

4. Diseño conceptual

Tuesday, 18 February 2020 8:32 AM

4.1] Formatos de presentación

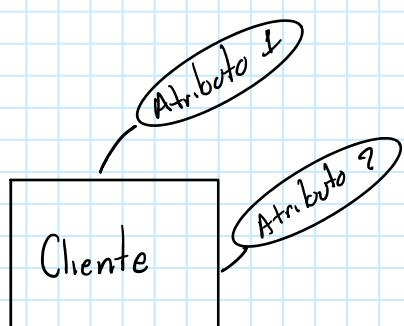
- Formato de Chen. El nombre se origina de su creador, Peter Pin-Shan Chen en 1976
- Artefacto: Modelo Entidad-Relación (E/R)
- Diseño Lógico
 - ① Formato IE | Formato de Martin | Formato crow's foot
 - ② Formato IDEF1X (1993)

4.2] Representación de entidades y atributos

► Entidades

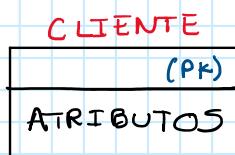
DISEÑO CONCEPTUAL

- * Se puede emplear CamelCase

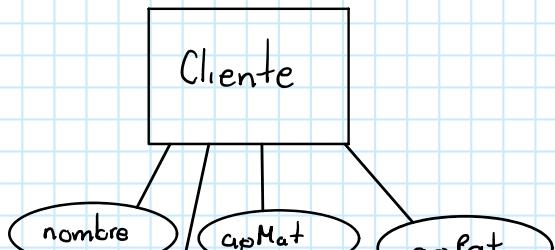


DISEÑO LÓGICO

- * Se emplea " - " para separar palabras,
- * Se elimina preposiciones y artículos
- * Generalmente todo en mayúsculas

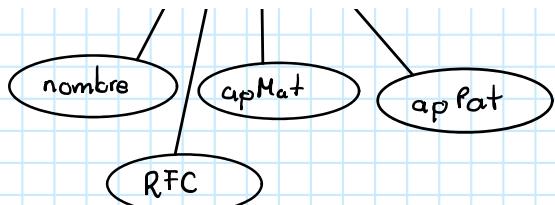


► Atributos



CLIENTE

cliente_id	numeric(10,0)	notnull	(PK)
nombre	varchar(40)	notnull	
apPat	varchar(40)	notnull	
amPat	varchar(40)	null	



apPat	<u>varchar(4)</u>	<u>notnull</u>
apMat	<u>varchar(40)</u>	<u>null</u>

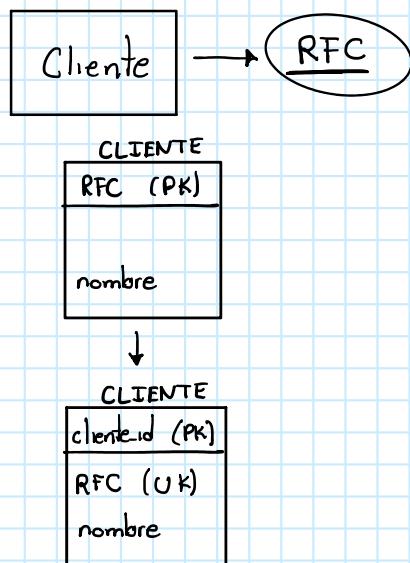
* Se puede omitir en examen

► Clasificación de Atributos

- Clave principal → Llave primaria natural
- Clave candidata → Llave primaria candidata
- Clave artificial → Llave primaria artificial
- Atributos opcionales y obligatorios
- Atributos simples y compuestos
- Atributos de valores múltiples y de valor simple
- Atributos naturales

► Clave principal (Conceptual)

- Toda entidad debe contar con su clave principal
- El atributo que se designa como clave principal se subraya
- Tratar de seleccionar claves principales que corresponden con atributos "propios" de la entidad
(Evitar a medida de lo posible claves artificiales)
- Proponer una clave artificial solo en casos donde las reglas o el enunciado no mencionen a una clave principal de forma clara o explícita

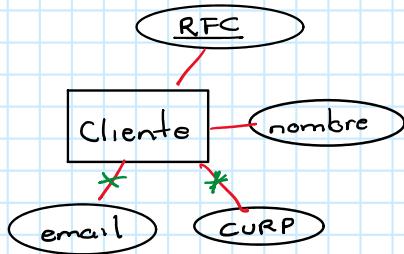


► Claves candidatas



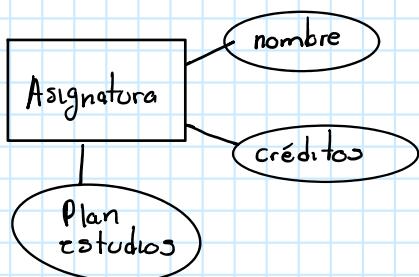
► Claves candidatas

- En diseño lógico no hay representación gráfica, sin embargo, estos atributos pueden tener asociado un constraint, unique y not null



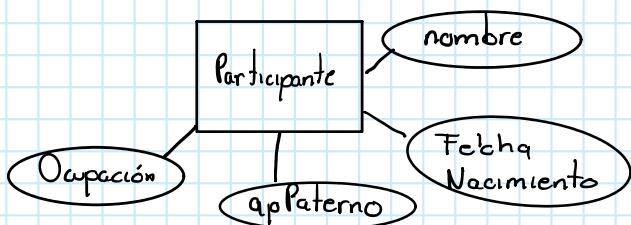
Ejemplo:

- Se desea guardar los siguientes datos: las asignaturas, nombre, créditos y plan de estudios



Ejemplo

- Se desea almacenar los datos de los participantes de un concurso, nombre, fecha nacimiento, ap Paterno y ocupación

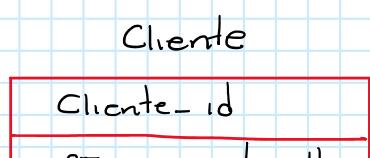


→ Cuál es la PK?

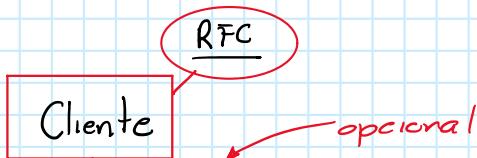
R = Al no existir alguna, se propone una artificial
ID - Participante

► Atributos opcionales y obligatorios

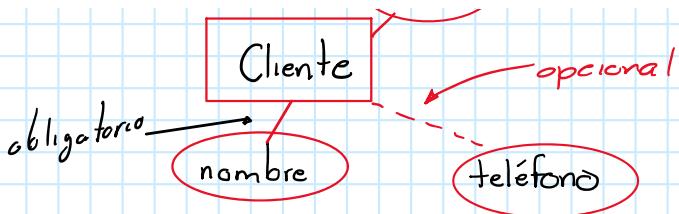
Diseño Lógico



Diseño Conceptual



Cliente_id
RFC not null
teléfono null
nombre not null



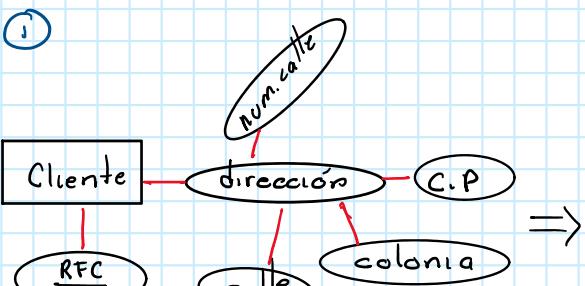
Atributo simple

- Es aquel que no puede dividirse en otros atributos

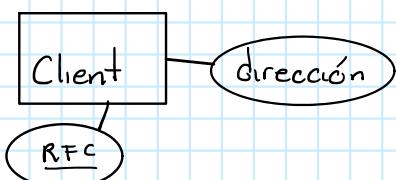
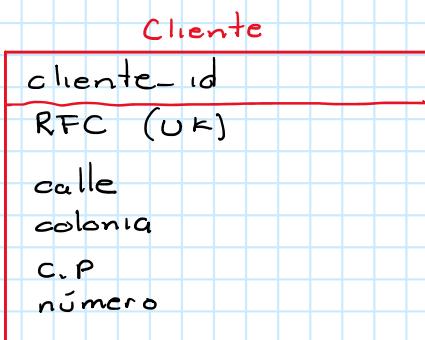
Atributo Compuesto

DISEÑO Conceptual

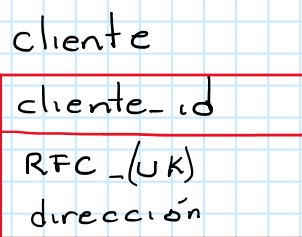
①



DISEÑO LÓGICO



=>



Atributo de valor simple

- Es un atributo que tiene un solo valor para cada instancia de una entidad

Atributo de valor múltiple

- Es un atributo que tiene más de valor para cada instancia de una entidad

Ejemplos

- Se desea almacenar el RFC del cliente y todos sus posibles teléfonos

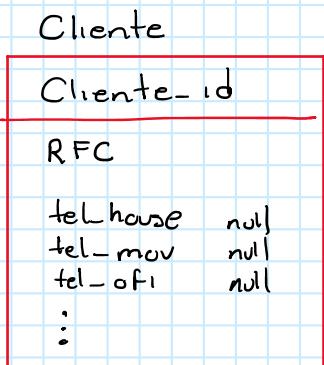
CONCEPTUAL



LÓGICO

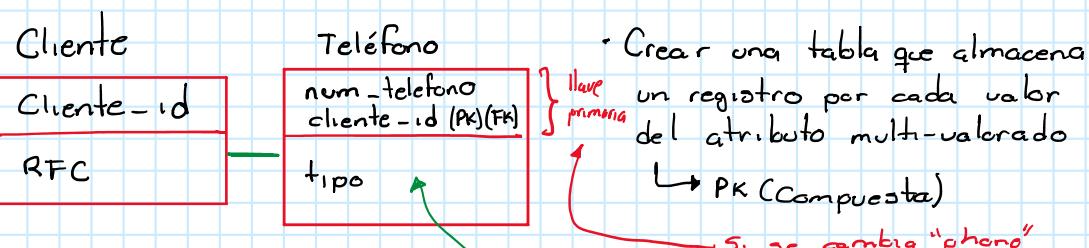
// El modelo relacional no soporta* el atributo multivalorado.
(Existen varias formas)

CASO 1

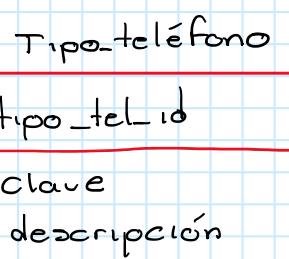
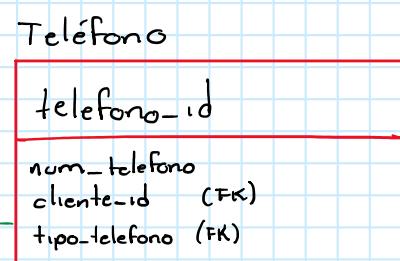
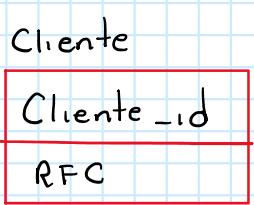


- Desventajas
 - Atributos nulos
 - d) Qué sucede si se desea guardar otro tipo de teléfono

CASO 2



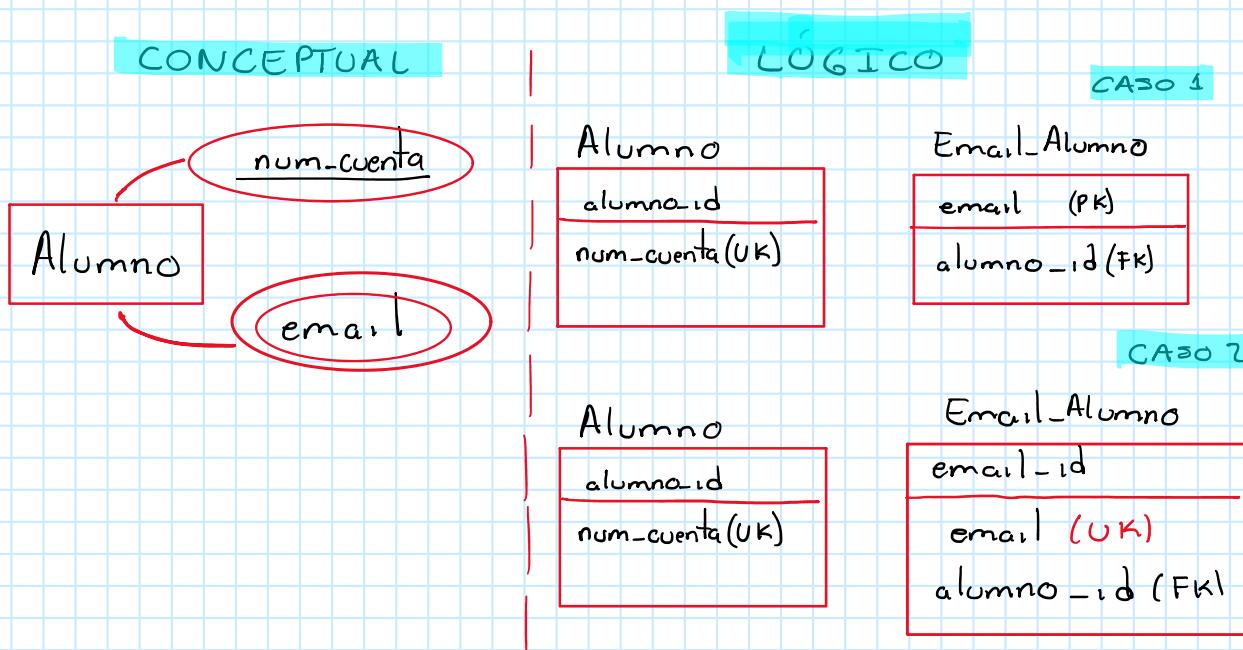
CASO 3 ← Versión normalizada



// Aunque este grado de normalización puede presentar problemas de consistencia

// Aunque este grado de normalización puede generar problemas de rendimiento.

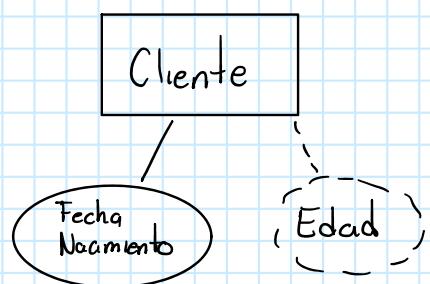
- Se descarta almacenar todas las direcciones de correo electrónico de los alumnos y su númer de cuenta



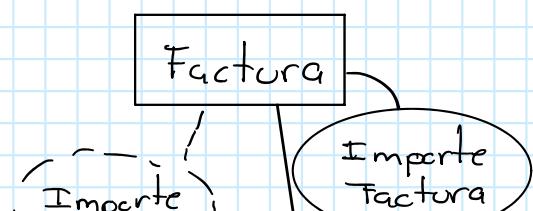
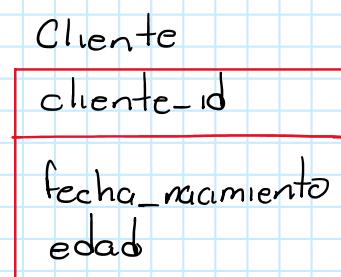
► Atributo derivado

- Atributo cuyo valor puede ser determinado o calculado a partir de fórmulas o expresiones que involucran a otros atributos.

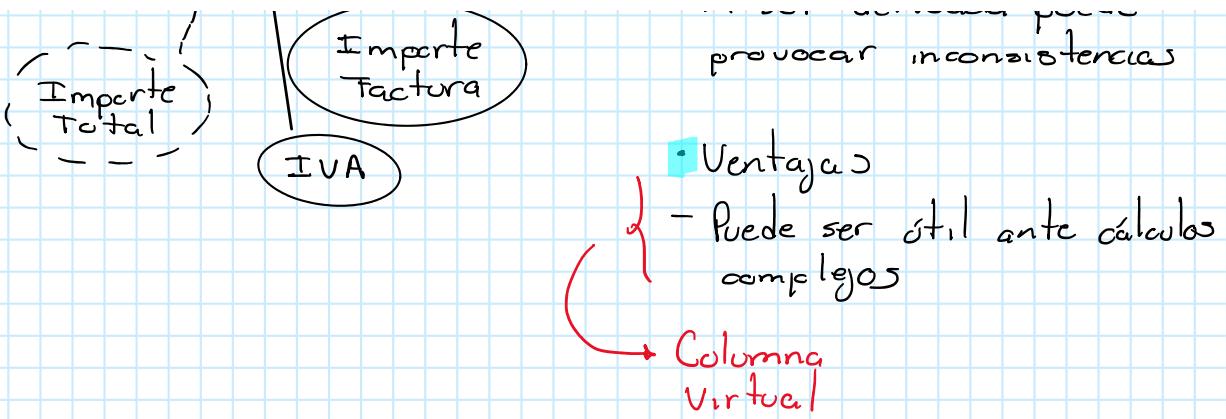
DISEÑO CONCEPTUAL



Diseño lógico



- Desventajas
 - Al ser derivada puede provocar inconsistencias



producir inconsistencias

• Ventajas

- Puede ser útil ante cálculos complejos

→ Columna Virtual

4.3 Relaciones

Thursday, 20 February 2020

8:46 AM

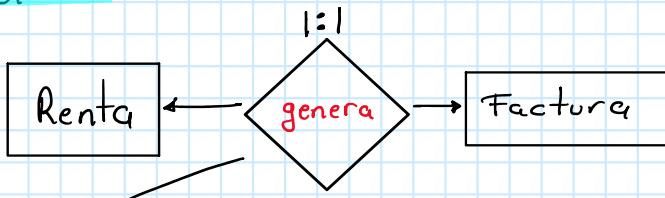
Abstract

- Representación General
- Niveles de dependencia
- Cardinalidad
- Dependencias de existencia
- Participación de una entidad en una relación
- Dependencia de identificación

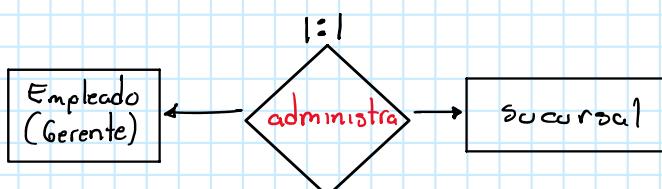
→ Representación General

Concept Design

- Rel 1:1

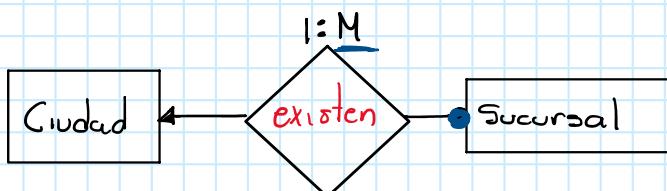


- nombre se forma un enunciado (Verbo)
 - Se lee de izq a derecha y de arriba hacia abajo
- ∴ Una renta genera una factura



Un gerente administra una sucursal

- Rel 1:M

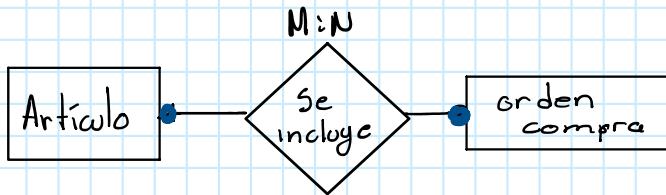


∴ En una ciudad existen varias sucursales

- es hacia donde se da el "muchos" o "varias"

