FP055-Introduccion a las bases de datos.

AA3. El modelo relacional. Sentencias DDL

Contenido

SECCIÓN A. PARTE TEÓRICA.

1. define cada uno de los siguientes elementos y propiedades del modelo relacional e indica con qué elemento del diagrama de relación de entidades (ERD) de la actividad están asociados.
Ejemplo: Las tablas se asocian a las entidades.

1.1. Elementos del modelo relacional. Completa la siguiente tabla:

Modelo relacional	Sinónimos	Definición corta	Se asocian a (ERD)
Tabla	relación	Conjunto de datos estructurados en filas (Entidad) y columnas (Atributos).	Entidades
Registro	Fila	Una instancia especifica o entrada en una tabla. Representa una entidad única.	-
Campo	Columna	Un espacio individual para datos en una tabla, que representa los atributos de una Entidad.	-
Clave primaria	Primary Key	Es la forma de identificar de manera única e inequívoca a una entidad, nunca puede ser 0.	-
Clave candidata	Candidate Key	Es una o conjunto de columnas (Atributos) que puede ser usa como clave primaria. No debe contener redundancias.	-
Clave foránea (FOREIGN KEY)	-	Columna o conjuntos de columnas que establecen una relación entre dos tablas, referenciando la clave primaria de una de las tablas.	Entidades
Relación	-	Asociación lógica entre tablas que define como las tablas están conectadas o relacionadas, normalmente se usa un Verbo (acción).	-

1.2 Reglas de Integridad del Modelo Relacional. Explica las siguientes reglas:

a. Reglas de Integridad de Modelo

- Reglas de Integridad de Entidades
  - Regla de Unicidad de la Clave Primaria (PRIMARY KEY).
    Definición: Cada valor en la columna de clave primaria debe ser único en esa columna y no puede ser nulo.
    Ejemplo: En una tabla "Clientes" con la columna "ID_Cliente" como clave primaria, cada valor en esa columna debe ser único, y no puede haber valores nulos.

  - Regla de Entidad de la Clave Primaria.
    Definición: La clave primaria de una entidad no puede tener valores nulos, ya que representa la identificación única de cada instancia de la entidad.
    Ejemplo: En una tabla "Empleados" con la columna "ID_Empleado" como clave primaria, cada empleado debe tener un valor único en la columna "ID_Empleado", y este valor no puede ser nulo.

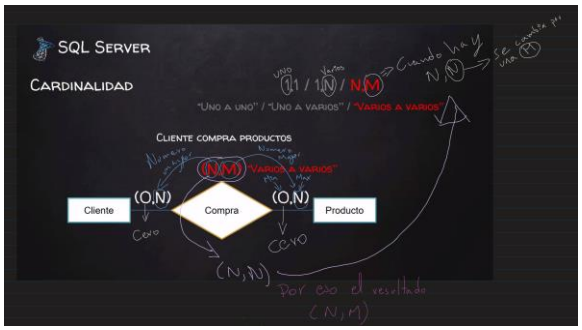
- ¿Qué es Integridad Referencial? Ejemplo:
  La integridad referencial garantiza la consistencia de las relaciones entre las tablas. Se asegura de que cualquier valor que aparece en una tabla que se refiere a otra tabla realmente exista en esa tabla.
  Ejemplo: Supongamos que tenemos dos tablas, "Ordenes" y "Clientes". La tabla "Ordenes" tiene una clave foránea "ID_Cliente" que se refiere a la clave primaria "ID_Cliente" en la tabla "Clientes".
  La integridad referencial garantiza que cualquier valor en la columna "ID_Cliente" de la tabla "Ordenes" debe existir en la columna "ID_Cliente" de la tabla "Clientes". No puede haber valores huérfanos en la tabla "Ordenes" que no tengan correspondencia en la tabla "Clientes".

b. Restricciones Semánticas. Explica las siguientes reglas:

- ¿Qué se conoce como DOMINIO en el ámbito de las Bases de Datos?
  Se refiere al conjunto de valores que puede tomar un atributo en una tabla. Define las posibles entradas o valores que son válidos para un atributo especifico.
- ¿Qué es una restricción de Dominio definida por el usuario? Ejemplo.
  Es una restricción que limita los valores que pueden ser insertados en una columna basándose en un conjunto de valores definido por el usuario.
  Ejemplo: Supongamos que tenemos una columna "Edad" en una tabla de "Usuarios". Si queremos imponer una restricción de dominio definida por el usuario, podríamos especificar que solo se permiten valores de edad entre 18 y 99. Cualquier intento de insertar un valor fuera de este rango violaría la restricción.

