Bases de Datos I

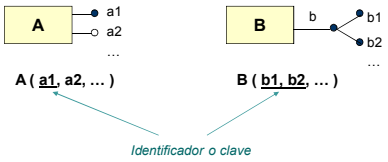
Cursada 2008

Clase 3: Creación y actualización de Tablas. Restricciones

Facultad de Ciencias Exactas
Universidad Nac. Centro de la Pcia. de Bs. As.

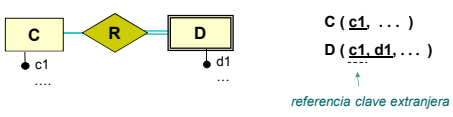
Derivación de Tablas: Entidades

- Por cada entidad A se crea una tabla que contenga todos los atributos simples de A y los atributos simples que componen los atributos compuestos de A
- El identificador de A se transforma en la clave primaria de la tabla (si el atributo clave elegido es compuesto, el conjunto de atributos simples que lo conforman serán la clave de A).



Derivación de Tablas: Entidades

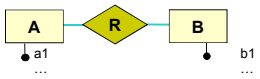
- Por cada entidad débil D se crea una tabla que contenga todos los atributos simples de D (y los atributos simples que compongan los atributos compuestos de D). Además se incluyen los atributos identificatorios de la entidad propietaria C (como clave extranjera)
- La clave primaria de D es la combinación de los atributos identificatorios de D y C



C (c1, . . .)
D (c1, d1, . . .)
...
referencia clave extranjera

Derivación de Tablas: Relaciones

- En las relaciones unarias o binarias con cardinalidad 1:N se adicionan los atributos identificatorios de la entidad del lado '1' a la entidad del lado 'N'
- Las relaciones binarias N:N y n-arias se transforman en una nueva tabla con los atributos claves de las entidades participantes en la relación y los atributos propios de la relación (si los tiene)

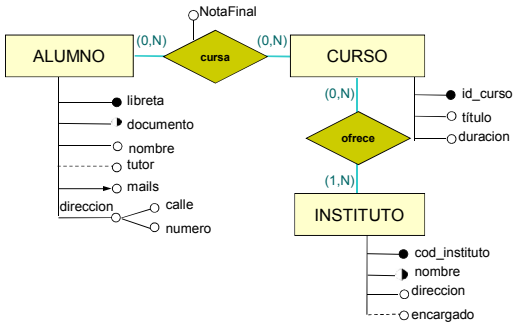


N:1 → A (a1, . . . , b1)
B (b1, . . .)

1:1 → A (a1, . . . , b1)
B (b1, . . . , a1)

N:N → A (a1, . . .)
B (b1, . . .) R (a1, b1)

Derivación de Tablas: Ejemplo



Derivación de Tablas: Ejemplo

ALUMNO (libreta, documento, nombre, tutor, dir_calle, dir_nro)

ALUMNO_MAIL (libreta, mail)

o:

ALUMNO (libreta, documento, nombre, tutor, dir_calle, dir_nro, mail1, mail2)

CURSO (id_curso, título, duracion)

INSTITUTO (cod_instituto, nombre, direccion, encargado)

CURSA (libreta, id_curso)

OFRECE (cod_instituto, id_curso)

Creación de Tablas con SQL

NOTA:
[…]: opcional
{…}: uno o más

```
CREATE TABLE NombreTabla
( { nombre_columna  TipoDato  [NOT NULL]  [UNIQUE]
    [DEFAULT valorDefecto]
    [CHECK (condicion)] }
  [ [CONSTRAINT nom_restr] PRIMARY KEY (lista_ColumnasPK),]
  { [ [CONSTRAINT nom_restr] UNIQUE (listaColumnas),] }
  { [ [CONSTRAINT nom_restr] FOREIGN KEY (lista_ColumnasFK)
        REFERENCES nombreTablaPadre [(lista_ColumnasRef)],
        [ON UPDATE AccionReferencial]
        [ON DELETE AccionReferencial] ] }
  { [CONSTRAINT nom_restr] [CHECK (condicion)] [,…] }
);
```