- Rel M:N

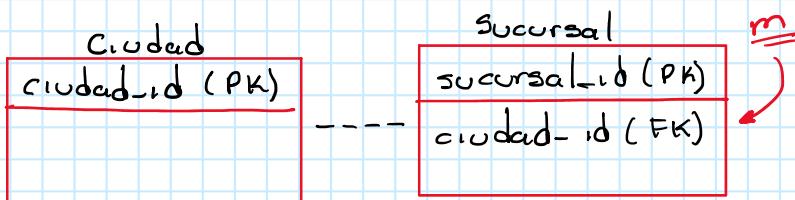
- Rel M:N



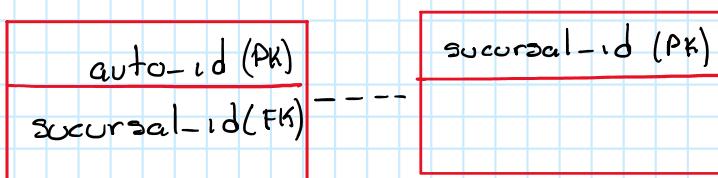
- Para diseño lógico
(Niveles de dependencia)
 - Relación suave/débil/No identificativa -----
 - Relación dura/fuerte/identificativa —————
- | - Indica qué tan fuerte es la relación entre 2 entidades

► Relación no identificativa

- La PK de la tabla Padre pasará únicamente como FK en la tabla hija (Sin formar parte de la PK)
- Empleada para relaciones 1:M aunque también puede emplearse en relaciones 1:1 y M:N



En una ciudad hay muchas sucursales



► Relaciones Identificativas

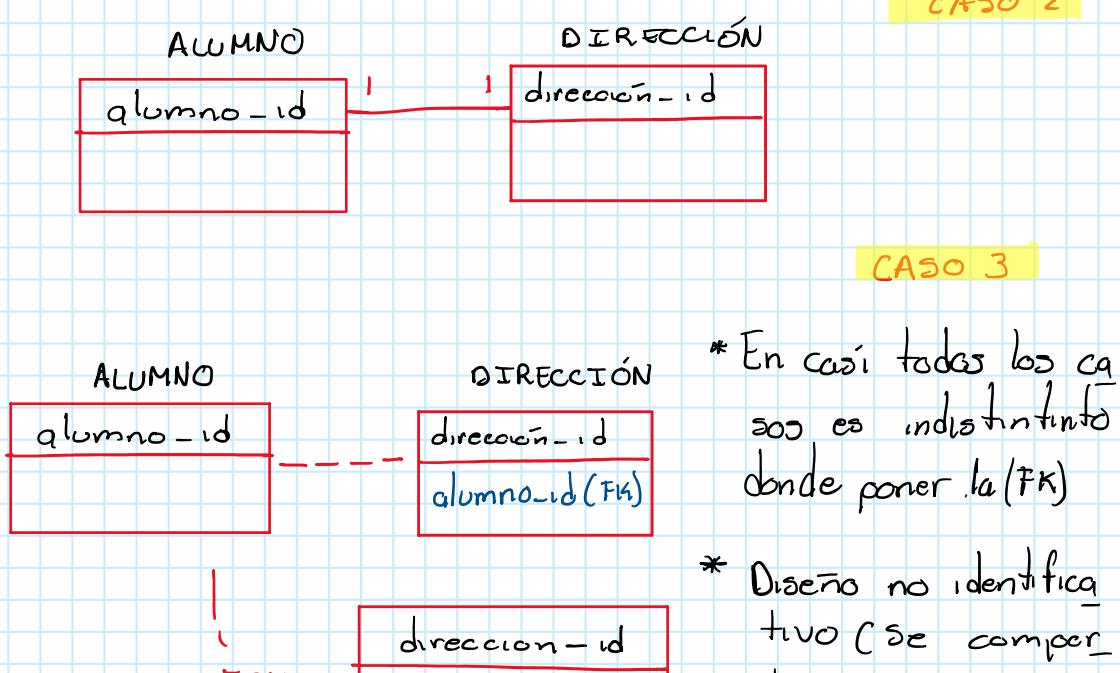
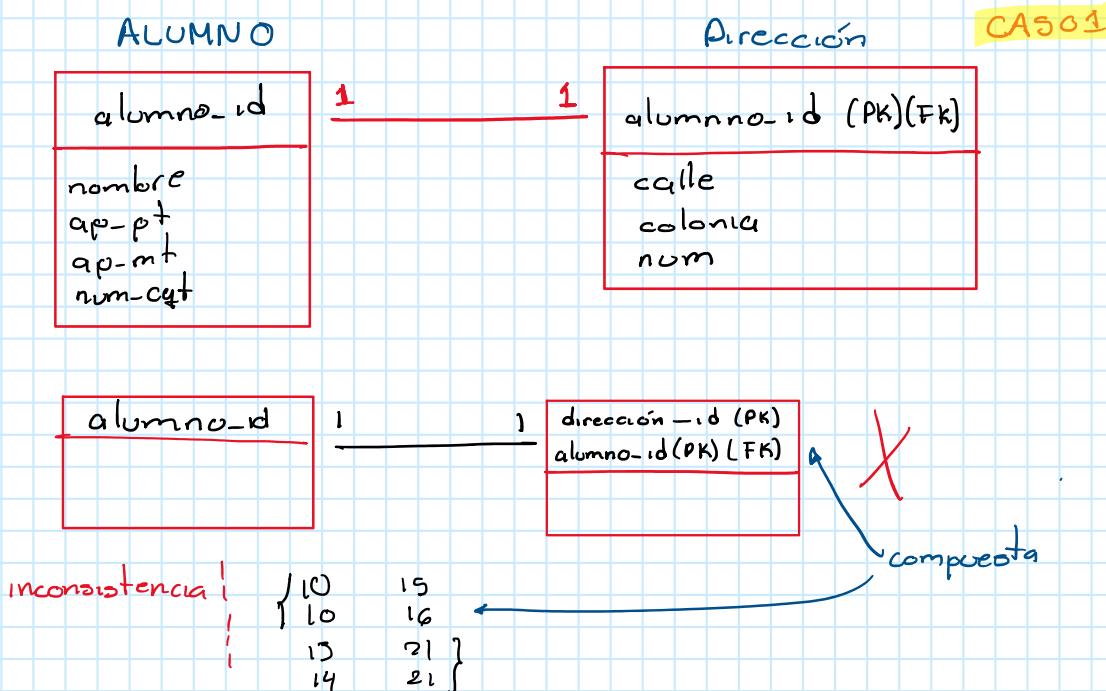
- La PK de la tabla Padre se propaga como FK y también como PK en la tabla hija

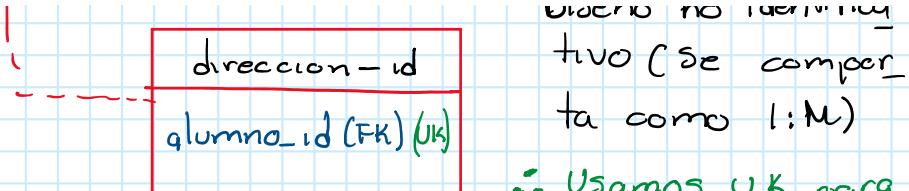
tk y también como rk en la tabla myq

- Se emplea en relaciones 1:1 y M:N

1 Ejemplo

- Se desea guardar nombre, apellidos y núm. cuenta de los alumnos.
 - Se desea guardar la dirección de los alumnos de forma separada y desglosada. las direcciones se consideran únicas por cada alumno.





vinculo no tiene univ
tivo (se comparte
ta como 1:N)

∴ Usamos UK para
forzar el 1:1

CASO 4

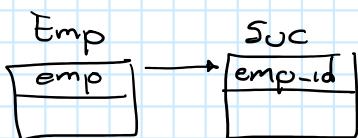


* Fusiona las tablas,
⇒ la menos propensa
a usarse

// Ejemplo

Empleado
(Gerente) > 1:1
Sucursal

* Caso 1 es algo raro para
este caso

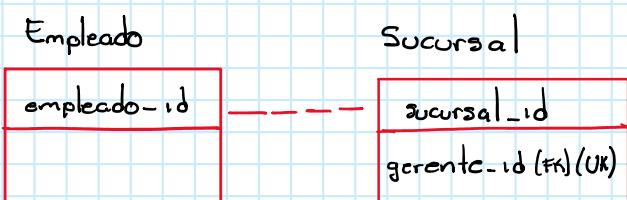


* la sucursal
se accede
mediante
el Id del
empleado

→ No tiene
sentido

* El caso contrario
haría algo incluso más
raro

∴ Caso 3 → Es la adecuada



{ Si metemos "sucursal_id" estaría
mal debido a que no todos los
empleados son gerente
La forma podría ser (NULL)(UK)

// Ejemplo 4

• Fn ante con nadrinas

// Ejemplo 4

Credencial > 1:1
Alumno

- En este caso podríamos hacer todos los casos
- ① El hijo sería credencial
- ③ Tendría que hacerse

► Cardinalidad

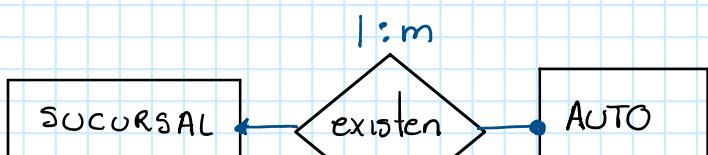
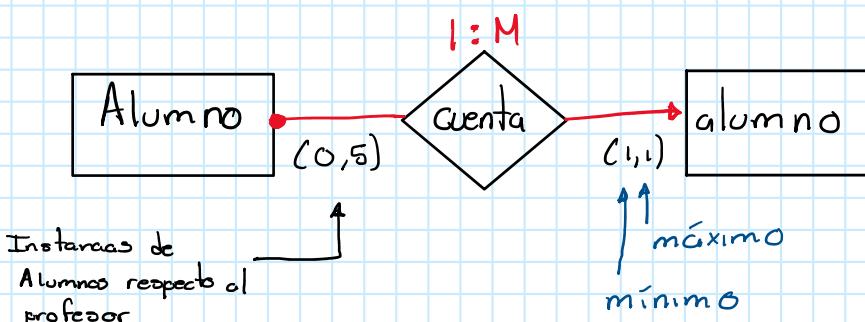
- Expresa el número mínimo y máximo de instancias (registros) asociados con una instancia de la otra entidad
- Se emplea el formato (x,y) donde
 $x \triangleq$ Valor mínimo $y \triangleq$ Valor máximo

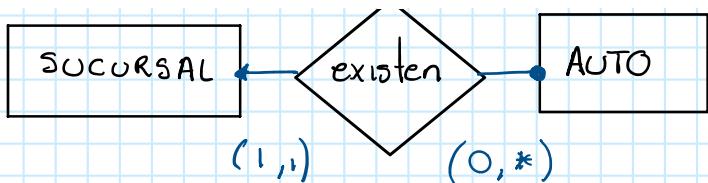
 \hookrightarrow Si no es conocido " x "
- La cardinalidad no es validada por el DBMS de forma automática, pero incluirla en los modelos permite entender claramente la forma en que se relacionan los datos.

// Ejemplos

Si lo desea: 0

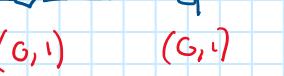
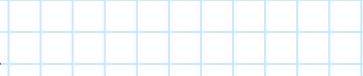
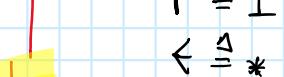
Un profesor ~~puede~~ asesora hasta ~~5~~ alumnos.
Cada alumno debe contar con un ~~1~~ asesor



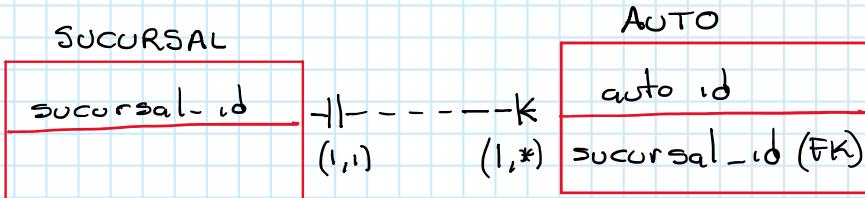


Cardinalidad | " (0,1) " o " (1,1) "

► Cardinalidad (Diseño Lógico)

Notación	Notación IDEF1X
Crows foot	
 $(1,*)$	 $(1,*)$
 $(0,*)$	 $(0,*)$
 $(1,1)$	 $(1,1)$
 $(0,1)$	 $(0,1)$
 $(1,*)$	 $(1,*)$
 $(0,*)$	 $(0,*)$
 $(1,1)$	 $(1,1)$
 $(0,1)$	 $(0,1)$
 $(1,*)$	 $(1,*)$
 $(0,1)$	 $(0,1)$
	$\parallel \stackrel{\Delta}{=} (1,1)$ $\diamond \stackrel{\Delta}{=} (0,1)$
	$\parallel \stackrel{\Delta}{=} (1,*)$ $\diamond \stackrel{\Delta}{=} (0,*)$ $\circ \stackrel{\Delta}{=} (0,1)$

//Ejemplo



4.3 Dependencia de Existencia

Tuesday, 25 February 2020 7:17 AM

- Una entidad se considera dependiente de existencia cuando sus instancias requieren de una instancia de la entidad con la que se relaciona para poder existir
- Este concepto se verifica siempre del lado de la entidad hija.

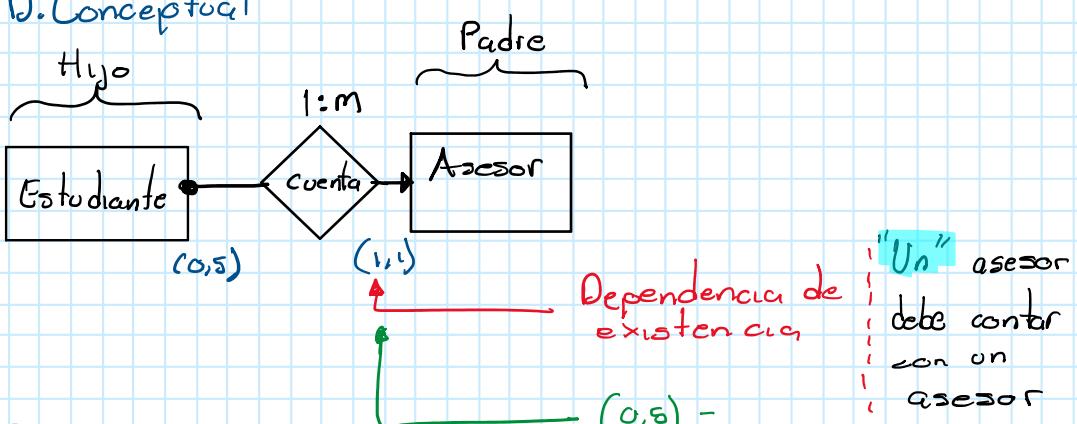
d Existirá algún mecanismo en el modelo relacional que implemente este concepto de forma automática?

- Si la condición anterior no se cumple la entidad-hija será independiente de existencia.

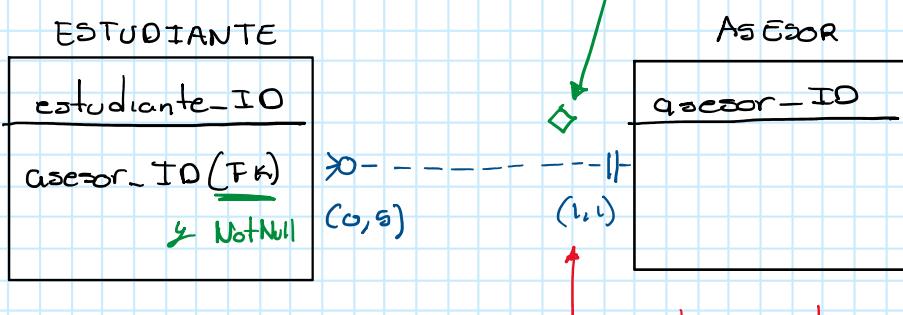
// Ejemplo

- Un estudiante **debe** contar con su asesor
- Un asesor si lo desea puede apoyar hasta 5 estudiantes

D. Conceptual



D. Lógico





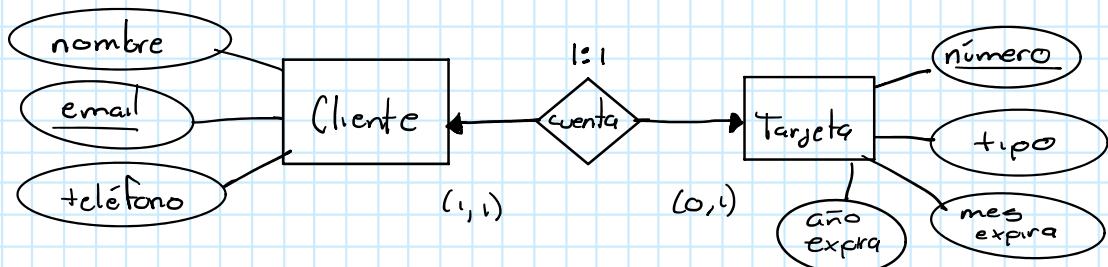
$R = \text{Constraints} \rightarrow$ De integridad \triangleq Not Null
 \hookrightarrow De referencia \triangleq FK

- Independencia de existencia
 - Colocar cardinalidad \Rightarrow
 - Indicar explícitamente que la FK es Not Null

// Ejemplo

- Un cliente puede pagar con tarjeta de crédito o con efectivo.
- Se almacena de forma separada los datos de un sólo tarjeta por cliente de forma separada: número, tipo, mes y año de expiración. Del cliente se registra nombre completo, email, y teléfono

► DISEÑO CONCEPTUAL



► DISEÑO LÓGICO

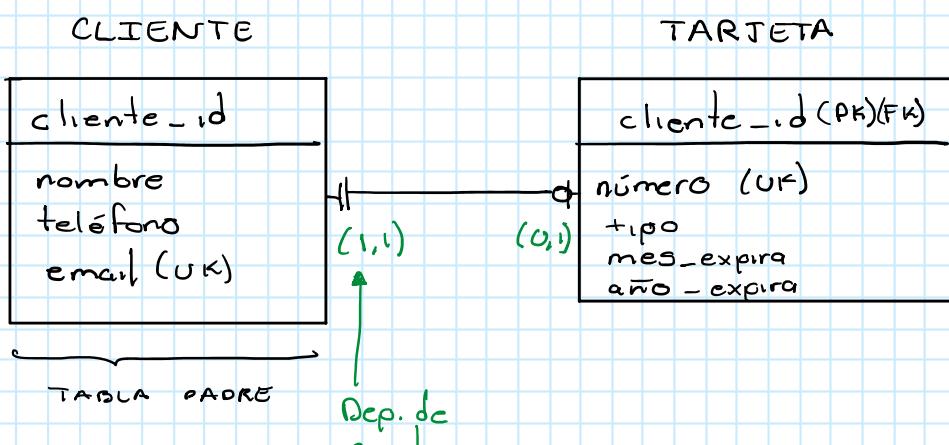


TABLA PADRE

Dep. de
existencia

► Diseño Lógico (Alternativo)

CLIENTE

cliente_ID	-	- -----	(1,1)	(0,1)
nombre				
teléfono				
email (UK)				

TARJETA

tarjeta_id	→
tipo	
mes-expira	
año-expira	
numero (UK)	
cliente_ID (FK)(UK)	

// Independencia sería con

↳ tarjeta_id (FK)(UK) NULL

► Participación de una entidad en
una Relación

- Se verifica en la entidad padre.
- Existen 2 tipos:

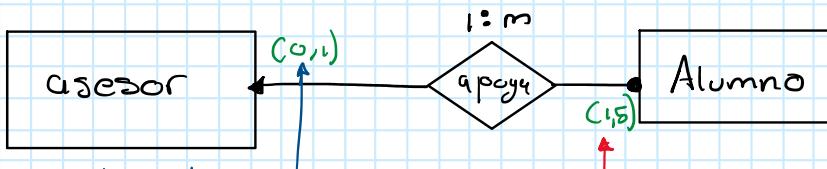
① Participación Obligatoria | Cada instancia de un Padre debe asociarse al menos a una instancia de la entidad hija.

