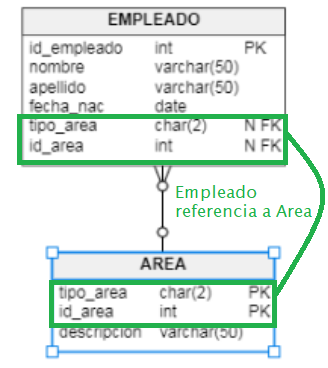
## Restricciones

“Un SGBD debe ayudar a prevenir el ingreso incorrecto de datos”



CREATE TABLE AREA (

tipo\_area char(2) NOT NULL,

id\_area int NOT NULL,

descripcion varchar(50) NOT NULL,

CONSTRAINT PK\_AREA PRIMARY KEY (tipo\_area,id\_area));

CREATE TABLE EMPLEADO (

id\_empleado int NOT NULL,

nombre varchar(50) NOT NULL,

apellido varchar(50) NOT NULL,

fecha\_nac date NOT NULL,

tipo\_area char(2) NULL,

id\_area int NULL,

CONSTRAINT PK\_EMPLEADO PRIMARY KEY (id\_empleado));

ALTER TABLE **EMPLEADO** ADD CONSTRAINT **FK\_EMPLEADO\_AREA**

**FOREIGN KEY** (tipo\_area, id\_area)

**REFERENCES AREA** (tipo\_area, id\_area) ;

Las claves extranjeras (FOREIGN KEY) especifican relaciones entre tablas y permiten mantener la consistencia entre registros de esas tablas.

### 

### Delete / update

ALTER TABLE EMPLEADO ADD CONSTRAINT

FK\_EMPLEADO\_AREA

FOREIGN KEY (tipo\_area, id\_area)

REFERENCES AREA (tipo\_area, id\_area)

**ON DELETE** acción referencial para borrado

**ON UPDATE** acción referencial para modificación;

-Cuando se borre/modifique un área hace tal cosa…

Cascade se borran los empleados que están referenciados en área.

#### Tipos de delete/update

**Borrar** (delete) un registro en la tabla AREA que está siendo referenciado en la tabla EMPLEADO por la FK

Opciones 1 - Rechazo de la operación

* NO ACTION: no permite borrar un registro cuya clave primaria está siendo referenciada por un registro en la Tabla EMPLEADO (es la opción por defecto)
* RESTRICT : misma semántica que NO ACTION, pero se chequea antes de las otras RI.

Opciones 2 - Acepta la operación y realiza acciones reparadoras adicionales: borra el registro en la tabla AREA y si es

* CASCADE: se propaga el borrado a todos los registros que referencian a dicha clave primaria mediante la FK en la tabla EMPLEADO
* SET NULL: les coloca nulos en la FK de los registros que referencian a dicha clave primaria en la tabla EMPLEADO (sólo si admite nulos)
* SET DEFAULT: les coloca el valor por defecto en la FK de los registros que referencian a dicha clave primaria en la tabla EMPLEADO (si es posible)

**Modificar** (update) la clave primaria de un registro en la tabla AREA que está siendo referenciada en tabla EMPLEADO por la FK?

Opciones 1 - Rechazo de la operación

* NO ACTION: no permite modificar un registro cuya clave primaria está siendo referenciada por un registro en la Tabla EMPLEADO (es la opción por defecto)
* RESTRICT : misma semántica que NO ACTION, pero se chequea antes de las otras RI

Opciones 2 - Acepta la operación y realiza acciones reparadoras adicionales: modifica la clave primaria del registro en la tabla AREA y

* CASCADE: propaga la modificación a todos los registros que referencian a dicha clave primaria mediante la FK en la tabla EMPLEADO
* SET NULL: coloca nulos en la FK de los registros que referencian a dicha clave primaria en la tabla EMPLEADO (sólo si admite nulos)
* SET DEFAULT: coloca el valor por defecto en la FK de los registros que referencian a dicha clave primaria en la tabla EMPLEADO (si es posible)