AccionReferencial ::= CASCADE | RESTRICT | …

SQL – tipos de datos

- ✓ Numéricos:
  - INTEGER - valores enteros
  - NUMERIC(*p*,*e*) – valores flotantes (*p:* nro. total de dígitos, *e:* decimales)
  - ……
- ✓ Caracteres:
  - CHAR(n) - cadenas de caracteres alfanuméricos de longitud fija *n*
  - VARCHAR(n) - cadenas de caracteres alfanuméricos de longitud variable (*n: longitud máxima*)
- ✓ Boolean

  valores TRUE, FALSE y UNKNOWN
- ✓ Datetime:
  - TIME - valores de HOUR, MINUTE and SECOND
  - DATE - valores de YEAR, MONTH, DAY, HOUR, MINUTE, SECOND
  - …

SQL – restricciones de nulidad y unicidad

✓ La restricción NOT NULL sobre un atributo indica que el sistema debe rechazar cualquier intento de insertar un nulo en esa columna.

✓ Las claves primarias deben especificarse como NOT NULL

✓ Se puede especificar un valor por defecto en una columna mediante la cláusula DEFAULT

✓ Las claves alternativas de una tabla se especifican mediante la cláusula UNIQUE (si se trata de un atributo simple se puede indicar en la definición del atributo o como restricción de tabla, si está compuesta de más de un atributo se debe indicar como restricción de tabla).

Ejemplo:

```
CREATE TABLE Alumno (
    ….
   documento  CHAR(8)  NOT NULL  UNIQUE,
   nombre  VARCHAR(50)  NOT NULL  DEFAULT ´no especificado´,
    …
```

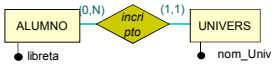
SQL – restricciones de integridad referencial

✓ Regla de Integridad Referencial (RIR): *"Si en una tabla hay una clave extranjera, sus valores deben coincidir con valores de la clave primaria a la que hace referencia, o bien deben ser nulos"*

✓ Cada una de las claves extranjeras se especifica mediante la cláusula
FOREIGN KEY (lista_ColumnasFK)
REFERENCES nombreTablaPadre [(lista_ColumnasRef)]

puede omitirse si es la PK de TablaPadre (se requiere igualdad de dominios)

Ejemplo:




```
CREATE TABLE Universidad
( nom_Univ  CHAR(30) NOT NULL,
  ……
  PRIMARY KEY (nom_Univ) );
```

```
CREATE TABLE Alumno
( libreta       INTEGER  NOT NULL,
  …
  nom_Univ  CHAR(30),
  PRIMARY KEY (libreta),
  FOREIGN KEY (nom_Univ)
        REFERENCES Universidad) ;
```


SQL – restricciones de integridad referencial

✓ Si una operación conduce a un estado ilegal →2 estrategias:
- ***Rechazar*** la operación (RESTRICT)
- ***Reparar*** realizando operaciones compensatorias adicionales que conduzcan a un estado legal (CASCADE), aceptando la operación

✓ Entonces:

```
FOREIGN KEY (lista_ColumnasFK)
      REFERENCES nombreTablaPadre [(lista_ColumnasRef)]
      [ ON UPDATE AccionReferencial ]
      [ ON DELETE AccionReferencial ]
```

AccionReferencial ::= CASCADE | RESTRICT | …

SQL – RIRs (Ejemplo borrado 1)

```
CREATE TABLE Alumno
( libreta       INTEGER  NOT NULL,
  …
  nom_Univ  CHAR(30),
  PRIMARY KEY (libreta),
  FOREIGN KEY (nom_Univ)
        REFERENCES Universidad)
        ON DELETE  restrict
  …
);
```

UNIVERSIDAD

nom_Univ	direccion
UNCPBA	Pinto 399-Tandil
UNLP	50 y 151- La Plata
~~UNS……~~	~~Av.Colón 80 B.B.~~

ALUMNO

libreta ….	Nom_Univ
123	UNCPBA
453	UNLP
680	UNCPBA

Qué sucede si se pretende eliminar la UNCPBA?