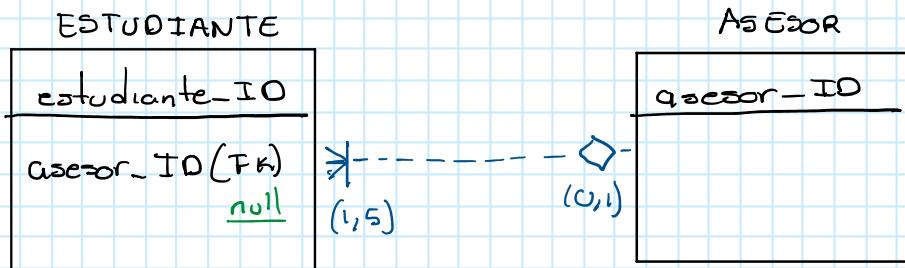
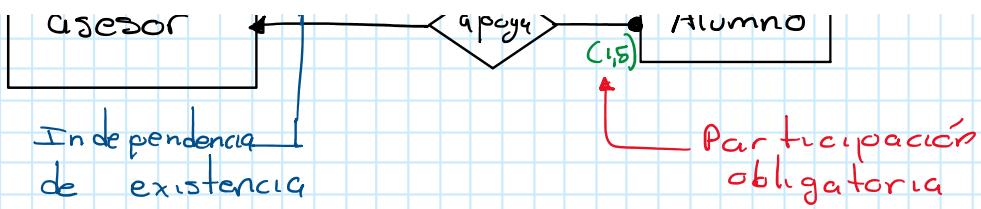
② Participación hija | Las instancias de la entidad padre pueden o no relacionarse con instancias de la entidad hija.

■ Ejemplo

→ NO hay algo que como tal válido este concepto de forma automática

- Un asesor debe apoyar de 1 a 5 estudiantes
- Un alumno si lo desea, puede solicitar a un asesor

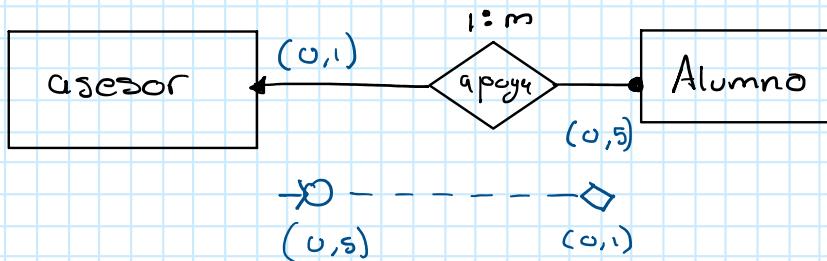




* No tiene importancia en la validación de este concepto, realmente, válida otras cosas.

// Ejemplo 2

- Un asesor puede apoyar hasta 5 estudiantes.
- Un estudiante ~~debe~~ ^{puede} contar con su asesor.



► Dependencia de identificación

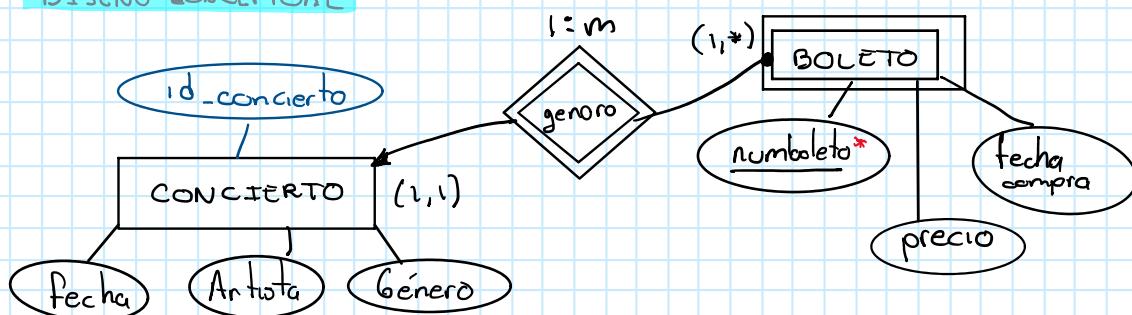
- Extensión del concepto de dependencia de existencia
- Ocurre en casos donde la PK seleccionada en la entidad ^{no es} suficiente para la identificar de forma única a sus instancias
- Lo anterior ocurre debido a que los valores de la PK se reinician o se repiten por su valor de la PK de la entidad padre
- Para resolver lo anterior, la clave principal de la entidad

- Para resolver lo anterior, la clave principal de la entidad hija se formará propagando la PK de la entidad padre formando una clave compuesta

Ejemplo

- Se desea guardar los datos de los conciertos que se realizan en un recinto, la fecha, el artista y el género.
- Se desea almacenar los datos de los boletos vendidos en cada concierto:
número boleto, precio, fechadecompra
- En cada concierto, el # de boleto, inicia en 00001

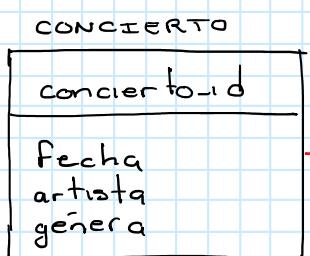
DISEÑO CONCEPTUAL



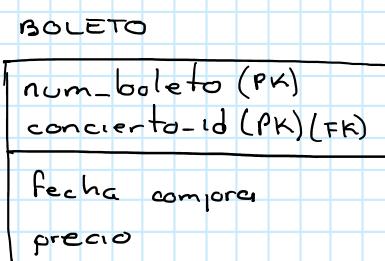
* Puede que se repita para cada concierto
 \therefore Clave compuesta

* Para el diseño conceptual se marca el nombre y la respectiva entidad
 \hookrightarrow Entidad débil

DISEÑO LÓGICO

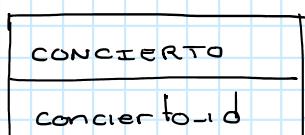


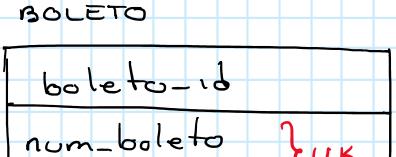
(1,1) (1,*)



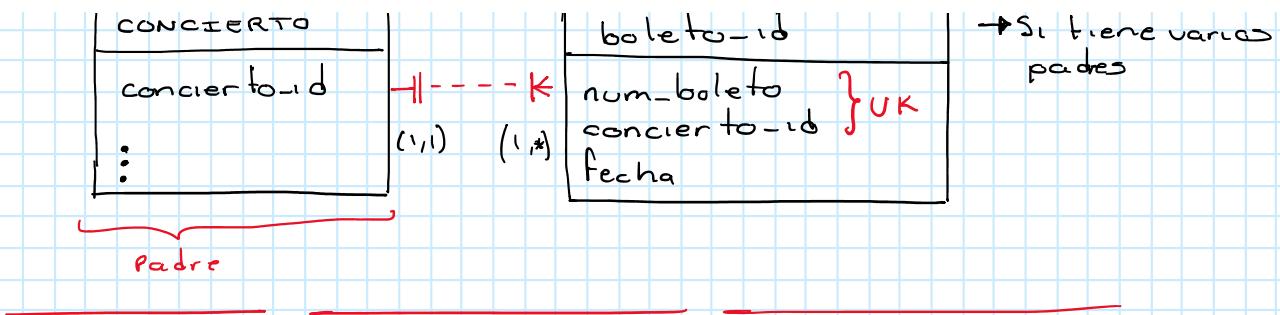
→ Si ya no se relaciona con otras (Podemos usar esta versión)

No tener más de un parente





→ Si tiene varios padres



► Grado de una relación

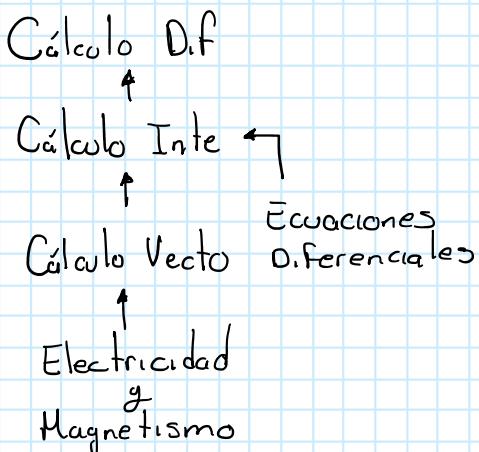
- Consiste en clasificar a las relaciones entre entidades, considerando a el número entre entidades en vez de considerar instancias.
- Unarias / Binarias / Ternarias

► Relación Unaria

- Conocidas también como relaciones recursivas
- Una instancia de una Entidad puede asociarse con varias instancias de la misma entidad.

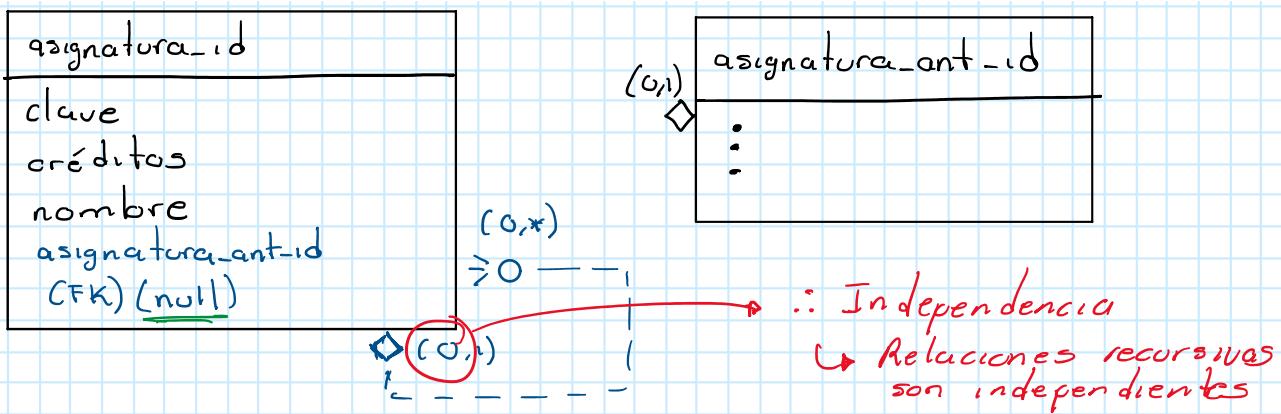
// Ejemplo

- Una asignatura puede requerir de una asignatura antecedente
- Una asignatura antecedente puede ser requerida para cursar otras asignaturas



ASIGNATURA

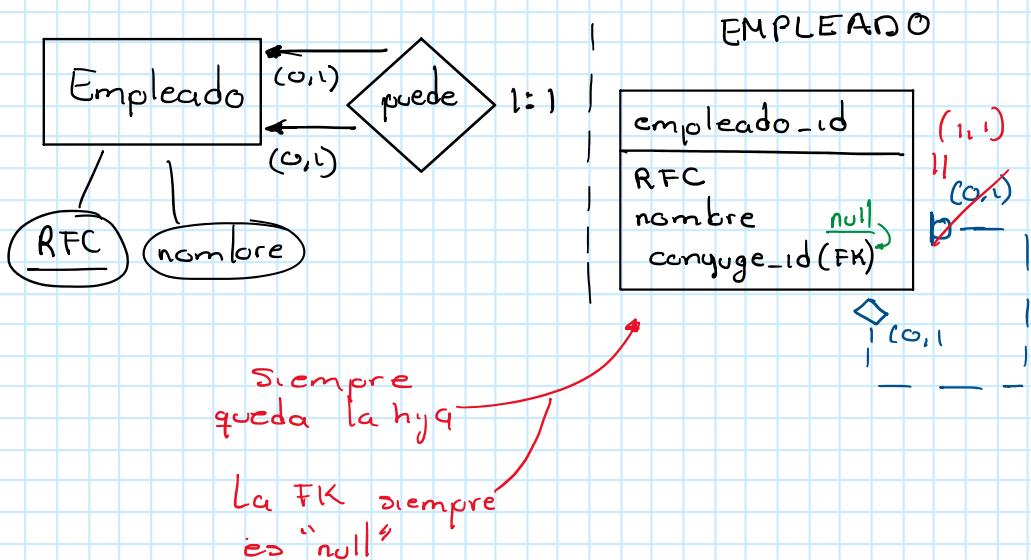
ANTECEDENTE



* Por el cero de la antecedente, toma en cuenta que hay materias que no tienen antecedente

// Ejemplo

- Se desea almacenar los datos de los empleados RFC, nombre.
- Algunos empleados son casados, algunos de ellos se casaron con otros empleados



► Relaciones Ternarias

- Participan 3 entidades

// Ejemplo

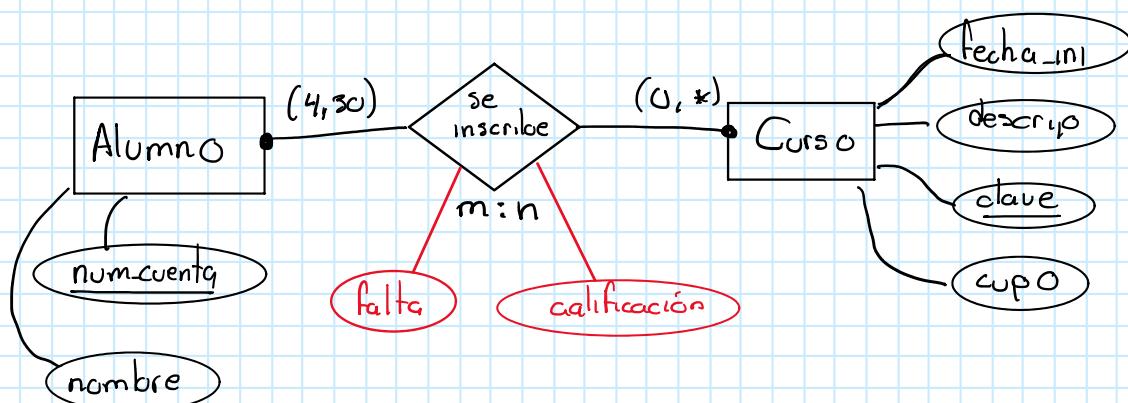
- Se desea almacenar los datos de los cursos de una escuela: clave, descripción, fecha, inicio, cupo máximo
- Se almacena los datos de los Alumnos, nombre, num cuenta

uuve, descripción, fecha_ini, cupo maximo

- Se almacena los datos de los Alumnos, nombre, num_cuenta
- Se desea registrar la calificación y el total de faltas que obtuvieron los alumnos en cada curso
- Un curso requiere al menos de 4 alumnos para ser abierto
- Cupo máximo de 30 personas
- No todos los alumnos se inscriben a cursos, en un mismo periodo el alumno puede inscribir hasta 3 cursos

* Faltas dependen del curso y alumno

↳ En relaciones M:N hay casos "combinados"



ALUMNO

alumno_id
num_cuenta
nombre
= (1,1)

CURSO

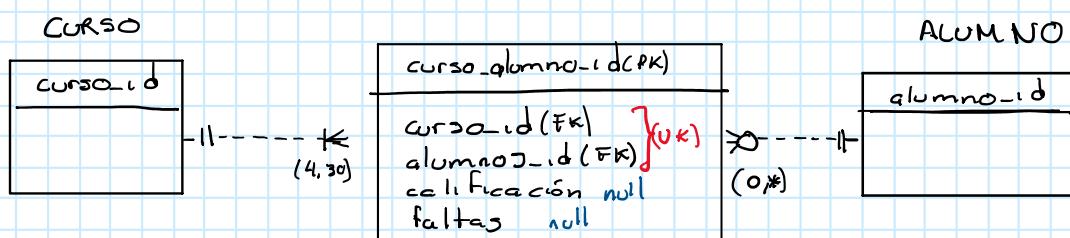
curso_id
clave
descripción
cupo
fecha_ini

alumno_cuenta
curso_id (PK)/FK
alumno_id (PK)/FK
Faltas (null)
calificación (null)

* Las cardinalidades en padres $(1,1) \triangleq 1$

Recuerda que siempre son $(0,1)$ o $(1,1)$

pero al ser ternaria sólo quedaría $(1,1)$



* Debes permitir que sean nulos debido a que al crearse no tendrán datos

4.4 Tipos de datos en diseño lógico

Monday, 2 March 2020 6:00 PM

- Juego de caracteres | Caracteres son representados → | ASCII (c)
carácteres | por un código | $\hat{=}$ 1 byte
| ISO (Windows)
| UTF - 8 (Linux)
| UNICODE (Java)

CADENAS

Clasificación de los tipos de datos para representar cadenas de caracteres

La siguiente tabla muestra la clasificación de las cadenas de caracteres. Del lado izquierdo, representa al tipo de dato definido por el estándar SQL, y las demás columnas representan la forma en la que cada manejador implementa al tipo de dato.

SQL 2003 (especificación)	ORACLE 11g (ISO)	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
CHAR [ACTER] [n]	CHAR [ACTER] [(n)]	CHAR [ACTER] [(n)]	CHAR [ACTER] [(n)]	CHAR [ACTER] [(n)]	CHAR [(n)]
CHAR [ACTER]	CHAR [ACTER]	CHAR [ACTER]	CHAR [ACTER]	CHAR [ACTER]	VARCHAR (n) ó TIMESTAMPTZ ó TEXT ó MEDIUMTEXT ó LONGTEXT
VARCHAR (n) ó VARCHAR2 (n)	VARCHAR (n) ó VARCHAR2 (n)	VARCHAR (n) ó LONGVARCHAR	VARCHAR (n) ó LONGVARCHAR	VARCHAR (n) ó TEXT	BINARY [(n)] VARBINARY (n)
CLOB ó CHARACTER LARGE OBJECT	CLOB ó LONG[VARCHAR]	CLOB [(n)][K(M, G)]	VARCHAR(MAX)	TEXT	

- Longitud fija → Es más rápido su manipulamiento que el de longitud variables
- [] $\hat{=}$ Puede declararse o no el tamaño
- CLOB $\hat{=}$ Es la extensión del varchar

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
NATIONAL CHAR [ACTER] [(n)] o NCHAR [(n)] o CHARACTER SET <character-name>	NATIONAL CHAR [ACTER] [(n)] o NCHAR [(n)]	GRAPHIC [(n)]	NATIONAL CHAR [ACTER] [(n)] o NCHAR [(n)] o NTEXT	-----	NATIONAL CHARACTER [(n)] o CHAR(n) o NCHAR SET <name> o NCHAR (n)
NATIONAL CHAR [ACTER] VARYING (n) o NCHAR VARYING (n) o CHARACTER VARYING (n) CHARACTER SET <name>	NATIONAL CHAR [ACTER] VARYING (n) o NCHAR VARYING (n) o NCHAR CHARACTER VARYING (n)	VARGRAPHIC [(n)] o LONG VARGRAPHIC (n)	NATIONAL CHARACTER VARYING [(n)] o NTEXT	-----	VARCHAR (n) CHARACTER SET <name> o NATIONAL VARCHAR (n) o NCHAR CHARACTER VARYING (n)
NATIONAL CHARACTER LARGE OBJECT	NCLOB	DBCLOB [(n)] [KLOB])	NCHAR (n)	-----	-----

- Las palabras entre [] son opcionales, las palabras entre <> se sustituyen por un valor.

Rangos reportados por los principales proveedores para cadenas de caracteres:

Tipo de dato	Oracle	Tipo de dato	DB2	Tipo de dato	SQL Server	Tipo de dato	MySQL
CHAR	1-2000	CHAR	1-254	CHAR	1-8000	CHAR	1-255
VARCHAR2	1-8000	VARCHAR	1-32762	VARCHAR	1-8000	VARCHAR	1-65535
NCHAR (n)code)	1-2000	LONG	1-32760	TEXT	2 GB	TEXT	0-255
NVARCHAR (n)code)	1-8000	CLOB	2 GB	NCHAR NVARCHAR	1-8000	NCHAR NVARCHAR	0-65535
CLOB	8 TB	GRAPHIC	1-127	NTEXT	1 GB	MEDIUM TEXT	0-16777215
NCLOB	8 TB	VARGRAPHIC	1-16, 336	VARCHAR (MAX)	2 GB	LONGTEXT	0-4294967295
		LCLOB	16,350				
		VARGRAPHIC					

Tipo de dato	PostgreSQL
CHAR	1-1GB
VARCHAR	1-1GB
TEXT	1-1GB

Cadenas binarias

► Cadenas Binarias

Se emplean para almacenar secuencia de bytes en la base de datos, es decir, se almacenan archivos binarios: fotos, música, video, huellas, documentos, etc. El tipo de dato más común empleado es BLOB (Binary Large Object), de forma similar a CLOB, no es necesario escribir la longitud máxima que puede almacenarse en la BD.