### Tipos de matching

La integridad referencial se satisface si para cada registro en la tabla referenciante se verifica lo siguiente:

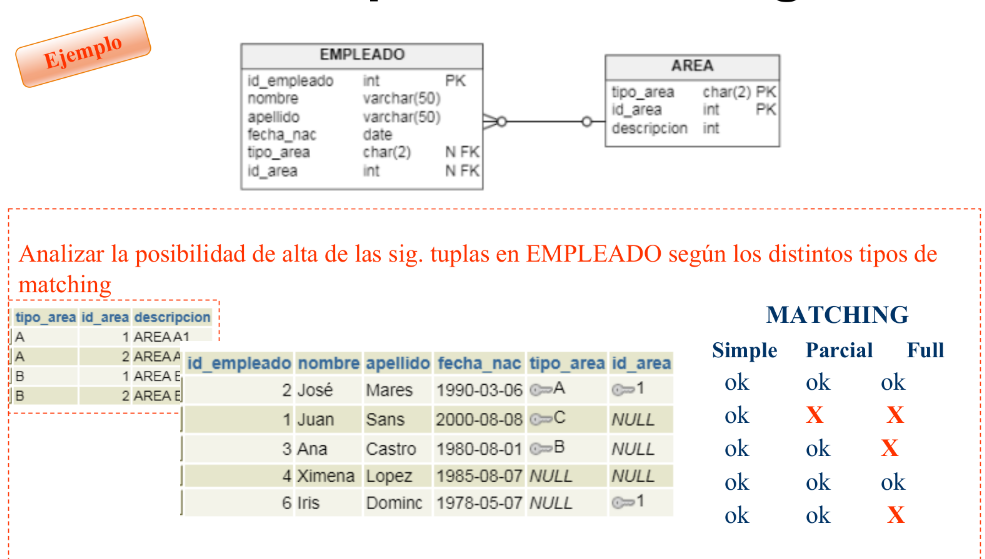
1- Ninguno de los valores de las columnas de la FK es NULL y existe un registro en la tabla referenciada cuyos valores de clave coinciden con los de tales columnas,

2- Al menos un valor en una de las columnas de la FK es NULL y puede o NO

el resto de los valores de las columnas hacer referencia a la PK (MATCH SIMPLE)

- Los valores de las columnas no nulos de la FK se corresponden con los correspondientes valores de clave en al menos en un registro de la tabla referenciada (MATCH PARTIAL)

-Todas las columnas de la FK son NULL (MATCH FULL) o hacen referencia a la PK completa



#### Restricciones declarativas

La **especificación declarativa** de RI sigue la estructura jerárquica del modelo relacional (atributo→tupla→tabla→BD):

* RI Dominio o de atributo (DOMAIN o CHECK de atributo)
* RI de tabla asociada a uno ó más atributos (CHECK de registro)
* RI de tabla asociada a varias tuplas (CHECK de tabla)
* RI generales (más de una tabla involucrada) de la base de datos (ASSERTION)

Se activan siempre que se realice alguna operación sobre los datos

afectados por la restricción. Su incumplimiento promueve el rechazo de la operación

##### Restricciones procedimentales

Otra alternativa para especificar RI → SQL Procedural:

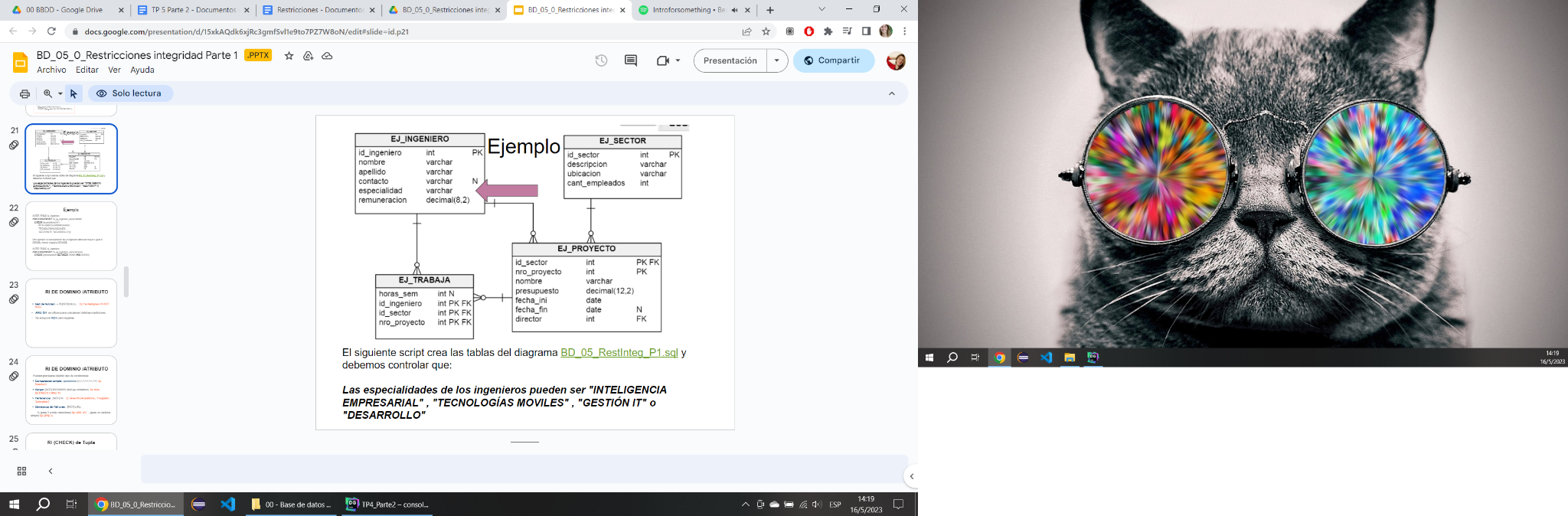
DISPARADORES (TRIGGERS): Es una pieza de código almacenada en la BD que “se dispara” automáticamente ante la ocurrencia de algún evento: procedimientos-funciones.

Recurso útil ante la imposibilidad de definir en los DBMS:

* restricciones complejas en forma declarativa
* ciertas acciones referenciales
* acciones específicas de reparación

#### CHECK ATRIBUTO

ALTER TABLE nombre\_tabla ADD CONSTRAINT ck\_tabla\_atributo CHECK (condición);



1. RESTRICCIÓN DE ATRIBUTO: Las especialidades de los ingenieros pueden ser "INTELIGENCIA EMPRESARIAL" , "TECNOLOGÍAS MOVILES" , "GESTIÓN IT" o "DESARROLLO"

ALTER TABLE ej\_ingeniero ADD CONSTRAINT ck\_ej\_ingeniero\_especialidad CHECK (especialidad IN ('INTELIGENCIA EMPRESARIAL', 'TECNOLOGÍAS MOVILES', 'GESTIÓN IT', 'DESARROLLO'));

1. La remuneración de un ingeniero debe ser mayor o igual a 25000$ y menor o igual a 250.000$

ALTER TABLE ej\_ingeniero ADD CONSTRAINT ck\_ej\_ingeniero\_remuneracion CHECK (remuneracion>=25000 AND remuneracion<=250000);

#### Test de Nulidad

IS [NOT] NULL

Ej: FechaIngreso IS NOT NULL

AND, OR se utilizan para concatenar distintas condiciones.