→ *Se rechaza la operación porque la modalidad de borrado es RESTRICT y hay 2 referencias a UNCPBA en la tabla Alumno*

Qué sucede si se pretende eliminar la UNS?

→ *La operación es aceptada porque no existen referencias a UNS en Alumno (aunque la modalidad sea RESTRICTA)*

SQL – RIRs (Ejemplo borrado 2)

```
CREATE TABLE Alumno
( libreta       INTEGER  NOT NULL,
  …
  nom_Univ  CHAR(30),
  PRIMARY KEY (libreta),
  FOREIGN KEY (nom_Univ)
        REFERENCES Universidad)
        ON DELETE  cascade
  …
);
```

UNIVERSIDAD

nom_Univ	direccion
~~UNCPBA~~	~~Pinto 399-Tandil~~
UNLP	50 y 151- La Plata
UNS…..	Av.Colón 80- B.B.

ALUMNO

libreta ….	Nom_Univ
~~123~~	~~UNCPBA~~
453	UNLP
~~680~~	~~UNCPBA~~

Qué sucede si se pretende eliminar la UNCPBA?

→La operación se acepta y el DBMS aplica acciones reparadoras para mantener la consistencia de la Base:

- *se elimina la UNCPBA de la Tabla Universidad*
- *se eliminan todas las filas de la tabla Alumno que referencian a UNCPBA*

SQL – RIRs (Ejemplo actualización 1)

```
CREATE TABLE Alumno
( libreta       INTEGER  NOT NULL,
  …
  nom_Univ  CHAR(30),
  PRIMARY KEY (libreta),
  FOREIGN KEY (nom_Univ)
        REFERENCES Universidad)
        ON UPDATE  restrict
  …
);
```

UNIVERSIDAD

nom_Univ	direccion
UNCPBA	Pinto 399-Tandil
UNLP	50 y 151- La Plata
UNSur	Av.Colón 80- B.B.

ALUMNO

libreta ….	Nom_Univ
123	UNCPBA
453	UNLP
680	UNCPBA

Qué sucede si se pretende modificar la PK de Universidad: UNCPBA a UNICEN?

→ Se rechaza la operación porque la modalidad de actualización es RESTRICT y hay 2 referencias a UNCPBA en la tabla Alumno

Qué sucede si se pretende modificar la PK de Universidad: UNS a UNSur?

→ La operación es aceptada porque no existen referencias a UNS en Alumno (aunque la modalidad sea RESTRICTA) y se hace la modificación

SQL – RIRs (Ejemplo actualización 2)

```
CREATE TABLE Alumno
( libreta       INTEGER  NOT NULL,
  …
  nom_Univ  CHAR(30),
  PRIMARY KEY (libreta),
  FOREIGN KEY (nom_Univ)
        REFERENCES Universidad)
        ON DELETE  cascade
  …
);
```

UNIVERSIDAD

nom_Univ	direccion
UNICEN	Pinto 399-Tandil
UNLP	50 y 151- La Plata
UNS…..	Av.Colón 80- B.B.

ALUMNO

libreta ….	Nom_Univ
123	**UNICEN**
453	UNLP
680	**UNICEN**

Qué sucede si se quiere modificar la PK de Universidad: UNCPBA a UNICEN?