Clasificación:

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
BLOB o BINARY LARGE OBJECT	BLOB o LONGBLOB o RAW (n)	BLOB [(n)] KLOB))	VARGRAPHIC (MAX) o VARGRAPHIC (n) o IMAGE	BYTEA	BINARY (n) o VARBINARY (n) o TINYBLOB o BLOB o MEDIUM BLOB o LONG BLOB

Rangos cadenas binarias.

Tipo de dato	Oracle	Tipo de dato	DB2	Tipo de dato	SQL Server
BLOB	8 TB	BLOB	2 GB	BINARY	8000
RAW	4000 BYTES			VARGRAPHIC	8000
				IMAGE	2147483647
				VARGRAPHIC (MAX)	2 GB

Extensión en
una columna

Tipos de datos numéricos.

Existen 2 clasificaciones de los tipos de datos numéricos:

- Tipos de datos numéricos exactos
- Tipos de datos numéricos aproximados.

Tipos de datos numéricos exactos

Los tipos de datos numéricos exactos pueden ser números enteros finitos o números con parte decimal finita.

Al definir un atributo de una tabla con un tipo de dato numérico exacto se puede definir 2 elementos:

- **Precisión:** Número total de dígitos. Parte entera + parte decimal.
- **Escala:** Número de dígitos que formarán a la parte decimal.

Por lo anterior, el número máximo de dígitos que podrá tener la parte entera de un valor numérico será:

$$\#dígitosParteEntera = \text{precisión} - \text{escala}.$$

Ejemplo.

¿Cuál será el dominio de la siguiente columna? PRECIO NUMBER (10, 4)

Precisión = 10

Escala = 4

Parte entera = 10-4=6

Dominio: [0, 999999, 9999]

Clasificación de los tipos de datos numéricos exactos.

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
INT [INTEGER]	NUMBER(11)	INTEGER ó BIGINT	INT [INTEGER] ó BIGINT	INTEGER ó MEDIUMINT [MEDIUMINT] ó SERIAL ó BIGINT	INT [INT] ó MEDIUMINT [MEDIUMINT] ó BIGINT [BIGINT]
SMALLINT	SMALLINT	SMALLINT	SMALLINT ó TINYINT	SMALLINT	SMALLINT [SMALLINT] ó TINYINT [TINYINT]
NUMERIC [(P,F)]	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER
DECIMAL [(P,F)]	[(P,F)] ó DECIMAL	[(P,F)] ó DECIMAL	[(P,F)] ó DECIMAL	[(P,F)] ó DECIMAL	[(P,F)] ó DECIMAL
NUMBER [(P,F)]	[(P,F)]	[(P,F)]	[(P,F)]	[(P,F)]	[(P,F)]

Tipos de datos numéricos aproximados.

Se emplean cuando no se conoce con exactitud los valores de la precisión y/o escala que puede tener los valores de una columna.

Clasificación de los tipos de datos numéricos aproximados.

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
FLOAT [(P)] ó NUMBER	FLOAT [(P)]	FLOAT [(P)]	FLOAT [(P)]	-----	FLOAT [(P,F)]
REAL	REAL	REAL	REAL	REAL	REAL [(P,F)]
DOUBLE PRECISION	DOUBLE PRECISION	DOUBLE	DOUBLE PRECISION	DOUBLE PRECISION	DOUBLE [PRECISION]

Rango tipos de datos numéricos para Oracle:

- INTEGER, SMALLINT se convierten a NUMBER (38)
- NUMERIC, DECIMAL se convierten a NUMBER
- Rango: 1×10^{38} a 9×10^{38} (38 dígitos).
- Oracle permite la definición de escalas negativas empleadas para redondear. Por ejemplo:
 - NUMBER (10,2), 6,345,768,903,678 será redondeado a 6,345,768,904

Observar que, para el caso de Oracle, en todos los tipos que define el estándar, Oracle lo representa a través del tipo de dato NUMBER. Por lo anterior, se recomienda que todos los tipos de datos numéricos (tanto exactos como aproximados) sean representados empleando NUMBER.

Tipos de datos para el manejo de fechas.

Esta clasificación de tipos de dato es la más complicada en el sentido de que prácticamente ningún manejador sigue el estándar SQL.

A nivel del estándar:

→ **Fechas**

Clasificación de los tipos de datos para manejo de fechas.

SQL Estándar (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
DATE	DATE	DATE	DATETIME ó SMALLDATETIME	DATE	DATE
TIME [(WITH TIME ZONE)]	DATE	TIME	DATETIME ó SMALLDATETIME	TIME [(P)] [(WITH (OUT) TIMEZONE)]	TIME
TIMESTAMP [(P)] [(WITH TIME ZONE)]	DATE ó TIMESTAMP [(P)] [(WITH TIME ZONE)]	TIMESTAMP	DATETIME SMALLDATETIME	TIMESTAMP [(P)] [(WITH (OUT) TIMEZONE)]	DATETIME TIMESTAMP
INTERVAL	INTERVAL DAY [(P)] TO SECOND [(P)] ó INTERVAL YEAR [(P)] TO MONTH	-----	-----	INTERVAL [(Field[(P)])]	YEAR

Observar nuevamente, para el caso de Oracle, se emplea el tipo de dato DATE para representar fechas, tiempo o ambas.

Otros tipos de datos.

BOOLEAN

- Se agrega al estándar SQL 2003. Posibles valores para este tipo de dato: TRUE, FALSE.
- Oracle, DB2, SQL server no tienen este tipo de dato. Para el caso de Oracle se emplea NUMBER (1, 0). Tipicamente 1 significa TRUE, 0 significa FALSE.
- En MySQL: BOOL, BOOLEAN, TINYINT (1)
- En PostgreSQL: BOOLEAN

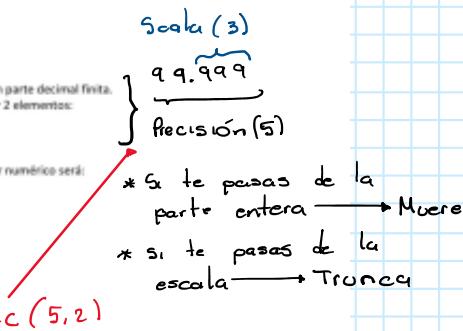
→ **ROWID:** Como se revisó en temas anteriores, ROWID es exclusivo de Oracle, se emplea para almacenar una dirección única para cada registro de una tabla que se crea para localizar los datos de una forma directa.

→ **UROWID:** Similar a ROWID, solo que este se emplea para tablas indexadas.

→ **BFILE:** Tipo dato exclusivo de Oracle, se emplea para leer datos binarios que están almacenados fuera de la base de datos (se considera una especie de puntero hacia la ubicación física del archivo).

→ **XML:** Tipo de dato empleado para almacenar documentos XML. Permite el manejo y explotación directa del documento a través de SQL en combinación de lenguaje empleado para navegar sobre los elementos de un documento XML.

→ **Modelado de casos de uso**



Base de datos para una empresa que administra colegios.

- Las siguientes reglas de negocio resultaron de las entrevistas con los dueños de una empresa que administra colegios particulares la cual ha solicitado el diseño de una base de datos.
- A. Al final se anexa una propuesta de diseño conceptual. Completar el diagrama indicando los nombres de las entidades en los espacios correspondientes.
 - B. Realizar el diseño lógico empleando para ello la transformación del modelo E/R del punto anterior a un modelo relacional con notación crow's foot.
 1. Cada colegio está formado por varias escuelas. De cada colegio se requiere almacenar su nombre.
 2. Para cada escuela se requiere almacenar el nombre y su clave. Cada escuela es administrada por un director el cual a su vez es un profesor. Cada director puede administrar una sola escuela. Para un profesor no es requisito que administre una escuela.
 3. Cada escuela está formada por varios departamentos. Se requiere almacenar el nombre del departamento.
 4. Cada departamento está dividido en un conjunto de áreas funcionales. Se desea almacenar los nombres de las áreas funcionales del departamento.

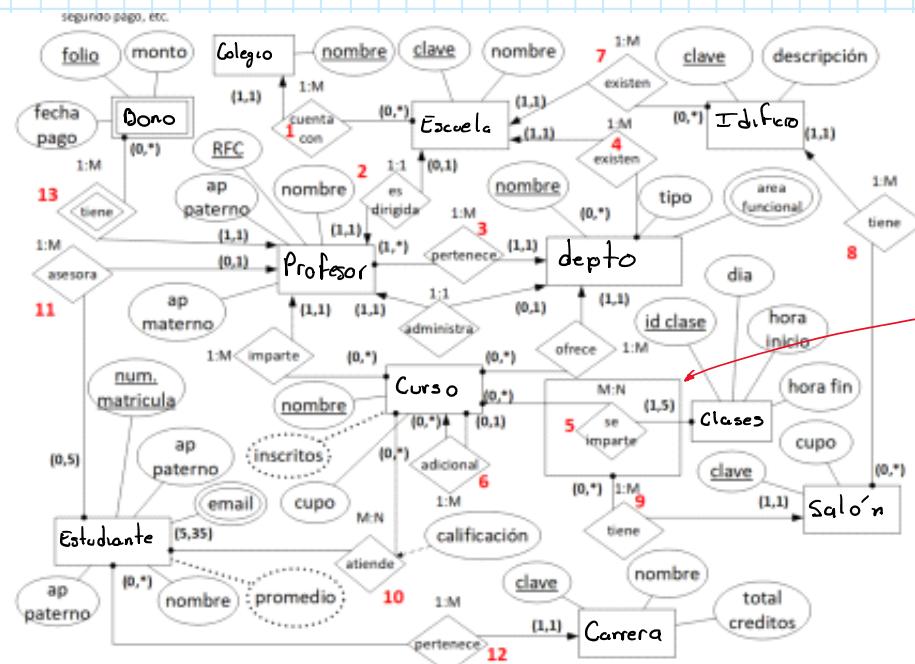
Ing. Jorge A. Rodríguez Campos

jorgejr@gmail.com

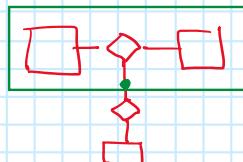
Página 7

Material de apoyo.

- FI-UNAM
5. Cada departamento ofrece cursos; otros departamentos se clasifican, por ejemplo, como departamentos de investigación por lo que no ofrecen cursos. Se debe almacenar el tipo de departamento ["de investigación", o "regular"]. Cuando se registra el curso se debe indicar en nombre del curso, el cupo máximo, número de alumnos inscritos y el departamento al que pertenece.
 6. Cada curso se imparte en varias clases a la semana. Para cada clase se indica la hora inicio, hora fin, el día de la semana y el salón [an] mismo curso puede impartirse en varios salones].
 7. Para cada curso, en una semana se pueden dar como mínimo una clase, y como máximo 5 clases. El curso lo imparte un único profesor.
 8. El colegio requiere guardar los siguientes datos de cada salón: clave de 5 caracteres, y cupo máximo. No en todos los salones se imparten cursos.
 9. Para contar con una mejor comprensión de un curso, en algunos casos se recomienda tomar un curso previo. Para dichos casos se requiere asociar el curso recomendado.
 10. Cada profesor pertenece a un departamento. Uno de estos profesores es el encargado de administrarlo (Jefe de departamento). Los datos que se registran de un profesor son: Nombre, apellidos, y RFC. Algunos profesores no imparten cursos.
 11. Un estudiante puede estar inscrito en 1 y hasta en 6 cursos en un mismo periodo. Cada curso puede tener como mínimo 5 estudiantes, y como máximo 35. Al final del periodo se registra la calificación del estudiante. Se requiere almacenar nombre, apellido del estudiante y número de matrícula. El estudiante debe proporcionar al menos un correo electrónico como medio de contacto.
 12. Cada estudiante pertenece a una carrera. El colegio cuenta con un catálogo de carreras en el que se almacena clave de la carrera, nombre de la carrera, y total de créditos.
 13. Cada estudiante puede tener un asesor, el cual es un profesor. Solo aquellos profesores que así lo deseen pueden asesorar hasta 5 estudiantes.
 14. Cada clase se imparte en un salón, y cada salón se encuentra localizado en un edificio perteneciente a la escuela. Se requiere almacenar los datos del edificio: Clave de 2 letras, y descripción. No en todos los edificios se imparten cursos.
 15. Cada año un asesor puede recibir varios bonos por su buen desempeño. Se requiere registrar la fecha de pago, el monto del bono y un número consecutivo (folio de pago) que indica el número de pago. El folio está formado por 5 dígitos. Ejemplo: 00001 para el primer pago, 0002 para el segundo pago, etc.



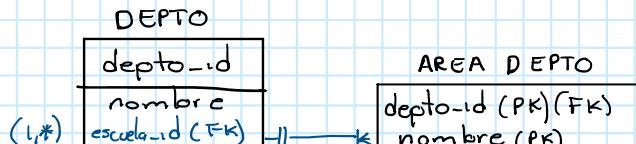
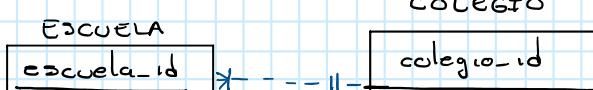
Agregación
(Tiene la solución en color rojo)

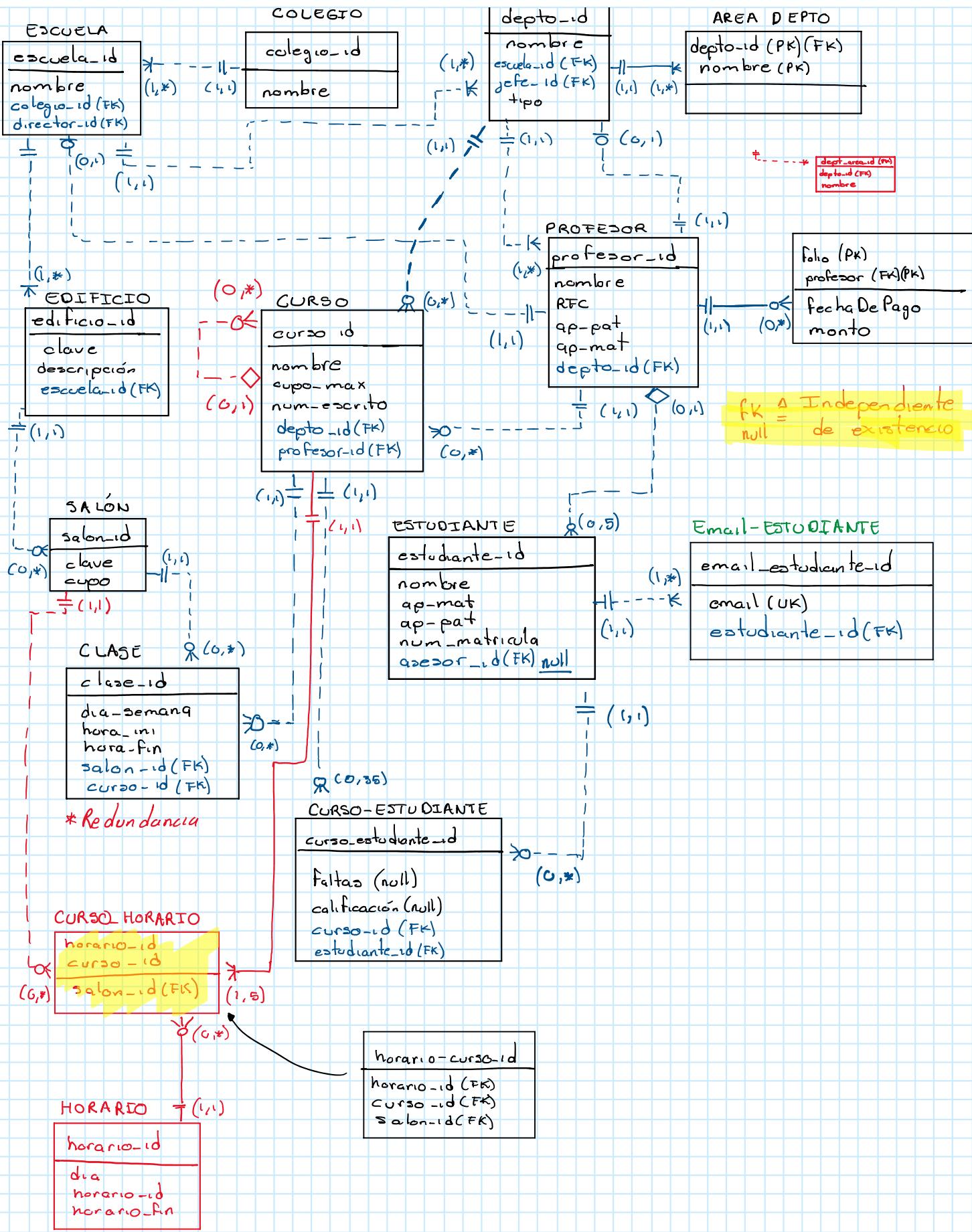


ENTIDADES

- Colegio
- Escuela
- Director
- Estudiante
- Bono
- Departamento
- Área Funcional
- Salón
- Clasificación

- Carrera
- Clase
- Edificio
- Boleta





5.4 Modelado de datos extendidos

Friday, 6 March 2020 7:17 AM

5.1] Supertipo y Subtipo

Profesor(nombre,apt,email,rfc) → AP
Investigador(...,email,cédula,cuit) ← AI
Admin(nombre,apt,edad,depto,horas) ← AA

A = Atributos en común

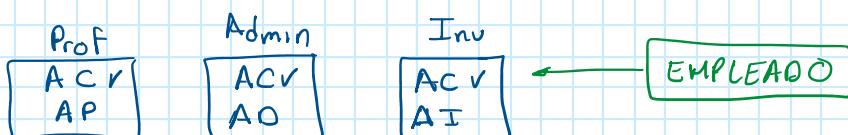
↳ Similar al concepto de herencia
en POO

- En BD existen conceptos de
- Supertipo $\hat{=}$ Contiene atributos en común
- Subtipo $\hat{=}$ Representa a un "Rol" del supertipo
- * Al menos un atributo en particular

NOTA! LO ANTERIOR HACE FORMAR
"JERARQUIA DE ENTIDADES"

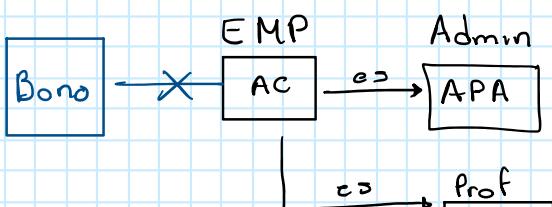
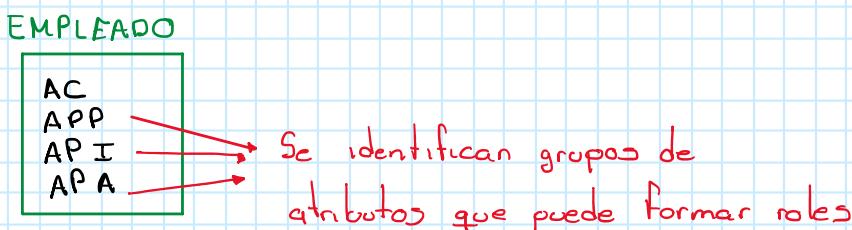
Proceso "Bottom Up"

- Existen los subtipos y se crea el supertipo
- Se parte de lo particular hacia lo general

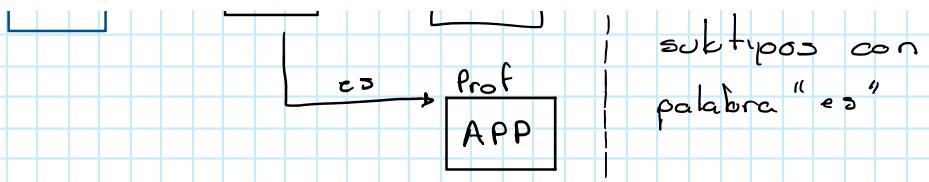


Proceso "Top-Down"

- Existe el supertipo y a partir de este se crea a los subtipos

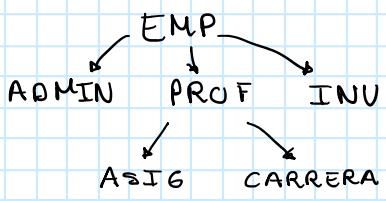


- La relación entre supertipos y subtipos con palabra "es"



5.3] Jerarquías de Especialización

- A nivel de jerarquía, puede existir jerarquías de supertipos y subtipos de cualquier cantidad de niveles.