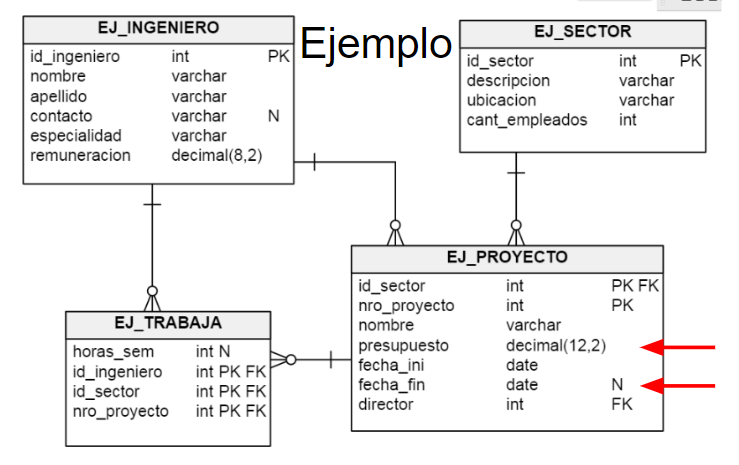
Se antepone NOT para negarlas.

Pueden plantearse distinto tipo de condiciones:

* Comparación simple: operadores (=,<,>,<=,>=,<>) Ej: Sueldo>0
* Rango: [NOT] BETWEEN (incluye extremos) Ej: nota BETWEEN 0 AND 10
* Pertenencia: [NOT] IN Ej: Area IN (‘Académica’, ‘Posgrado’, ‘Extensión’)
* Semejanza de Patrones: [NOT] LIKE

% (para 0 o más caracteres) Ej: LIKE ‘s%’ - (para un carácter simple) Ej: LIKE ‘s\_’

#### CHECK DE TUPLA (filas)



1. Los tipos de especialidad DESARROLLO sólo pueden tener id\_ingeniero que van del 1 al 7 para el resto no habría controles.

(CHECK TUPLA, dos atributos en la misma tabla)

ALTER TABLE ej\_ingeniero ADD CONSTRAINT ck\_ej\_ingeniero\_especialidad\_id\_ingeniero CHECK (id\_ingeniero BETWEEN 1 AND 7);

1. Los proyectos sin **fecha de finalización** asignada no deben superar $100.000 de **presupuesto**

(CHECK TUPLA. Tabla ej\_proyecto atributos fecha\_fin y presupuesto)

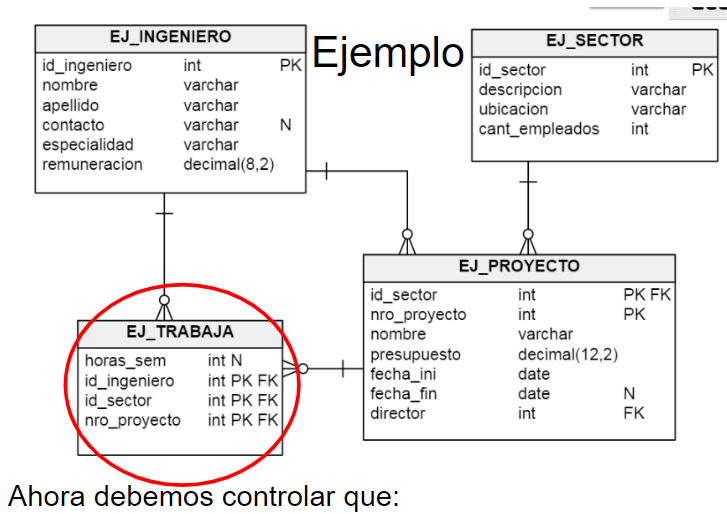
ALTER TABLE ej\_proyecto ADD CONSTRAINT ck\_ej\_proyecto\_fecha\_fin\_presupuesto CHECK ((fecha\_fin IS NULL AND presupuesto<=100000)

OR (fecha\_fin IS NOT NULL));

Le agregamos (fecha\_fin IS NOT NULL) porque sino se queja de las filas que no cumplen con is NULL.

