eliminar (1, 2) en **R**?

Llaves Foráneas en SQL

Eliminar con llaves foráneas

Opción 3: dejar en nulo

S	A	C
	1	3
	2	2

R	A	B
	1	2
	2	3

Qué ocurre al eliminar (1, 2) en **R**?

Respuesta: la tupla (1, 3) en S ahora es (`null`, 3)

Restricciones locales

Podemos declarar otras restricciones al momento de crear una tabla

```
CREATE TABLE R(
 …
 <atributo 1> <tipo 1> CHECK <condición 1>,
 <atributo 2> <tipo 2> CHECK <condición 2>,
 …
)
```

Restricciones locales

Un atributo nota que está entre 1 y 7

```
nota int CHECK (1 <= nota AND nota <= 7)
```

Restricciones locales

Un atributo **A** es menor que 10 y ocurre como valor en la columna **C** de otra tabla

```
A int CHECK (A IN
              (SELECT R.C FROM R WHERE R.C < 10))
```

Restricciones locales

El atributo **A** ocurre precisamente una vez como valor del atributo **B** en la relación **R**

```
A int CHECK (1 = (SELECT COUNT(*) FROM S WHERE A = B))
```