- A mayor nivel hay menor rendimiento
- Definir 3 opciones de diseño

∴ Tenemos 3 opciones de diseño

○ Nivel alto de jerarquía

- Se desea crear n niveles en la jerarquía
- ✗ Rendimiento bajo debido a operaciones como join
- ✓ Reduce redundancia
- ✓ El diseño tiene una directa correspondencia con el diseño orientado a objetos del software que accede al BD

○ Tabla simple sin subtipos

EMPLEADO

AC
APP (null)
AP I (null)
AP A (null)

Dependiendo de que específicamente es se cambiaría el null por algo

- ✓ Sin problemas de desempeño
- ✗ No existe un mapeo con respecto al modelo POO del software
- ✗ Valores nulos → Puede que se esté forzando algo
 - ↳ Puede generar una inconsistencia
 - ↳ Debes poner todos los atributos

→ suele generar una inconsistencia
↳ Debes poner todos los atributos
del rol a ingresar.

NOTA: Útil cuando

- Supertipos con pocos "roles"
- Cada rol tiene pocos atributos particulares
- Existen y predominan atributos en común

③ Conjunto de tablas sin supertipo

PROFESOR	ADMIN	INV
AC APP	AC APA	AC API

- ✓ Sin posible penalización de desempeño
- ✗ Existe redundancia en atributos en común.

- Útil cuando tiene pocos atributos en común
- Una gran cantidad de subtipos
- Útil cuando cada registro cuenta con un sólo rol

5.4] Relaciones entre Supertipos y Subtipos

- Las relaciones son 1:1 de cada una de las entidades hijas respecto al padre
- La PK del supertipo se distribuye a cada uno de los subtipos
- Existen 4 variantes para relacionar un supertipo con subtipos

④ Restricción Excluyente/Traslape

- **Excluyente:** Cada instancia del supertipo se asocia (Disjunta) a lo más con una instancia de sus subtipos

(Disjoin) a lo más con una instancia de sus subtipos

- Traslape: La instancia del supertipo puede asociarse (Overlapping) con más de una instancia de sus subtipos

② Restricción Total/Parcial

- Total: Cada instancia del supertipo se asocia con al menos una instancia de sus subtipos
- Parcial: La instancia del supertipo puede o no asociarse con sus subtipos

→ Discriminante del Subtipo

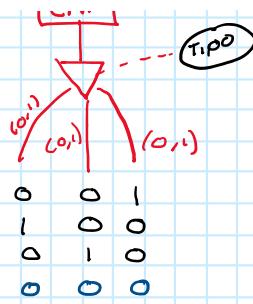
- Atributo o conjunto de atributos que permiten conocer al subtipo o subtipos con los que se relaciona cada instancia del supertipo.
- Evita realizar búsquedas en todos los subtipos para localizar un dato,

NOTACIONES

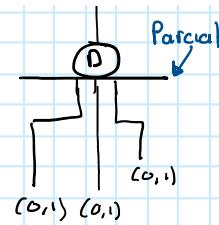
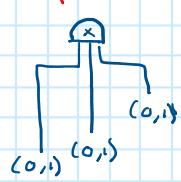
Tipo de relación	Reglas de Negocio	CHEN (D. Conceptual)	Crow's Foot (D. Lógico)	IDEIX (D. Lógico)
<u>Exclusión Total</u>	Un empleado debe contar con uno y sólo un rol.	<p>Diagrama CHEN de Exclusión Total:</p> <p>Relación EMP (Total) con Admin (0..1), Prof (0..1) y Inv (0..1). Un emp es de tipo Admin, Prof o Inv.</p> <p>* Discriminante se asocia al triángulo además de tener sólo la letra inicial</p> <pre> 0 0 1 1 0 0 0 1 0 </pre>	<p>Diagrama Crow's Foot de Exclusión Total:</p> <p>EMP (empleado_id PK, tipo char). Exclusión entre Admin (emp_id PK, tipo char), Prof (emp_id PK, tipo char) y Admin (emp_id PK, tipo char).</p>	<p>Diagrama IDEIX de Exclusión Total:</p> <p>Relación D (Total) con D1 (0..1), D2 (0..1) y D3 (0..1). Exclusión entre D1, D2 y D3.</p>
<u>Exclusión Parcial</u>	En algunos casos los empleados no cuentan con su único rol asignado.	<p>Diagrama CHEN de Exclusión Parcial:</p> <p>Relación EMP (Total) con TIPO (dashed line). Un emp tiene un tipo.</p>	<p>Diagrama Crow's Foot de Exclusión Parcial:</p> <p>EMP (empleado_id PK, tipo char). Exclusión entre TIPO (dashed line) y tipo char NULL.</p>	<p>Diagrama IDEIX de Exclusión Parcial:</p> <p>Relación D (Parcial) con D1 (0..1), D2 (0..1) y D3 (0..1). Exclusión entre D1, D2 y D3.</p>

Parcial

empleados no cuentan con su único rol asignado. Si cuentan con un rol, debe ser uno solo



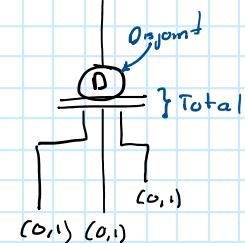
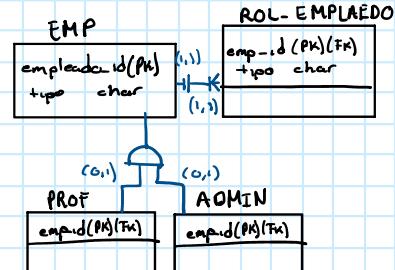
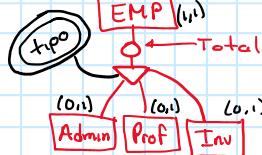
tipo char NULL



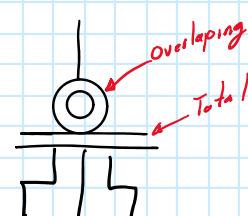
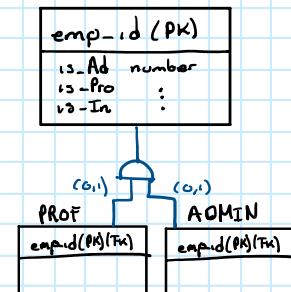
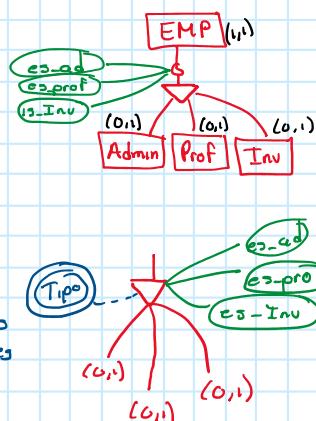
Traslape Total

Los empleados deben contar al menos un rol

* Discriminante se vuelve multivalorado.

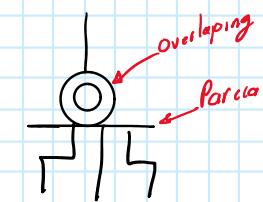
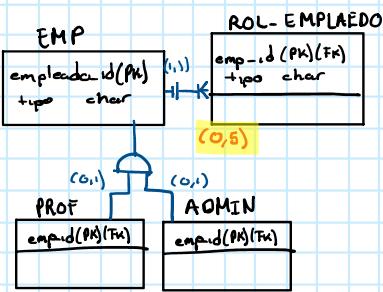
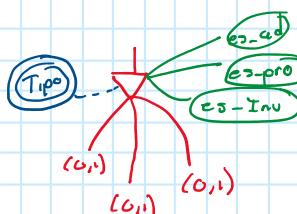


//Alternativa



Traslape Parcial

Los empleados pueden o no tener id asignado, otros tienen varios valores asignados.



* solo cambia la dependencia
* La versión alternativa queda igual

5.5 Modelado de catálogos y datos con histórico

Tuesday, 10 March 2020 8:05 AM

- En algunos casos se requiere almacenar cambios de valor de uno o más atributos de una tabla con respecto al tiempo
- Para implementar este requerimiento se emplea el concepto de **“Histórico”**

2. Diseño de una base de datos para una agencia de viajes.

Empleando el concepto de manejo y diseño de entidades e históricos realizar el diseño conceptual y lógico para el siguiente enunciado:

Una agencia de viajes aéreos desea llevar el control de los viajes que son adquiridos por sus clientes. Los datos del viaje son los siguientes:

- Lugar de salida (origen)
- Lugar de llegada (destino)
- Fecha y hora de salida

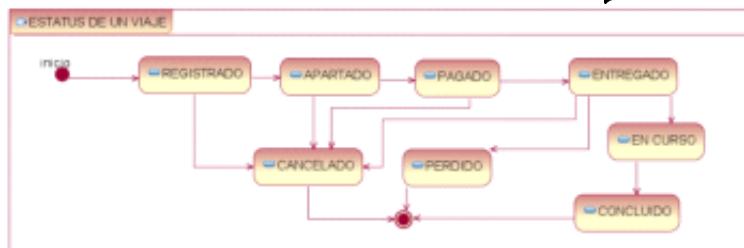
Cada destino y origen contiene una clave única de 3 caracteres, así como el nombre y descripción del lugar.

Ejemplo:

LUGAR_ID	CLAVE_LUGAR	DESCRIPCION
1	AGS	AQUASCALIENTES
2	LCB	LOS CABOS
3	CAN	CANCUN

DIAGRAMA DE ESTADOS

Adicional a esta información, para que un viaje se realice, debe pasar por las siguientes etapas:



Reglas de Negocio

se modelan como un catálogo

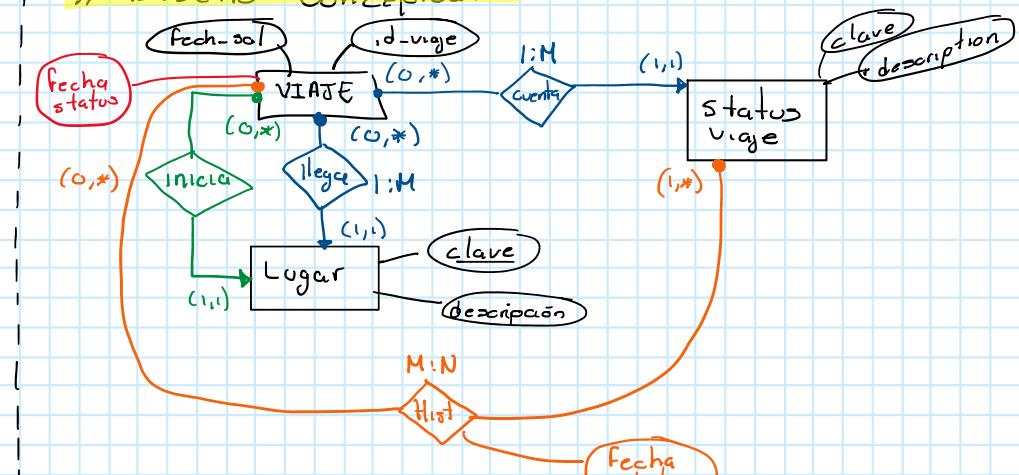
- REGISTRADO:** El cliente captura en el sistema los datos del viaje: origen, destino, nombre y apellidos de cada pasajero y la fecha de salida. El viaje se considera como REGISTRADO.
- APARTADO:** El sistema le da como tiempo máximo 1hr para que el cliente seleccione los números de asientos del avión para cada pasajero y para que el cliente capture los datos de su tarjeta de crédito. Mientras el cliente no proporcione los datos de su tarjeta, y no expire el tiempo antes mencionado, el viaje permanece como APARTADO.
- PAGADO:** Un viaje se considera PAGADO, cuando el cliente capture su número de tarjeta de crédito en el sistema, y es validado contra el banco correspondiente. Si el banco autoriza el pago, el viaje se considera como PAGADO. En caso contrario, se le notifica al cliente, y si el cliente no corrige el error, el viaje se considera como CANCELADO.
- ENTREGADO:** El cliente deberá acudir como máximo, un día antes a la agencia para recoger sus boletos. En este momento el viaje se considera como ENTREGADO. Si no acude el viaje se CANCELA.
- PERIODO:** Ocurre cuando el cliente ha recibido sus boletos, pero este no se presenta al aeropuerto a tomar su vuelo.
- EN CURSO:** El viaje toma este valor cuando se le notifica a la agencia que el cliente ha tomado el vuelo. Permanece en este estado hasta que el viaje termine.
- CONCLUIDO:** Un viaje se considera concluido cuando:
 - El pasajero ha descendido del avión en el lugar destino para viajes sencillos.
 - El pasajero ha descendido del avión en el lugar origen para viajes redondos.
- CANCELADO:** Finalmente, el viaje se puede cancelar por las siguientes causas:
 - El cliente no captura los datos de su tarjeta dentro del tiempo máximo permitido
 - El cliente no pudo comprobar su pago vía tarjeta de crédito.
 - La agencia cancela el viaje una vez que este ha sido ENTREGADO por alguna causa de fuerza mayor (fenómeno natural, fallas mecánicas, etc.)

La agencia requiere que se almacene en la base de datos, el estado actual que tiene cada viaje, así como la fecha en la que adquirió dicho estado.
Se requiere adicionalmente almacenar toda la secuencia de estados que ha tenido un viaje, desde su inicio hasta que llegue a su estado terminal.

// Lista de Entidades

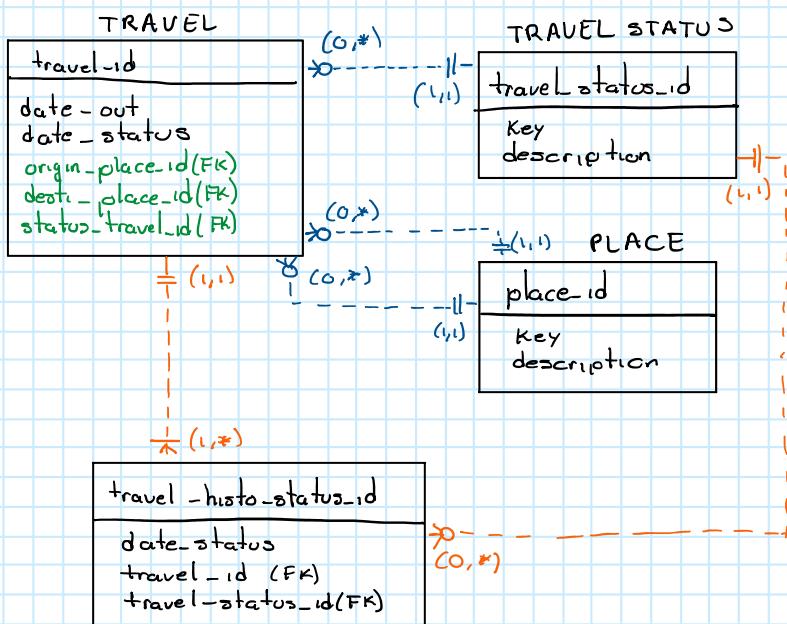
- Viaje
- Origen } lugar
- Destino }
- Tarjeta //No necesaria para este caso

// Diseño Conceptual





// DISEÑO LÓGICO



// Prueba Ejecutorio

VIAJE

v_id	s_id	f_sa	origen	dest
10	PAG	10-3-20-8:45	C.U.	XOCHI

HISTO

h_id	v_id	s_id	f_sa	date
1	10	REG		10-3-20-8:30
2	10	APA		10-3-20-8:40
3	10	PAG		10-3-20-8:45

6 Normalización

Thursday, 12 March 2020 7:16 AM

6.1] Proceso de Normalización

Normalización: Proceso empleado para corregir y evaluar la estructura de tablas con la finalidad de minimizar redundancias y la posibilidad de anomalías.

Este proceso se desarrolla aplicando una serie de pasos o estados llamados "Formas normales"

- Primera Forma Normal (1 FN) - Forma normal
- 2º Forma Normal (2 FN) Boyce-Codd
- 3º Forma Normal (3 FN) ← Mayormente usado
- 4º Forma Normal (4 FN)

- La aplicación de un menor o mayor nivel de normalización puede generar la siguiente situación:

Mayor nivel
(Normalización)

Menor nivel
(Normalización)

- | | |
|---|---|
| ✓ Menor o prácticamente ausencia de redundancia | ✓ Sin problemas de desempeño |
| ✗ Posible impacto en desempeño | ✗ Mayor probabilidad de anomalías y redundancia |

// Nivel utilizado típicamente es el 3

→ A mayor nivel, mayor número de tablas y ∴ mayor número de operaciones "Join"

→ No siempre un nivel alto de normalización es lo mejor

// Ejemplo

1.1. EJEMPLO PARA NORMALIZAR EN 1FN, 2FN Y 3FN.

Para realizar el proceso de normalización, considerar la siguiente tabla de datos en la que se almacenan las faltas y las calificaciones de los alumnos de una universidad, en donde cada alumno pertenece a una carrera la cual también se especifica en la tabla de datos.

num_estudiante	nombre	apellidos	clave_institucion	clave_asignatura	calificacion	faltas	clave_carrera	clave_nivel	clave_carrera	clave_nivel	
1001	Juan	Méndez	Kris	3783	Algebra	10	1	9	CDL	Colima	110
1002	Mario	Luna	Ulaldo	3431	Calculo 2	8	0	7	CDL	Colima	110
1003	Eva	Aguirre	Salas	3431	Calculo 1	10	3	7	CHH	Chihuahua	110
1004	Laura	Julieta	Altamira	1763	Calculo 2	8	5	8	NL	Nuevo Leon	111
1005	Alonso	Lugo	López	1890	Calculo 1	10	2	5	SON	Sonora	111
1006	Maria	Luna	Ulaldo	1763	Algebra	10	2	8	CHH	Chihuahua	110
1006	Eva	Lugo	Macias	1790	Estadistica	8	9	6	QRO	Queretaro	111

• ¿Qué anomalías existen en esta tabla de datos que podrían ser eliminadas al aplicar un proceso de normalización?

- Empleando los conceptos de Normalización, aplicar el proceso hasta obtener la 3FN.

→ Existe(n) atributo(s) con valor(es) nulo(s) (NULL)

→ Anomalía de Inserción

TIPOS DE ANOMALIAS

- | | |
|--------------------------|---|
| • Anomalías de inserción | Forzar inserción |
| • Actualización | Atributos nulos |
| • Eliminación | Modificar copias parciales de los datos
↳ Genera inconsistencias |
| • Se pierden datos | Alumno 1006
Perdemos datos de asignaturas y de cotizados |

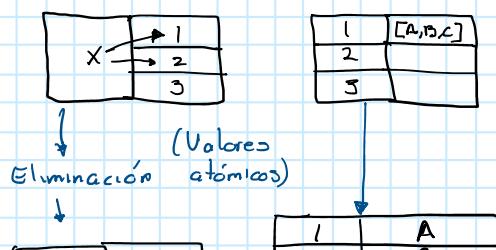
6.2 Aplicación 1FN

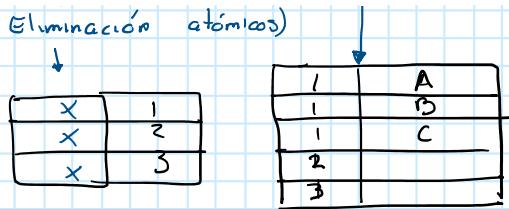
- Una tabla estará en 1FN cuando:

- ① Se han eliminado grupos de repetición
- ② Se han identificado la PK, dependencia parciales y dependencias transitivas.