- ¿Qué es una restricción de Usuario? Ejemplo.
  Es una restricción adicional impuesta por el usuario para garantizar la integridad y calidad de los datos. Estas restricciones pueden ir más allá de las restricciones estándar proporcionadas por el sistema de gestión de bases de datos (DBMS).
  Ejemplo: Supongamos que en una tabla "Empleados" queremos asegurarnos de que ningún empleado pueda tener más de 40 horas de trabajo a la semana. Podemos definir una restricción de usuario que verifique esta condición y evite que se insertan o actualicen registros que violen esta regla específica del negocio.

1.3 Transformación de Relaciones.



A. Describe la forma en la cual se transforman las relaciones 1:1 del Modelo Entidad-Relación al Modelo Relacional. Pon un ejemplo.

En una relación 1:1 del Modelo Entidad-Relación (ER), donde una entidad en un conjunto está relacionada con una entidad en otro conjunto, la transformación al Modelo Relacional implica fusionar ambas entidades en una sola tabla. La clave primaria de una entidad se convierte en una clave foránea en la otra entidad.

Ejemplo:

Supongamos que tenemos dos entidades, "Persona" y "Pasaporte", con una relación 1:1. La transformación se vería así:

Modelo Entidad-Relación (ER):

Entidad "Persona" (ID_Persona, Nombre, Edad, ...).
Entidad "Pasaporte" (ID_Pasaporte, Numero, FechaExpiracion, ...).

Modelo Relacional:

Tabla "Persona" (ID_Persona PRIMARY KEY, Nombre, Edad, ID_Pasaporte FOREIGN KEY REFERENCES Pasaporte(ID_Pasaporte)).
Tabla "Pasaporte" (ID_Pasaporte PRIMARY KEY, Numero, FechaExpiracion, ...).

B. Describe la forma en la cual se transforman las relaciones 1:n del Modelo Entidad-Relación al Modelo Relacional. Pon un ejemplo.

En una relación 1:n, donde una entidad en un conjunto puede estar relacionada con varias entidades en otro conjunto, la transformación al Modelo Relacional implica agregar la clave primaria de la entidad del conjunto "1" como clave foránea en la entidad del conjunto "n".

Ejemplo:

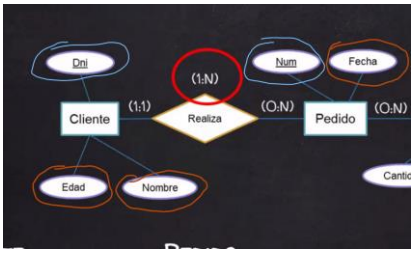
Supongamos que tenemos dos entidades, "Departamento" y "Empleado", con una relación 1:n. La transformación se vería así:

Modelo Entidad-Relación (ER):

Entidad "Departamento" (ID_Departamento, Nombre, ...).
Entidad "Empleado" (ID_Empleado, Nombre, Salario, ID_Departamento).

Modelo Relacional:

Tabla "Departamento" (ID_Departamento PRIMARY KEY, Nombre, ...).
Tabla "Empleado" (ID_Empleado PRIMARY KEY, Nombre, Salario, ID_Departamento FOREIGN KEY REFERENCES Departamento (ID_Departamento)).



C. Describe la forma en la cual se transforman las relaciones n:m del Modelo Entidad-Relación al Modelo Relacional. Pon un ejemplo.

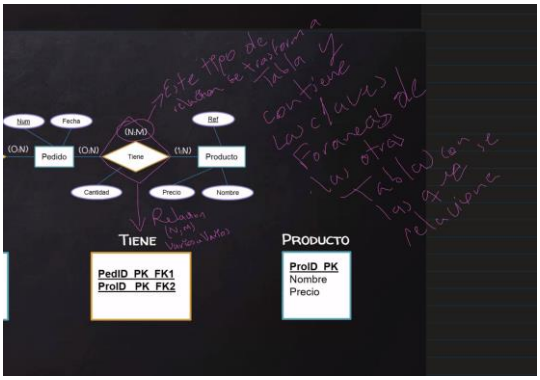
En una relación n:m, donde varias entidades en un conjunto pueden estar relacionadas con varias entidades en otro conjunto, la transformación al Modelo Relacional implica crear una tabla intermedia que contiene las claves foráneas de ambas entidades.