→La operación se acepta y el DBMS aplica acciones reparadoras para mantener la consistencia de la Base:

- *se modifica el campo nom_Univ de la Tabla Universidad a UNICEN*
- *se modifican todas las referencias en la tabla Alumno como UNICEN*

Creación de Tablas (ejemplo)

Entidades:

```
CREATE TABLE Alumno
( libreta   INTEGER  NOT NULL,
  documento  CHAR(8)  NOT NULL  UNIQUE,
  nombre  VARCHAR(50)  NOT NULL  DEFAULT ´no especificado´,
  tutor  VARCHAR(30),
  mail1  VARCHAR(50)  NOT NULL,
  mail2  VARCHAR(50),
  direc_calle  VARCHAR(30) NOT NULL,
  direc_numero  CHAR(5) NOT NULL,
  CONSTRAINT  Pk_Alumno  PRIMARY KEY (libreta) );
```

```
CREATE TABLE Curso
( id_curso  INTEGER  NOT NULL,
  titulo  VARCHAR(100)  NOT NULL,
  duracion INTEGER  NOT NULL
                    DEFAULT  60,
  PRIMARY KEY (id_curso) );
```

```
CREATE TABLE Instituto
( cod_instituto  CHAR(10)  NOT NULL,
  nombre  VARCHAR(50)  NOT NULL,
  direccion VARCHAR(40) NOT NULL,
  encargado  VARCHAR(50),
  PRIMARY KEY (cod_instituto)
  UNIQUE  (nombre) );
```


Creación de Tablas (ejemplo)

Relaciones:

```
CREATE TABLE Cursa
( libreta   INTEGER  NOT NULL,
  id_curso INTEGER NOT NULL,
  nota_final  NUMERIC(4,2) NOT NULL,
  CONSTRAINT  Pk_Cursa  PRIMARY KEY (libreta, id_curso),
  CONSTRAINT  Fk_Cursa_Alu FOREIGN KEY libreta REFERENCES Alumno
     ON UPDATE restrict ON DELETE restrict,
  CONSTRAINT  Fk_Cursa_Cur FOREIGN KEY id_curso REFERENCES Curso
     ON UPDATE cascade ON DELETE restrict );
```

```
CREATE TABLE Ofrece
( cod_instituto CHAR(10) NOT NULL,
  id_curso INTEGER NOT NULL,
  CONSTRAINT  Pk_Ofrece  PRIMARY KEY (cod_instituto, id_curso),
  CONSTRAINT  Fk_Ofrece_Inst FOREIGN KEY cod_instituto REFERENCES Instituto
     ON UPDATE cascade ON DELETE cascade,
  CONSTRAINT  Fk_Ofrece_Cur FOREIGN KEY id_curso REFERENCES Curso
     ON UPDATE cascade ON DELETE restrict);
```


Incorporación de restricciones simples

- ✓ Algunas restricciones se pueden asociar a atributos de la tabla, indicando los valores que pueden asumir

  Ejemplo: el sueldo de un Empleado no puede ser un valor negativo, la nota de un Alumno puede tomar valores entre 0 y 10, las carreras de la Facultad son alguna de las siguientes: ´Ing.Sistemas´, ´Prof.Matemática´, ´Prof.Física´

- ✓ Se especifican mediante la cláusula
  [CONSTRAINT nombre_restr] CHECK (condición)
- ✓ La condición es un predicado lógico que debe evaluarse a Verdadero (True) o Falso (False)
- ✓ Se puede usar AND y OR para componer condiciones

Incorporación de restricciones simples

- Comparación simple
  Ej: Sueldo>0
- Rango → BETWEEN, NOT BETWEEN (incluye los extremos)
  Ej: cantMaterias BETWEEN 0 AND 45
  edad NOT BETWEEN 14 AND 20
- Membresía o pertenencia → IN, NOT IN
  Ej: NomCarrera IN ('Sistemas', Fisica', 'Matematica')
  NomCarrera IN (<*columna seleccionada de otra tabla*>)
- Test de Nulidad → IS NULL, IS NOT NULL

Ej: FechaCreacion IS NOT NULL OR FechaModif IS NOT NULL

Nota: expresión IS NOT NULL Ξ NOT (exp IS NULL)