NOTA: En este nivel no se ha dividido o no generan tablas nuevas

// Grupo de repetición





- 2) Para identificar la PK y las dependencias se emplea el concepto de **dependencia funcional**

$$\underbrace{A \rightarrow B}_{\begin{array}{l} \text{Atributo(s)} \\ \text{Determinante(s)} \\ (\text{PK}) \end{array}} \quad \begin{array}{l} \text{Atributos} \\ \text{Dependientes} \end{array}$$

Tabla de estudiantes

num_estudiante	nombre	ap_paterno	ap_materno	clave_asignatura	nombre_asignatura	créditos_asignatura	faltas	calificación	clave_nacimiento	Lugar_nacimiento	clave_carrera	nombre_carrera
3001	Juan	Méndez	Kim	1768	Algebra	10	1	9	COL	Colima	110	I. Civil
3002	Mario	Luna	Ulaldo	1890	Calculo 2	8	0	7	COL	Colima	110	I. Civil
3003	Eva	Aguirre	Silas	3411	Calculo 1	10	3	7	CHH	Chihuahua	110	I. Civil
3004	Lidia	Jáñez	Aldama	1763	Algebra	10	0	10	NL	Nuevo León	111	I. Electro
3005	Alonso	Lugo	López	1890	Calculo 1	10	2	5	MICH	Michoacán	111	I. Electro
3002	Mario	Luna	Ulaldo	1763	Algebra	10	2	8	SON	Sonora	111	I. Electro
3006	Eva	Lugo	Macías	1790	Estadística	8	9	6	QRO	Chihuahua	110	I. Civil
										Querétaro	111	I. Electro

→ No podríamos saber (Necesitan de asignatura)

// Con el num_estu puedo ver la clave de nacimiento y por lo tanto lugar nacimientos

∴ La PK de la tabla está formado por:

- ① num_estudiante
- ② Clave_asignatura

No incorporar PK artificiales

Dependencia Parcial

- Existen en tablas con llaves primarias (PK) compuestas
- En este tipo de dependencias, cada uno de los atributos que forman a la PK pueden determinar por si sólo a al menos un atributo

↳ Es decir, que pueden existir N de dependencias parciales tal que N es el n.º de columnas que forman la PK

- ① La tabla tiene PK compuesta ✓
- ∴ Dependencias parciales = 2
debido a que es 1 por cada elemento de la (PK)

DP1

num_estudiante → nombre, ap-mat, ap-pat
 clave_carr, nombre_carr-
 clavenac, lugar_naci

DP2

clave_asignatura → nombre_asignatura, créditos

→ faltas y calificación dependen de DP1 y DP2

► Dependencia Transitiva

- Existe al detectar una dependencia funcional ($A \rightarrow B$) en la que un atributo determinante NO forma parte de la PK
- Su existencia no depende del tipo de (PK)

DT1

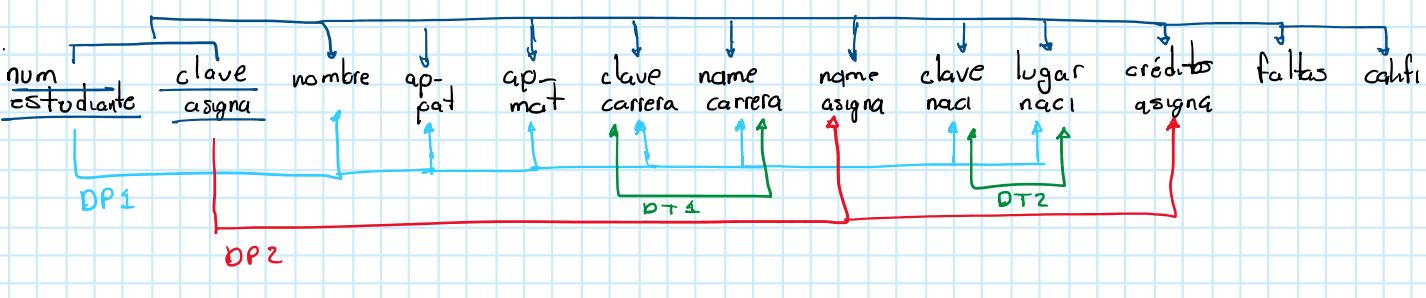
clave_nacimiento → lugar_nacimiento

DT2

clave_carrera → nombre_carrera

// Todo lo anterior se puede expresar en un

DIAGRAMA DE DEPENDENCIAS



6.2] Aplicación de la 2FN

- La tabla estará en 2 FN cuando:

- La tabla está en 1FN
 - Se han eliminado dependencias parciales

¿Qué sucede si la tabla no

Tiene dependencias parciales? → Ya está en 2FN

$\hookrightarrow S_1$ la PK \Rightarrow simple

II Proceso para eliminar una dependencia

- 1) Para cada dependencia se creará una tabla una nueva tabla
 - Su PK será el atributo determinante
 - Sus atributos corresponden con los atributos dependientes
 - 2) De la tabla original se eliminan a los atributos dependientes y se conserva el atributo determinante.

2 FN

$T_1 : \text{num_estudiante} \rightarrow \{\text{nombre, apt-mat, apt-pat}\}$

(PK)

$\left. \begin{array}{l} \text{clave_carrera, nombre_carrera} \\ \text{clave_naci, lugar_naci} \end{array} \right\} ESTUDIANTE$

T.O. num_estudiante, clave_asig → faltas,
(PK)(FK) (PK)(FK) calificación

NOTA: Siempre serán los que dependen de ambas

$$y = d_1 d_2 \dots d_n$$

• La no se elimina dato de asignatura

→ Todavía tenemos eliminación de datos

en el estudiante y lugar nacimiento

6.4] Aplicación de la 3FN

→ No hay artificiales,
sólo naturales (Simples,
Compuestas)

- Está en 2.FN

Compuestas)

- Est醤 en 2FN
 - Se han eliminado dependencias transitivas, por cada transitiva, se crea tabla nueva, se sigue el mismo proceso de eliminaci髇

T3: clave_carrera → nombre_carrera } CARRERA
(PK)

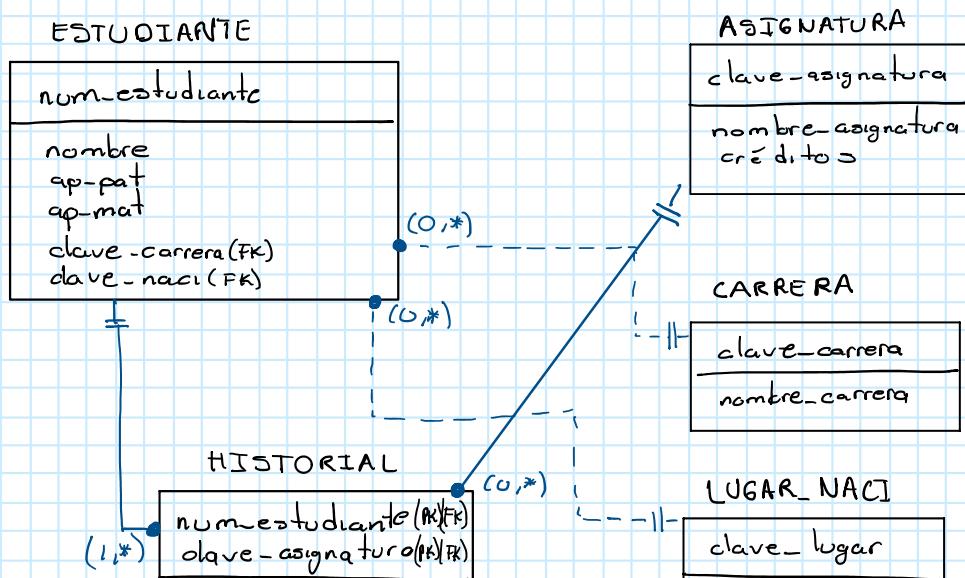
T4: clave_nacimiento → lugar_nacimiento } LUGAR
NACIMIENTO

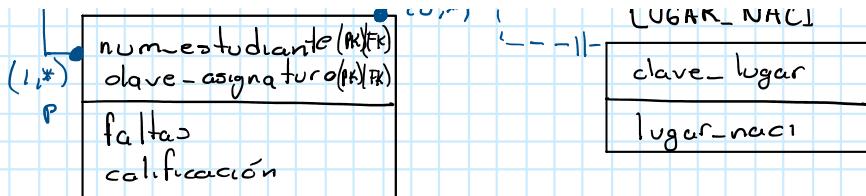
T₁: num-estudiante → nombre, apt-mat, apt-pat }
 (PK) { ESTUDIANTE
clave_carrera, nombrecarrer
clave_naci, lugarnaci
 Son determinantes (se quedan)
 ↳ Se vuelven (FK)
 Sino como rediriges // No sufren cambios

T.O. num_estudiante, clave_asig → faltas, calif_carrera } HISTORIAL
(PK) (FK) (PK) (FK)

► Construcción Modelo Relacional

- Debe corresponder con la definición de tablas obtenidas en el proceso de normalización





// Refinamiento

- 1] Homologar a PKs artificiales
- 2] Renombrar columnas

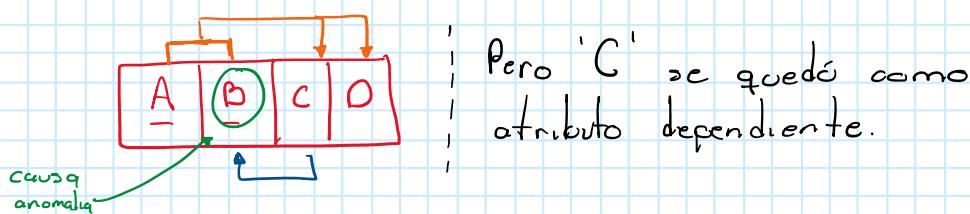
asignatura_id clave nombre creditos
--

\uparrow
No hay atributos determinantes con dependientes $\therefore 3NF$

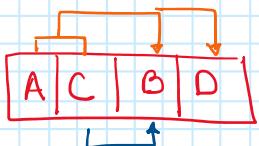
6.5] Formas Normales de Orden Superior

► Forma Normal Boyce-Codd (BCNF)

- Es una variante de la 3NF que resuelve anomalías no soportadas
- Ocurre en tablas con PK compuesta en donde existen atributos dependientes que pudieran haber sido seleccionado como determinantes y por lo tanto, forma parte de la PK, pero, por alguna razón no fue seleccionado.



- Observar que 'C' determina a los valores de B esto implica que 'C' pudiera ocupar en lugar de B
- \therefore La tabla puede reescribirse de la siguiente forma

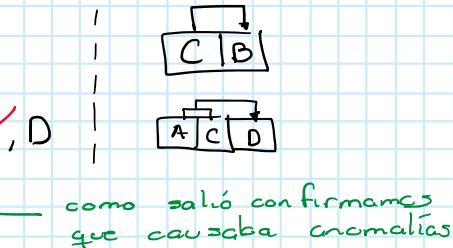


Dependencia Parcial 1 Podemos eliminarla

Al formar una D Parcial ésta puede eliminar:

$T_1: C \rightarrow D$

$T_0: A, C \rightarrow B, D$



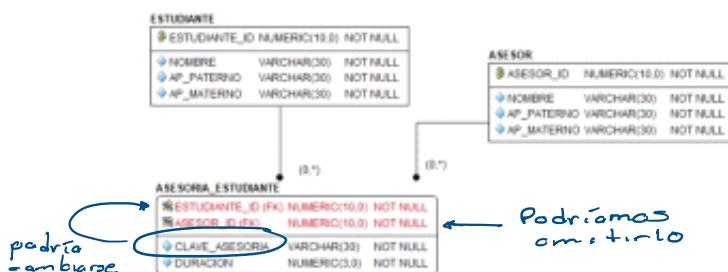
// Ejemplo

1.2. EJEMPLO PARA APLICAR BNF

Registro de las asesorías y de los profesores asignados a un alumno.

- Un estudiante puede tener a varios asesores asignados durante un semestre y un asesor puede tener asignados a varios alumnos.
- Un asesor puede dar varias asesorías. Cada asesoría se identifica por una clave. Una asesoría la ofrece un solo asesor.
- El estudiante puede solicitar asesoría por parte de uno de sus asesores y al acudir se guarda el tiempo que dura dicha asesoría.

El modelo relacional que se ha generado para implementar las reglas de negocio es el siguiente:



La siguiente tabla muestra el comportamiento de los datos para la tabla ASESORIA_ESTUDIANTE

ESTUDIANTE_ID	ASESOR_ID	CLAVE_ASESORIA	DURACION (MIN)
1	1	ALG-001	15
2	1	ALG-001	22
3	2	BD-022	19
4	3	MATH-089	34
4	4	CALC-020	32
5	4	CALC-020	27
6	5	JAVA-093	21

si eliminamos estudiante 6 se pierde relación ASESOR_ID 5 tiene Java-093

[ANOMALÍA DE ELIMINACIÓN]

Comúnmente en tabla intermedia y además M:N

est_id, asesor_id $\xrightarrow{\text{dP}}$ clave-asesoria, duración

clave-asesoria \rightarrow asesor_id ✓ si es válido

// Por lo tanto estructuramos como

est_id, clave-asesoria \rightarrow asesor_id, duración

TIP

chechar M:N
y si hay
muchos atribu
tos en Tab
intermedia

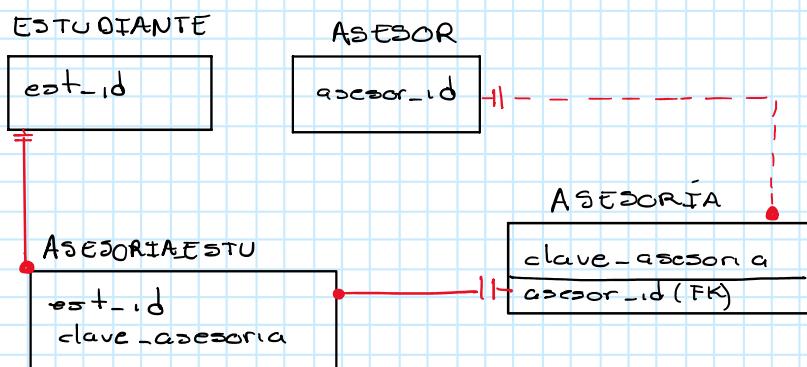
II Dependencias

DP 1 clave-asesoria → asesor_id

// Eliminando DP 1

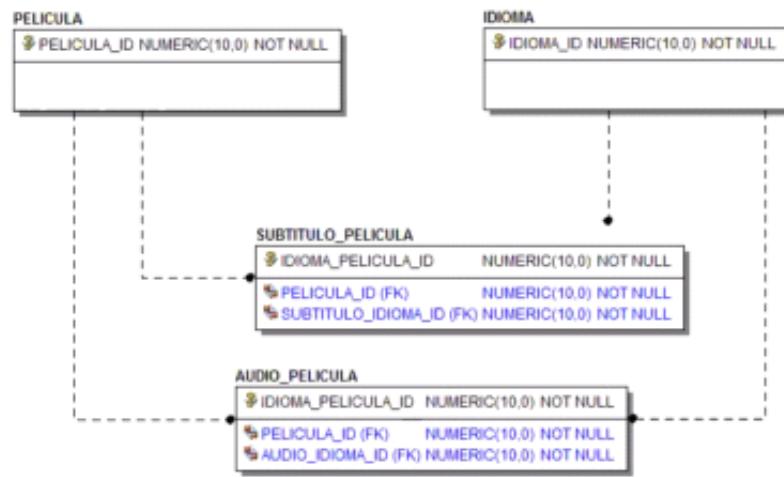
T₁: clave_asesoria → asesor_id } ASESORÍA

T_0 : $\text{est_id}, \text{clave_asesoria} \rightarrow \text{duración}$ } Asesoría
 $(PK)(FK)$ (PK) } Estudiante



6.6] 4 Forma Norma]

- Es sobre todo con malos diseños, es el caso de llaves foráneas de más



NOTA! No podría ser una jerarquía debido a que no comparten atributos similares (No los heredados)

Serie 5

Tuesday, 24 March 2020 6:01 AM

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
FACULTAD DE INGENIERIA
BASES DE DATOS

TEMA 5.
MODELADO DE DATOS EXTENDIDO

NOMBRE: Murrieta Villegas Alfonso
FECHA DE ENTREGA: 27 Marzo 2020

GRUPO: 3

PUNTAJE (MAX 32):

33/33

- Todos los ejercicios de este documento deberán entregarse por separado y de forma individual, formato libre.
- Existe un archivo en la carpeta compartida del tema 5 llamado `ejercicios-tema5-opcionales.pdf`
- Este archivo contiene un resumen del tema, así como estrategias adicionales que pueden ser empleadas en diseños complejos de bases de datos, así como 3 ejercicios adicionales. Si se entregan, se tendrán puntos extras. Entregar por separado en caso de realizarlos (En un documento los ejercicios 1 y 2, y en otro documento los ejercicios opcionales). Se recomienda ampliamente al menos leer el documento para conocer estas 3 estrategias.
- De forma adicional, existe una subcarpeta dentro de la carpeta correspondiente al tema 5 llamada `clase-virtual-modelado`. Esta carpeta contiene material adicional en el que se desarrolla el modelo conceptual y lógico de un caso de estudio complejo. El diseño se hace a través de una `clase virtual` la cual fue grabada y se puede acceder a ella. Revisar el archivo `README.md` de la carpeta para obtener acceso a dicha clase.

1.1. Ejercicio 1.

Considerar el caso de estudio "Diseño de una base de datos para una aseguradora", del cual se anexa su respuesta. Al final se presenta una serie de cambios que se han aplicado a las reglas de negocio. Genere una nueva versión tanto del diseño conceptual como del lógico para implementar las reglas nuevas.

Diseño de una base de datos para una aseguradora.

Una compañía requiere almacenar la información de sus empleados y sus dependientes asegurados. Cada empleado tiene un número de empleado, nombre, fecha de contratación y título. En caso de que un empleado se le haya asignado el rol de inspector, se almacena la fecha en la que se certificó como inspector, la clave de la certificación (cadena alfanumérica de 10 caracteres) y su calificación.

Un empleado puede registrar hasta 5 dependientes económicos. Para cada uno de ellos se almacena el número de seguridad social (alfanumérico de 18 posiciones), el nombre y apellidos. Cada dependiente debe ser asociado con un solo empleado. A cada dependiente, se le asigna un folio consecutivo iniciando en 1 y se reinicia por cada empleado.

Otros empleados son catalogados como gerentes. Para ellos se requiere registrar la fecha en la que se iniciaron como gerentes. La empresa cuenta con un catálogo de agencias en las que se almacena el nombre, número de sucursal (4 dígitos), y su dirección (no se requiere desglosar). Al registrar al gerente, se le asocia la gerencia que va a administrar. Cabe mencionar que un gerente no puede ser inspector al mismo tiempo y tampoco puede administrar más de una agencia.