Ejemplo:

Supongamos que tenemos dos entidades, "Estudiante" y "Curso", con una relación n:m. La transformación se vería así:

Modelo Entidad-Relación (ER):

Entidad "Estudiante" (ID_Estudiante, Nombre, ...).
Entidad "Curso" (ID_Curso, Nombre, ...).
Relación n:m "Inscripcion" (ID_Estudiante, ID_Curso).
Modelo Relacional:
Tabla "Estudiante" (ID_Estudiante PRIMARY KEY, Nombre, ...).
Tabla "Curso" (ID_Curso PRIMARY KEY, Nombre, ...).
Tabla "Inscripcion" (ID_Estudiante FOREIGN KEY REFERENCES Estudiante(ID_Estudiante), ID_Curso FOREIGN KEY REFERENCES Curso(ID_Curso), PRIMARY KEY (ID_Estudiante, ID_Curso)).



2. Sentencias DDL (Data Definition Languaje) de SQL. Para cada objeto de la primera columna.

Marca con una x las sentencias que admite:

Instrucción | sentencia	CREATE	ALTER	DROP	RENAME	TRUNCATE
Objetos	Crear	Modificar	Eliminar	Renombrar	Vaciar
DATABASE | SCHEMA	x	x	x	X	
TABLE * [TEMPORARY TABLE]	x	x	x	X	X
INDEX	x	x	x	X	x
VIEW	x	x	x	X	

CREATE: Puede ser utilizada para crear todos los objetos mencionados.

ALTER: Puede ser utilizada para modificar (alterar) todos los objetos mencionados.

DROP: Puede ser utilizada para eliminar (drop) todos los objetos mencionados.

RENAME: Puede ser utilizada para renombrar todos los objetos mencionados.

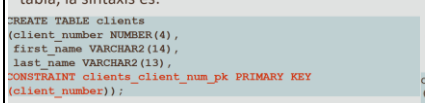
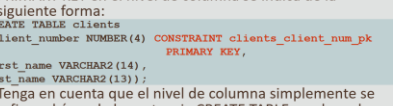
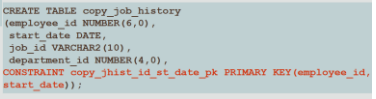
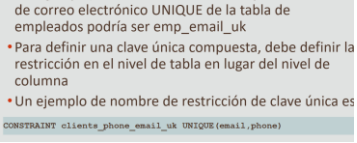
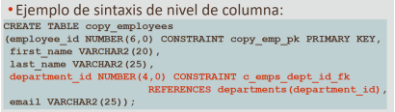
TRUNCATE: Puede ser utilizada para vaciar (truncar) las tablas y las tablas temporales.

3. Restricciones.

La sentencia ALTER TABLE de SQL tiene dos propósitos:

- Permite agregar, modificar o eliminar columnas en una tabla:
  - **Agregar una Columna:**
    - `ALTER TABLE nombre_tabla`
    - `ADD nombre_columna tipo_dato;`
  - **Modificar una Columna:**
    - `ALTER TABLE nombre_tabla`
    - `MODIFY nombre_columna nuevo_tipo_dato;`
  - **Eliminar una Columna:**
    - `ALTER TABLE nombre_tabla`
    - `DROP COLUMN nombre_columna;`
- Permite agregar y eliminar restricciones (constraints) en una tabla.
  - **Agregar una Restricción (por ejemplo, PRIMARY KEY):**
    - `ALTER TABLE nombre_tabla`
    - `ADD CONSTRAINT nombre_restriccion PRIMARY KEY (nombre_columna);`
  - **Eliminar una Restricción:**
    - `ALTER TABLE nombre_tabla`
    - `DROP CONSTRAINT nombre_restriccion;`
  - **Agregar una Restricción UNIQUE:**
    - `ALTER TABLE nombre_tabla`
    - `ADD CONSTRAINT nombre_restriccion UNIQUE (nombre_columna);`
  - **Agregar una Restricción FOREIGN KEY:**
    - `ALTER TABLE nombre_tabla`
    - `ADD CONSTRAINT nombre_restriccion FOREIGN KEY (nombre_columna) REFERENCES otra_tabla(otra_columna);`

En referencia a las restricciones, completa la siguiente tabla:

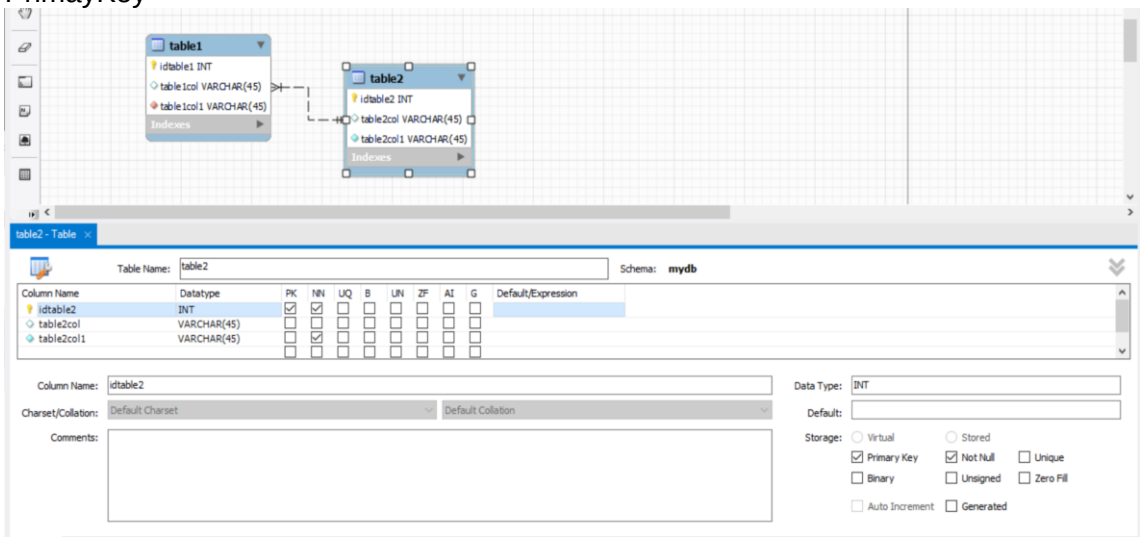
<table>
<tr><th>RESTRICCIÓN</th><th>OBJETIVO</th><th>EJEMPLO</th></tr>
<tr><td rowspan="2">PRIMARY KEY</td><td>Identificar registros únicos.</td><td>CREATE TABLE Ejemplo (ID INT PRIMARY KEY, nombre VARCHAR(255));</td></tr>
<tr><td>Restricciones PRIMARY KEY
• Una restricción PRIMARY KEY es una regla de que los valores de una columna o una combinación de columnas deben identificar de forma única cada fila de una tabla
• No puede aparece ningún valor de clave primaria en más de una fila de la tabla
• Para cumplir una restricción PRIMARY KEY, las dos condiciones siguientes deben ser verdaderas:
– Ninguna columna que forme parte de la clave primaria puede contener un valor nulo
– Una tabla solo puede tener una clave primaria
• Las restricciones PRIMARY KEY se pueden definir en nivel de columna o de tabla
• Sin embargo, si se crea una CLAVE PRIMARIA compuesta, se debe definir en el nivel de tabla
• Al definir columnas PRIMARY KEY, es una buena práctica utilizar el sufijo _pk en el nombre de restricción
• Por ejemplo, el nombre de restricción para la columna PRIMARY KEY denominada client_number de la tabla llamada CLIENTS podría ser clients_client_num_pk</td><td>• Para crear la restricción PRIMARY KEY en el nivel de tabla, la sintaxis es:
 
• Tenga en cuenta que el nombre de la columna PRIMARY KEY sigue el tipo de restricción y está entre paréntesis
Restricciones PRIMARY KEY
• En una sentencia CREATE TABLE, la sintaxis de restricción PRIMARY KEY en el nivel de columna se indica de la siguiente forma:
 
• Tenga en cuenta que el nivel de columna simplemente se refiere al área de la sentencia CREATE TABLE en el que las columnas están definidas
• El nivel de tabla hace referencia a la última línea de la sentencia debajo de la lista de nombres de columna individuales
• Para definir una CLAVE PRIMARIA compuesta, debe definir la restricción en el nivel de tabla en lugar del nivel de columna
• A continuación se muestra un ejemplo de restricción de clave primaria compuesta:
  </td></tr>
<tr><td rowspan="2">UNIQUE</td><td>Garantizar valores únicos.</td><td>CREATE TABLE Ejemplo (Email CRACHAR(100) UNIQUE, OtroCampo INT);</td></tr>
<tr><td>Restricción UNIQUE
• Una restricción UNIQUE necesita que todos los valores de una columna o juego de columnas (clave compuesta) sean únicos; es decir, que dos filas de una tabla no pueden tener valores duplicados
• Por ejemplo, puede ser importante para un negocio garantizar que no hay dos personas que tengan la misma dirección de correo electrónico
• La columna de correo electrónico se puede definir con una restricción UNIQUE
• La columna o juego de columnas que se define como UNIQUE se denomina una clave única
• Si la combinación de dos o más columnas debe ser única para cada entrada, se dice que la restricción es una clave única compuesta
• La indicación de que todas las combinaciones de correo electrónico y apellido deben ser ÚNICAS es un ejemplo de clave única compuesta
• La palabra "clave" hace referencia a las columnas, no a los nombres de restricciones</td><td>Ejemplo de Restricción UNIQUE
• Si la columna de correo electrónico de la tabla se define con una restricción UNIQUE, ninguna otra entrada de cliente puede tener el mismo correo electrónico
• ¿Qué ocurre si dos clientes viven en el mismo domicilio y comparten una dirección de correo electrónico?
 