Incorporación de restricciones simples

- Semejanza de Patrones → LIKE, NOT LIKE
  % secuencia de 0 ó más caracteres
  _ cualquier carácter simple

Ej:
(NOT) LIKE 's%' → cualquier cadena que empiece con s
NOT LIKE '%s' → cualquier cadena que no termine con s
LIKE 's__' → cualquier cadena de 3 caracteres comenzando con s
(NOT) LIKE '%RA% → cadena que contenga RA en cualquier parte

Ejemplo:
Si se desea restringir los mails a cuentas de yahoo o gmail:

```
mail varchar(50) NOT NULL
  check (value LIKE '%@gmail.com' OR value LIKE '%@yahoo.com'
```

Se usa al definir chequeos a nivel de atributo

Modificación de Tablas

- Una vez creada una tabla es posible realizar algunas modificaciones en su definición (ej, agregar o quitar columnas, especificar valores por defecto, incorporar restricciones o quitarlas, etc.

ALTER TABLE NombreTabla

[ADD [COLUMN] nombre_columna TipoDato [NOT NULL] [UNIQUE]
[DEFAULT valorDefecto] [CHECK (condicion)]]

[DROP [COLUMN] nombre_columna> [RESTRICT | CASCADE]]

[ADD [CONSTRAINT nom_restr (definición-restricción)]

[DROP CONSTRAINT nom_restr] [RESTRICT | CASCADE]]

[ALTER [COLUMN] SET DEFAULT valorDefecto]

[ALTER [COLUMN] DROP DEFAULT]

ADD → incorporar
DROP → eliminar

Modificación de Tablas (ejemplos)

ALTER TABLE Alumno
ADD COLUMN concepto VARCHAR(10) DEFAULT ´Bueno´;
→ Incorpora una nueva columna concepto en Alumno

ALTER TABLE Instituto
DROP COLUMN encargado;
→ elimina la columna encargado de Alumno

ALTER TABLE Curso
ADD CONSTRAINT U_tit UNIQUE (titulo);
→ define una restricción de unicidad para el título del Curso

ALTER TABLE Ofrece
DROP CONSTRAINT Fk_Ofrece_Cur;
→ elimina la restricción de clave extranjera en Ofrece

Borrado de Tablas

DROP TABLE nombre_tabla [CASCADE | RESTRICT]

→ Se elimina la definición de la tabla y todas las filas que contiene

• Si es RESTRICT, se rechaza si hay objetos definidos a partir de la tabla (Es la opción por defecto).
• Si es CASCADE, se eliminan todos los objetos dependientes de la tabla

→ tener precaución en su uso

Actualización de la Base de Datos

Sentencias del DML para agregar, modificar y eliminar datos de tablas:

INSERT
INSERT INTO nombre_tabla [(lista_columnas)]
VALUES (lista_valores);

DELETE
DELETE FROM nombre_tabla
[WHERE (condición)] ;

UPDATE
UPDATE nombre_tabla
SET nom_columna1= valor1 [,nom_columna2= valor2 …]
[WHERE (condición)] ;

Actualización de la Base de Datos (ejemplos)

```
INSERT INTO Curso (id_curso, titulo)
    VALUES ( 208, 'Bases de Datos' );
```
→ *Al no especificarse el valor de duración, éste tomaría su valor por defecto (60)*

```
INSERT INTO Curso VALUES (134, ´Comunicación de Datos´, 45);
```
→ *no se requiere el nombre de columnas porque se insertan valores para todas*

```
UPDATE Curso SET duracion = duracion + 20;
```
→ *se agregan 20 (horas) a todos los cursos*

```
UPDATE Instituto SET encargado = 'Juan Perez'
    WHERE cod_instituto = 'ISBD';
```

```
DELETE  FROM  Ofrece;
```
→ *Elimina todas las tuplas de Ofrece!*

```
DELETE  FROM  Ofrece WHERE cod_instituto = 'ISBD';
```
→ *Elimina los vínculos de cursos ofrecidos por el instituto ISBD*