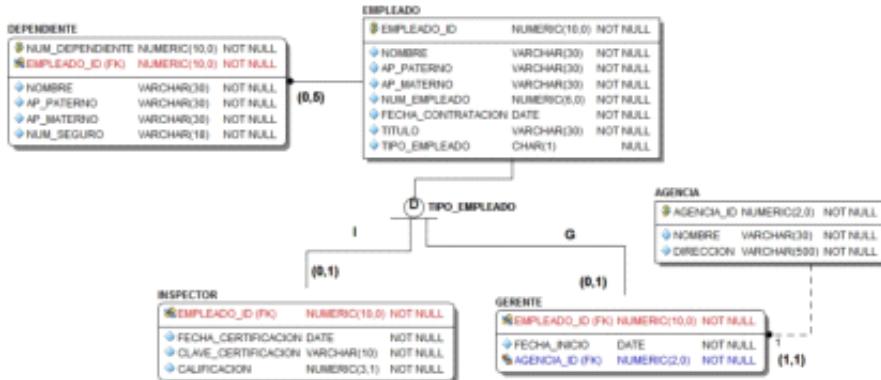
Respuesta:

- En este caso se trata de un supertipo `EMPLEADO` con sus subtipos `INSPECTOR` y `GERENTE`. Los subtipos son válidos, ya que existen atributos en particular para ambas tablas.
- Las restricciones del supertipo con sus subtipos son:
 - Restricción parcial, ya que puede haber empleados que no tengan el rol de `INSPECTOR` ni el de `GERENTE`.
 - Restricción excluyente, ya que un empleado no puede tener ambos roles a la vez.
 - El discriminante está representado por el campo `TIPO_EMPLEADO`, y debe ser definido como `NULL` por ser restricción parcial.
 - Observar que se requiere de la entidad `AGENCIA`, la cual representa el catálogo de agencias y esta se asocia con `GERENTE`. Sería incorrecto asociarla con `EMPLEADO` o `INSPECTOR` ya que solo tiene sentido para los gerentes.
- Se omiten escribir las cardinalidades con valor (1,1)
- Notar que la entidad `DEPENDIENTE` es débil. Se emplea el concepto de dependencia de identificación ya que su clave principal es un folio el cual se reinicia por cada dependiente del empleado: El dependiente 1 del empleado 100, el dependiente 2 del empleado 100, el dependiente 1 del empleado 101, etc.

Diseño conceptual:



Diseño lógico:

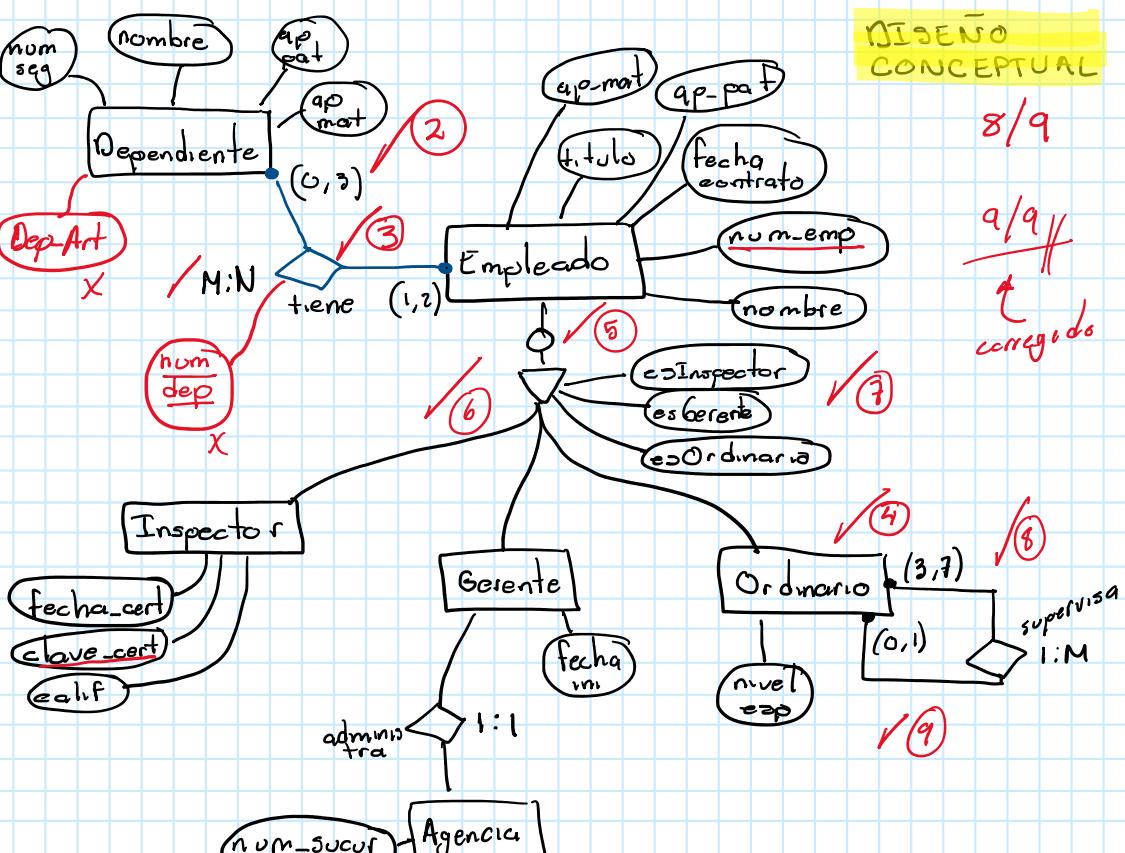


Cambios solicitados:

- Ahora un dependiente puede tener asociados a más de un empleado (máximo 2). Por ejemplo, un menor de edad puede tener asociado a su papá y a su mamá, los cuales son empleados. A un empleado se le permite a lo más, el registro de 3 dependientes.
- Se decide que cada empleado debe tener asignado al menos un rol de forma obligatoria. Se agrega un nuevo tipo de empleado llamado ordinario, el cual es asignado a todos los empleados que no cuentan con alguno de los otros 2 roles disponibles. Para este tipo de empleados, se almacenará su nivel de especialidad (1, 2, o 3). Adicionalmente, algunos empleados ordinarios son jefes de equipo y tienen asignados a sus integrantes, los cuales también son empleados ordinarios. Cuando se registran los datos del empleado ordinario, se le debe asignar (asociar) a su jefe de equipo. Los equipos deben estar formados por mínimo 3 empleados, máximo 7.
- Ahora la empresa permite un empleado puede tener más de un rol a la vez. Por ejemplo, un inspector, puede ser también gerente.

se elimina arco

18P

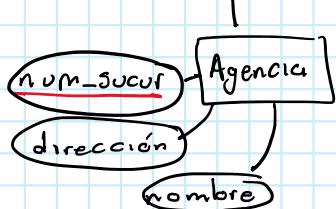


DISEÑO CONCEPTUAL

8/9

9/9 //
corregido

✓ 8
✓ 9



DISEÑO LÓGICO

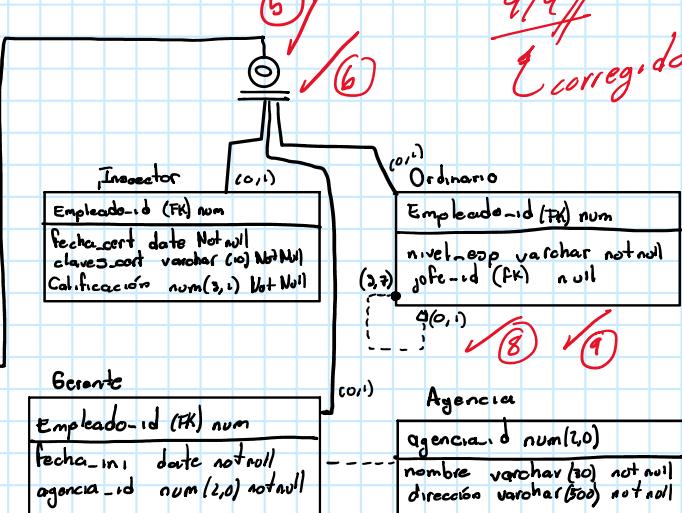
8/9

DEPENDIENTE	CREAR OTRO (PK)
nom_dependiente num(0,0) (PK)	
nombre varchar(30) not null	
ap-patr varchar(30) not null	
ap-matr varchar(30) not null	
nom_segundo varchar(30) not null	

Empleado_o_depen	(0..1)
nom_dependiente (PK) (FK)	
empleado_id (PK) (FK)	

EMPLEADO

empleado_id num(10,0) (PK)
nombre varchar(30) not null
ap-patr varchar(30) not null
ap-matr varchar(30) not null
nom empleado num(8,0) not null
fecha_nacim date not null
titulo varchar(30) not null
es_inspec bool not null
es_geren bool not null
es_ordi bool not null



Diseño de una base de datos para registrar solicitudes de VISA.

La empresa encargada de administrar las solicitudes de VISA para visitar un país decide diseñar una nueva base de datos empleando las siguientes reglas de negocio:

- Cuando un ciudadano decide solicitar una VISA, se crea una nueva solicitud con los siguientes datos: nombre y apellidos del solicitante, CURP, país de nacimiento, país a visitar y ocupación.

Ing. Jorge A. Rodríguez Campos

jorgerdc@gmail.com

Página 2

Material de apoyo.

FI-UNAM

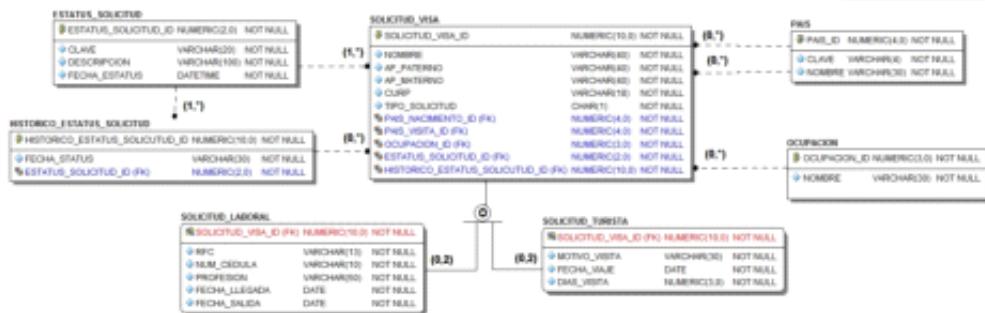
- Para el país de nacimiento y para la ocupación la empresa cuenta con catálogos:
- Ejemplo de los datos del catálogo de países:
 - Clave del país: MX
 - Nombre del país: MEXICO.
- Ejemplo de los datos del catálogo de Ocupaciones:
 - id de la ocupación: 1
 - Nombre de la ocupación: PROFESIONISTA.
- Existen 2 modalidades de VISA:
 - VISA turística: Se requiere capturar adicionalmente motivo de la visita (campo abierto), fecha en la que se planea viajar, y número de días aproximado de estancia en el país.
 - VISA laboral: Se requiere capturar adicionalmente, el RFC del solicitante, su número de cédula, el título profesional o profesión del solicitante, y el período de tiempo planeado que se desea estar en el país por motivos laborales.
- Cabe mencionar que un ciudadano si así lo desea puede solicitar ambas modalidades, ya que se han tenido casos que al término del periodo de la VISA laboral, el solicitante permanece en el país pero con la modalidad de turista. En estos casos, se registran ambos tipos asociados a una sola solicitud.
- Para llevar el seguimiento del procesamiento de una solicitud, se han diseñado una serie de estados por los cuales pasa la solicitud en el siguiente orden: REGISTRADA (R), EN REVISIÓN (V) APROBADA (A), RECHAZADA(D) en caso de que la VISA no sea otorgada. Se requiere llevar el control de cambios de este status a lo largo del tiempo que dure el proceso de revisión y aprobación de la VISA.

Diagrama ER con errores:





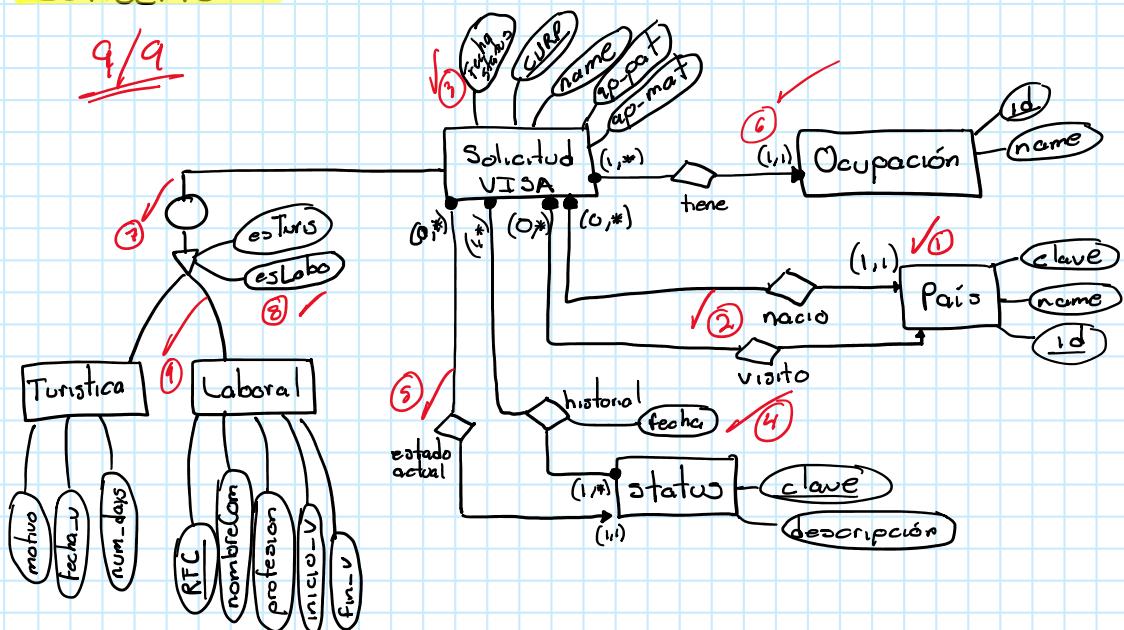
Modelo relacional con errores:



9P, 1P por cambio

DISEÑO CÓNCEPTUAL

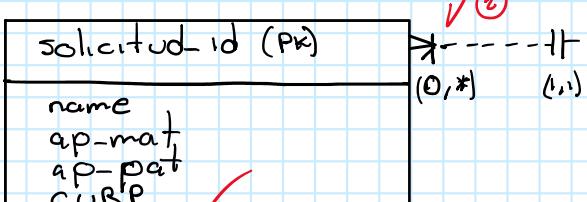
9/9



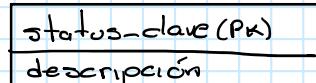
DISEÑO CONCEPTUAL

6/6

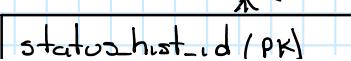
Solicitud_Visa

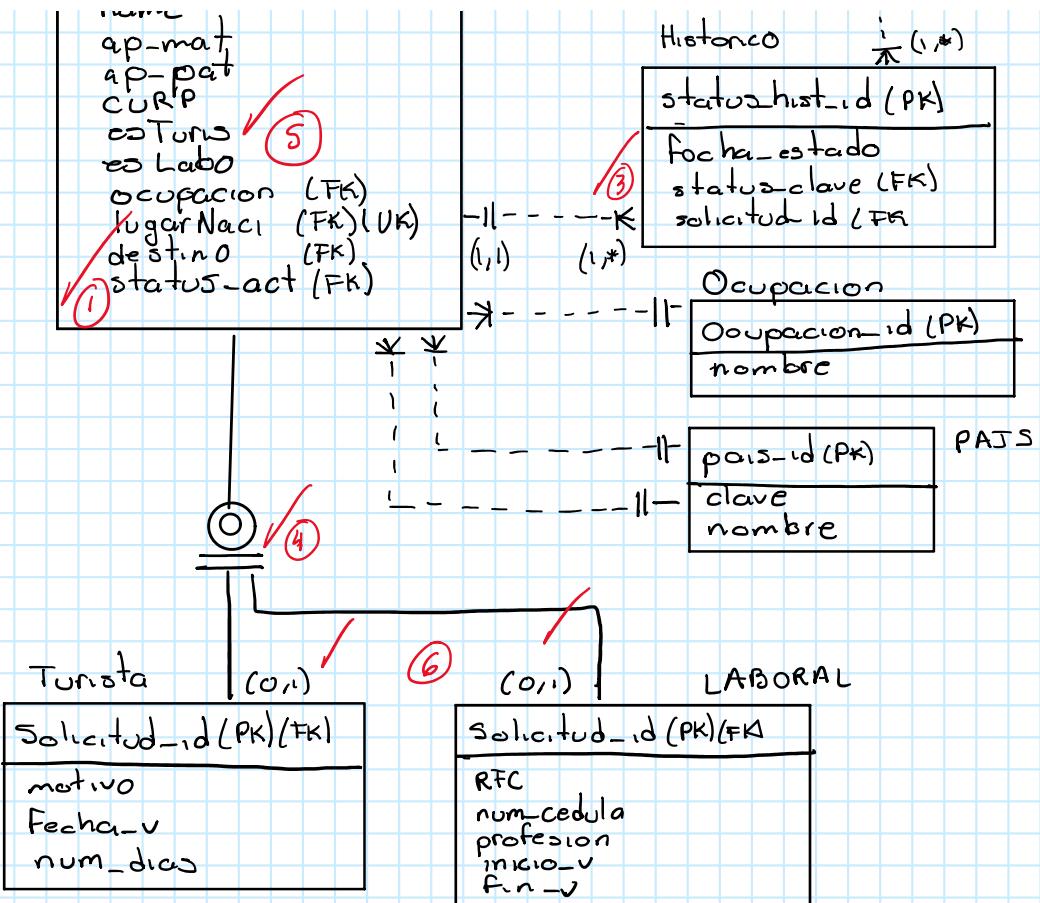


Status



Historico





Serie 6

Friday, 27 March 2020 5:57 AM

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
FACULTAD DE INGENIERIA
BASES DE DATOS

TEMA 6.
NORMALIZACIÓN

NOMBRE: Murrieta Villegas Alfonso
FECHA DE ENTREGA: 27 Marzo

GRUPO: 3

PUNTAJE (MAX 60):

60/60

Todos los ejercicios de este documento deberán entregarse de forma individual.

1.1. EJERCICIO 1.

Consideré el siguiente modelo relacional empleado para almacenar los cursos que imparte un instituto. Cada curso se imparte en varias clases de una hasta 5 clases a la semana.



El instituto comienza sus operaciones a las 16:00, cada clase independientemente del curso siempre tiene una duración de 2 hrs., por lo que sus cursos se imparten a las 16:00, a las 18:00 y a las 20:00. Para un mismo curso puede darse el caso que sus N clases se imparten en distintos horarios. A pesar de que todas las clases duran 2 horas, el usuario desea que el campo FECHA_FIN se conserve.

La siguiente tabla muestra un extracto de los datos almacenados:

CURSO

CURSO_ID	NOMBRE	FECHA_INICIO	FECHA_FIN
1	COCINA	01/01/2011	01/03/2011
2	CARPINTERIA	01/01/2011	01/03/2011
3	ELECTRONICA	01/01/2011	01/03/2011
4	MUSICA	01/01/2011	01/03/2011

CLASE_CURSO

CLASE_CURSO_ID	DIA	HORA_INICIO	HORA_FIN	SALON	CURSO_ID
1	LUNES	18:00	20:00	B-14	1
2	MIERCOLES	18:00	20:00	b-16	1
3	MARTES	16:00	18:00	B-20	2
4	JUEVES	16:00	18:00	B-21	2
5	LUNE	18:00	20:00	B17	3
6	miércoles	18:00	20:00	B-17	3
7	Martes	16:00	18:00	B-25	4
8	JUEVES	16:00	18:00	b-25	4
9	VERNES	20:00	22:00	B-21	4

El DBA ha detectado cierta redundancia y ligeras inconsistencias en la tabla CLASE_CURSO, en los campos DIA, HORA_INICIO y HORA_FIN y SALON, por lo que le ha solicitado al diseñador solucionar este problema. El diagrama resultante debe estar libre de redundancias innecesarias considerando las reglas de negocio antes mencionadas.

A. Determine en qué forma normal se encuentra la tabla.

2P

B. Aplicando los conceptos de normalización reescribir el modelo relacional anterior que permita la eliminación de redundancias innecesarias.