#### CHECK TABLA

* Representa una restricción que afecta diferentes tuplas de una misma tabla
* Ámbito de la restricción: tabla
* Casos particulares: PRIMARY KEY, UNIQUE (a nivel tabla)



1. En cada proyecto pueden trabajar 10 ingenieros como máximo.

ALTER TABLE ej\_trabaja ADD CONSTRAINT ck\_trabaja\_cant\_ingenieros CHECK (NOT *EXISTS* (SELECT id\_sector, nro\_proyecto FROM ej\_trabaja GROUP BY id\_sector, nro\_proyecto HAVING *count*(\*)>10));

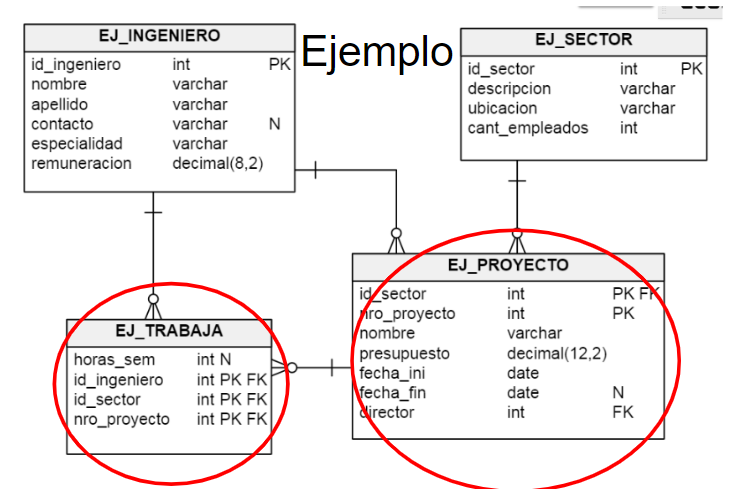
#### CHECK GLOBALES

* Permiten definir restricciones sobre un número arbitrario de atributos de un número arbitrario de tablas
* Ámbito de la restricción: base de datos
* No están asociadas a un elemento (tabla o dominio) en particular

CREATE ASSERTION NomAssertion CHECK (condición);

* Su activación se daría ante actualizaciones sobre las tablas involucradas
* Requerirían alto costo para comprobación y mantenimiento

→ los DBMS comerciales no soportan ASSERTIONS !



1. El director asignado a un proyecto debe haber trabajado al menos en 5 proyectos ya finalizados

agrupar por director, contar proyectos que tengan fecha\_fin nulos

SQL no proporciona un mecanismo para expresar la condición «para todo X, P(X)»

(P=predicado) → se debe utilizar su equivalente «no existe X tal que no P(X)»

Nuevamente la pregunta que me tengo que hacer es…. Cómo busco lo que

está mal.. cómo selecciono los proyectos que tienen un director que trabajan en

menos de 5 proyectos?

SELECT P.director, COUNT(\*) AS "cantidad proyectos"

FROM EJ\_PROYECTO P JOIN EJ\_TRABAJA T ON (P.director = T.id\_ingeniero)

JOIN EJ\_PROYECTO PP ON (PP.id\_sector = T.id\_sector AND PP.nro\_proyecto = T.nro\_proyecto) WHERE PP.fecha\_fin IS NOT NULL GROUP BY P.director

HAVING COUNT(\*) < 5;

CREATE ASSERTION CK\_PROY\_DIRE

CHECK ( NOT EXISTS

(SELECT P.director, COUNT(\*) AS "cantidad proyectos" FROM EJ\_PROYECTO P JOIN EJ\_TRABAJA T

ON (P.director = T.id\_ingeniero) JOIN EJ\_PROYECTO PP ON (PP.id\_sector = T.id\_sector

AND PP.nro\_proyecto = T.nro\_proyecto) WHERE PP.fecha\_fin IS NOT NULL GROUP BY P.director

HAVING COUNT(\*) < 5));