4P

C. Reescribir la tabla de datos en las nuevas tablas para confirmar la eliminación de la redundancia e inconsistencias.

1.A]

2/2

clase-curso_id	dia	hora_inic	hora_fin	salon	curso_id
1	lunes	18:00	20:00	B-14	1

clase-curso_id	dia_id	dia	hora_id	hora_inic	hora_fin	salon_id	salon	curso_id
1	1	lunes	1	18:00	20:00	1	B-14	1

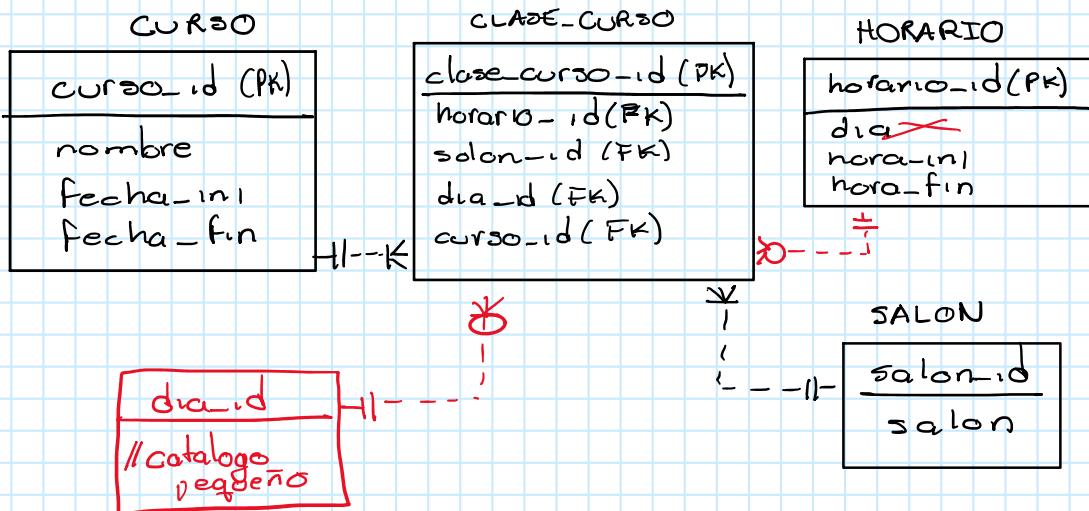
TABLA 1: $\text{clase-curso_id} \rightarrow \text{dia_id}, \text{horario_id}, \text{salon_id}, \text{curso_id}$

TABLA 1: $\text{dia_id} \rightarrow \text{dia}$ TABLA 1: $\text{horario_id} \rightarrow \text{hora_inic}, \text{hora_fin}$

TABLA 2 dia_id → dia TABLA 3 horario_id → hora_ini, hora_fin

TABLA 4 salon_id → salon

B] Modelo



4/4

C) Tabla de Registros

3/4

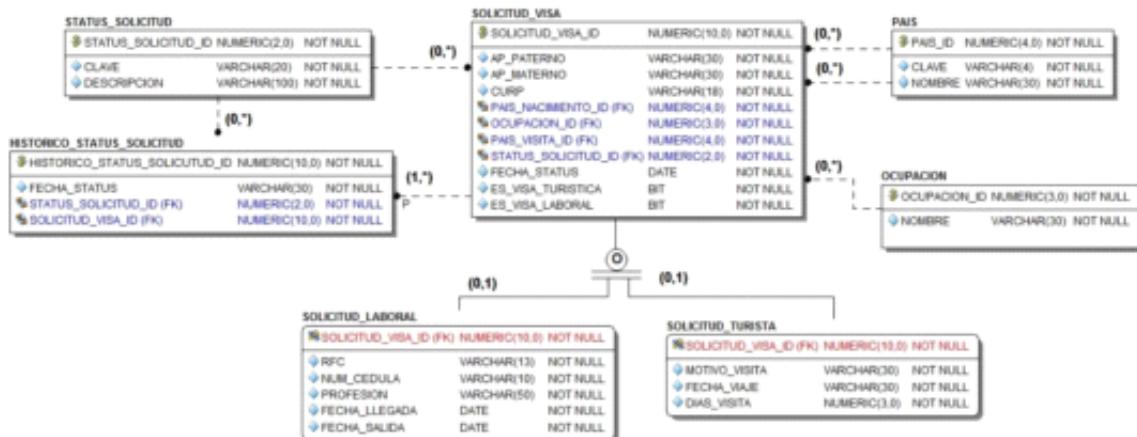
Horario_id	dia	Hora_ini	Hora_fin
1	LUNES	18	20
2	MIERCOLES	18	20
3	MARTES	16	18
4	JUEVES	16	18
5	VIERNES	20	22

Clase_id	salon	Horario_id
1	B-14	1
2	B-16	2
3	B-20	3
4	B-21	4
5	B-17	1
6	B-17	2
7	B-25	3
8	B-25	4
9	B-21	5

Clase_curso_id	Clase_id	curso_id
1	1	1
2	2	1
3	3	2
4	4	2
5	5	3
6	6	3
7	7	4
8	8	4
9	9	4

EJERCICIO 2

Considere el siguiente modelo relacional que muestra el diseño de una base de datos que guarda la información de las solicitudes de VISA para viajar a diversos países.



El DBA ha detectado problemas de desempeño por lo que le ha solicitado al diseñador las siguientes acciones:

- Determine el nivel de normalización para cada una de las tablas del diagrama. 2P
- Se requiere bajar un nivel de normalización para la tabla SOLICITUD_VISA considerando para ello las tablas STATUS_SOLICITUD, PAÍS y OCUPACIÓN. Reescribir la tabla SOLICITUD_VISA con los cambios necesarios. 4P
- Se requiere que al consultar los datos de una VISA se emplee una sola sentencia SQL sin tener que asociar con las tablas SUBTIPOS. Reescribir la tabla SOLICITUD_VISA con los cambios necesarios. 4P

A]

Solicitud_VISA → Sin DT ∴ 3FN

2/2

Solicitud_Laboral → 3FN

Solicitud_Turista → 3FN

Ocupación / País / Histórico / Status → 3FN

B]

Tabla Solicitud_VISA

4/4

Solicitud_visa_id (PK) numeric (10,0) nn

nn = Not Null

ap-pat	varchar2(30)	nn
ap-mat	varchar2(30)	nn
CURP	varchar2(18)	nn
país-naci_id (FK)	numeric(8,0)	nn
nombre-país-nac	varchar(30)	nn
país-visita_id (FK)	numeric(8,0)	nn
nombre_pais-visi	varchar2(30)	nn
ocupación_id (FK)	numeric(3,0)	nn
ocupacion-nombre	varchar2(30)	nn
status-solicitud_id (PK)	num(2,0)	nn
descripción-status	varchar2(100)	nn
fecha-status	date	nn
esVisa	bit	nn
esTurista	bit	nn

c) Reescribir tabla

4/4

solicitud_visita_id (PK)	numeric (10,0)	nn
ap-pat	varchar2(30)	nn
ap-mat	varchar2(30)	nn
CURP	varchar2(18)	nn
pais_naci_id (FK)	numeric (9,0)	nn
pais_visita_id (FK)	numeric (8,0)	nn
ocupacion_id (FK)	numeric (3,0)	nn
status_solicitud_id (FK)	num(2,0)	nn
fecha_status	date	nn
estTurista	bit	nn
motivo_visita	varchar2(30)	nn
fecha_viaje	varchar2(30)	nn
dias_visita	num(3,0)	nn
esLaboral	bit	nn
RFC	varchar2(30)	nn
num_Cedula	varchar2(10)	nn
profesion	varchar2(50)	nn
fecha_llegada	date	nn
fecha_salida	date	nn

► Ejercicio 3

1.3. EJERCICIO 3:

La siguiente tabla muestra los datos almacenados de un sistema web que ofrece renta de películas a sus clientes por internet (Streaming). Considerar las siguientes reglas de negocio:

- Un cliente cuenta con una sola membresía. Al terminar el periodo, si el cliente decide continuar o renovarla, se modifica la fecha de vigencia.
- La duración de las membresías es variable. Algunos clientes tienen membresías por 2 años, por 1 año, etc.
- La duración de la renta de la película también es variable. Cuando la fecha de entrega expira, ya no se permite realizar el proceso de Streaming.
- Si un cliente decide rentar la misma película varias veces, solo se actualiza la fecha de la renta y la fecha de entrega.

MEMBRESIA	NOMBRE CLIENTE	APELLIDO PATERNO	MIEMBRO DESDE	MIEMBRO HASTA	PELICULA	FECHA DE ENTREGA	TIPO PELICULA	PRECIO RENTA	FECHA DE RENTA	DESCRIPCION TIPO
8902493	Ramiro	Martinez	10/05/2000	10/04/2002	HALLOWEEN 2	14/05/2000	C	56.50	10/05/2000	Adolescentes y adultos
8902493	Ramiro	Martinez	10/05/2000	10/04/2002	BIUTIFUL	13/05/2000	B	100.0	10/05/2000	Familiar
7823493	Mariana	Juárez	20/04/1998	20/05/1998	LAS 2 TORRES	26/04/1998	B	40.5	20/04/1998	Familiar
443490	Magdalena	Alcazar	01/01/2010	01/02/2011	LAS 2 TORRES	11/01/2010	B	40.5	05/01/2010	Familiar
443490	Magdalena	Alcazar	01/01/2010	01/02/2011	TOY STORY 3	07/01/2010	A	140.0	25/12/2009	Infantil
443490	Magdalena	Alcazar	01/01/2010	01/02/2011	BIUTIFUL	08/01/2010	B	100.0	25/12/2009	Familiar
345345	Pablo	Jimenez	11/05/2000	11/05/2001	FINAL FANTASY	21/05/2000	A	50.0	06/05/2000	Infantil

A. Realice el proceso de normalización hasta su 3^a forma normal empleando diagramas de dependencias.

10P

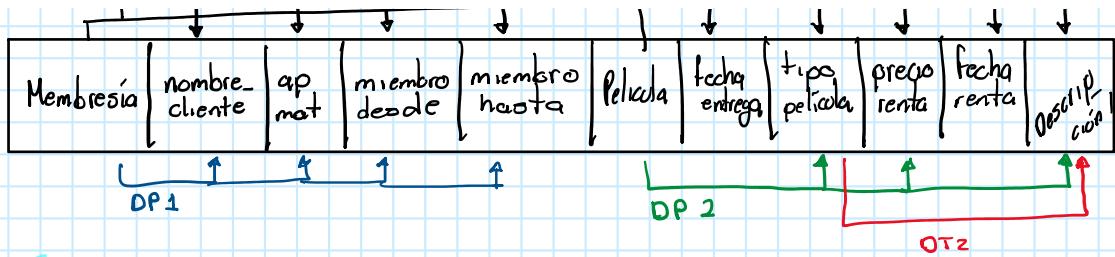
B. Construir el modelo relacional con base al resultado del proceso de normalización del punto anterior (No forzar el resultado, debe coincidir con el proceso obtenido en el inciso anterior).

5P

A)

10/10





→ FN2

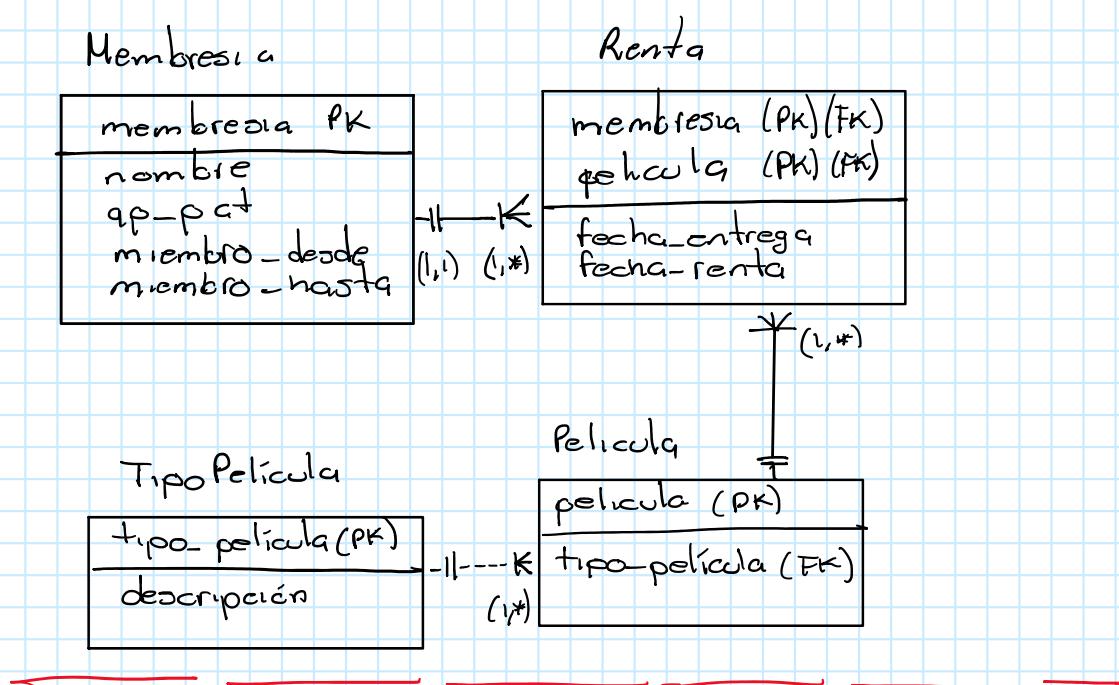
- membresia = nombre, ap_mat, ap_mat, miembro_desde, miembro_hasta
- pelicula = tipo_pelicula, precio_renta, descripción
- membresia, pelicula = fecha_entrega, fecha_renta

→ FN3

OT1 : tipo_pelicula → descripción

b) Modelo Relacional

5/5



► Ejercicio 4

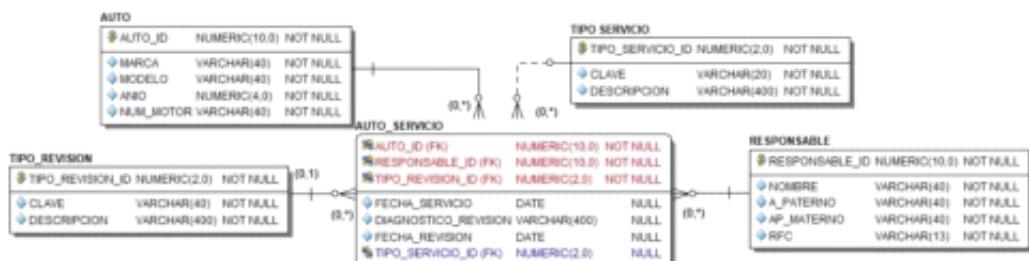
Las revisiones tienen como objetivo la prevención de fallas en los autos y se ofrecen de forma gratuita. La agencia cuenta con un catálogo de tipos de revisión a ofrecer. Cada vez que un auto es llevado a revisión se almacena el tipo de revisión realizada, la fecha de revisión y su diagnóstico.

Servicios:

El auto puede recibir varios servicios a lo largo de su vida útil. De forma similar, la agencia tiene un catálogo de tipos de servicio a ofrecer. Cada vez que un auto es llevado a servicio se registra la fecha del servicio, el tipo de servicio y el responsable. Cabe mencionar que cada tipo de servicio cuenta con un solo responsable asignado el cual certifica que el servicio se realizó de forma correcta.

El diseñador que implementó la BD cometió algunas anomalías. A continuación, se presenta el diseño realizado y una muestra de datos:

Modelo relacional:



Muestra de datos:

TIPO_SERVICIO

Tipo_servicio_id	clave	Descripción
0	Sin valor	Sin valor
1	SB	Servicio Básico
2	SI	Servicio Intermedio
3	SA	Servicio Avanzado

TIPO_REVISION

Tipo_revision_id	Clave	Descripción
0	Sin valor	Sin valor
1	RF	Revisión del sistema de frenos
2	RN	Revisión de neumáticos
3	RFI	Revisión de filtros

RESPONSABLE

Responsable_id	nombre	A_paterno	A_materno	RFC
0	NA	NA	NA	NA
100	Juan	Lopez	Lara	LOLAJ870304
200	Mary	Martinez	Mora	MAMR89731
300	Hugo	Morales	Ruiz	RUMOHU79233

AUTO_SERVICIO

Auto_id	Tipo_servicio_id	Tipo_revision_id	Fecha_servicio	Diagnostico_revision	Fecha_revision	Responsable_id
1	1	0	01/01/2017	null	null	100
1	2	0	01/02/2017	null	null	200
2	1	0	02/01/2017	null	null	100
2	2	0	02/02/2017	null	null	200
3	0	1	null	Sin problemas	30/01/2017	0
3	0	2	null	Con defectos encontrados	30/03/2017	0

A. Genere una lista de las anomalías que presenta este diseño haciendo referencia a los datos de muestra.

10P

A]

Revisiones | catálogo de tipos revisión

• el auto cada vez que se lleva a revisión se almacena el tipo de revisión la fecha y diagnóstico

Servicios

- El auto puede recibir varios servicios
- Catálogo de tipos de servicios
- Al llevarlo en servicio se almacena
 - Tipo - Responsable ← Just one guy

10/10

∴ Anomalías | ① Hay registros que indican el servicio o

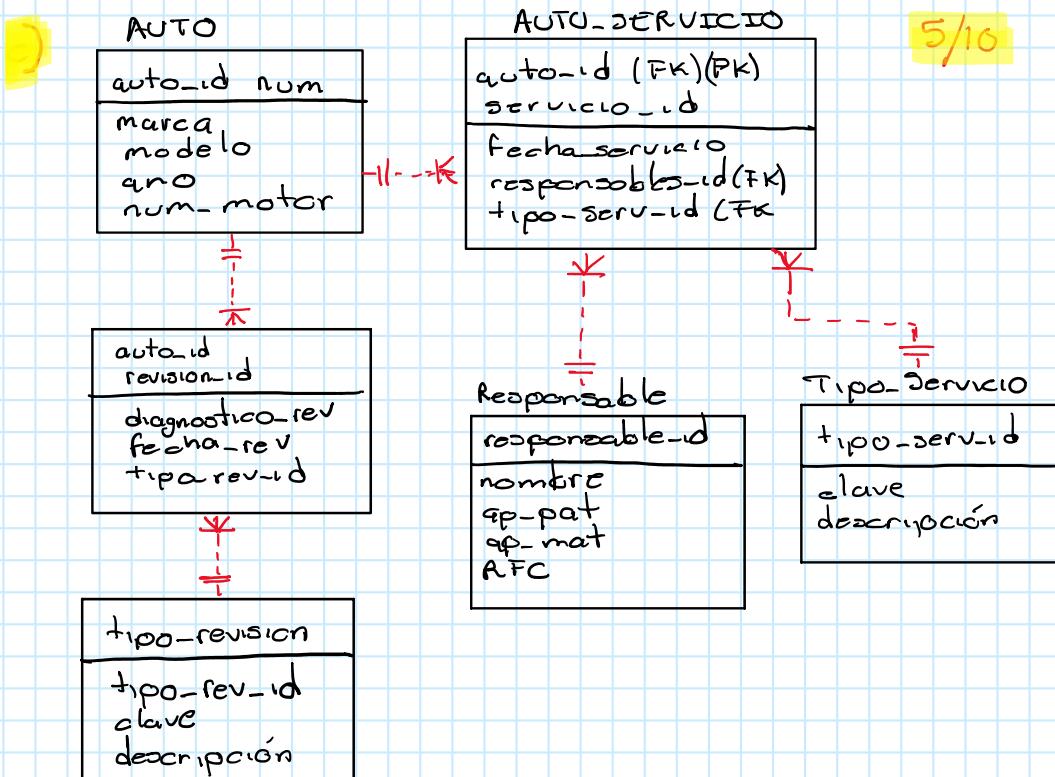
- ∴ Anomalías
- ① Hay registros que indican el servicio o revisión
 - ② Hay registros para un nulo responsable (Forzado)
 - ③ Para los registros de responsable de servicio no debería ser NULL.

C)

5/5

II Se requiere una 4FN debido a que servicio y revisión son independientes, es decir, realizar un refinamiento para omitir anomalías.

- Específicamente como servicio y revisión no tienen relación esto genera registro que sirven como PK que tristemente tienen casos nulos.
- * Lo que se puede hacer es una tabla intermedia que relacione a ambas tablas



VERSIÓN BOYCE CORRECTION