Definición de Restricciones UNIQUE
• Al definir restricciones UNIQUE, es habitual utilizar el sufijo _uk en el nombre de restricción
• Por ejemplo, el nombre de restricción para la columna de correo electrónico UNIQUE de la tabla de empleados podría ser emp_email_uk
• Para definir una clave única compuesta, debe definir la restricción en el nivel de tabla en lugar del nivel de columna
• Un ejemplo de nombre de restricción de clave única es
  </td></tr>
<tr><td rowspan="2">FOREIGN KEY</td><td>Mantener integridad referencial.</td><td>CREATE TABLA Detalles (ID INT, EjemploID INT. FOREIGN KEY (EjemploID) REFERENCES Ejemplo(ID));</td></tr>
<tr><td>Restricción FOREIGN KEY
• Para definir una restricción FOREIGN KEY, es una buena práctica utilizar el sufijo _fk en el nombre de restricción
• Por ejemplo, el nombre de restricción para la columna de CLAVE AJENA department_id de la tabla de empleados podría ser emps_dept_id_fk</td><td>Sintaxis de Restricción FOREIGN KEY
• La sintaxis para definir una restricción FOREIGN KEY necesita una referencia a la tabla y columna de la tabla principal
• Una restricción FOREIGN KEY en una sentencia CREATE TABLE se puede definir de la siguiente forma
• Ejemplo de sintaxis de nivel de columna:
  </td></tr>
<tr><td></td><td>Establecer condiciones especiales.</td><td>CREATE TABLE Empleados (Edad INT CHECK (Edad >= 18));</td></tr>
</table>

CHECK	Restricciones CHECK • La restricción CHECK define explícitamente una condición que se debe cumplir • Para cumplir la restricción, cada una de las filas de la tabla debe hacer que condición sea True o desconocida (debido a un valor nulo) • La condición de una restricción de CHECK puede hacer referencia a cualquier columna de la tabla especificada, pero no a columnas de otras tablas	Ejemplo de Restricción CHECK • Esta restricción CHECK garantiza que un valor introducido para end_date sea posterior a start_date `CREATE TABLE copy_job_history (employee_id NUMBER(6,0), start_date DATE, end_date DATE, job_id VARCHAR2(10), department_id NUMBER(4,0), CONSTRAINT cjhist_emp_id_st_date_pk PRIMARY KEY(employee_id, start_date), CONSTRAINT cjhist_end_ck CHECK (end_date > start_date));` • Como esta RESTRICCIÓN CHECK hace referencia a dos columnas de la tabla, se DEBE definir en el nivel de tabla
DEFAULT	Proporcionar valor predeterminado.	CREATE TABLE Ejemplo (FechaInicio DATE DEFAULT '2023-11-28');

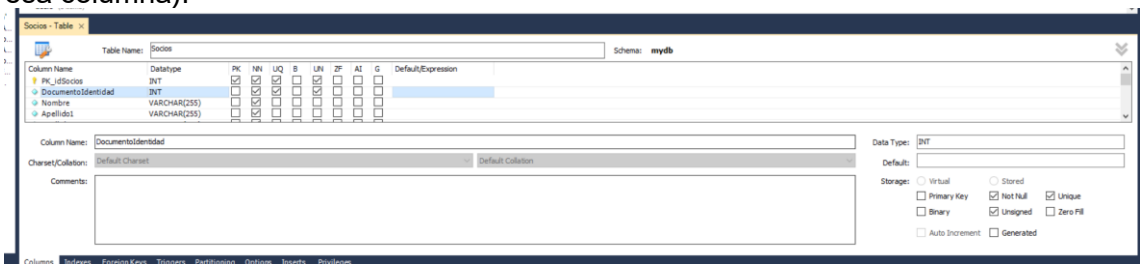
4. Claves compuestas y otras propiedades.

Indica realizando una captura de pantalla de MySQL Workbench.

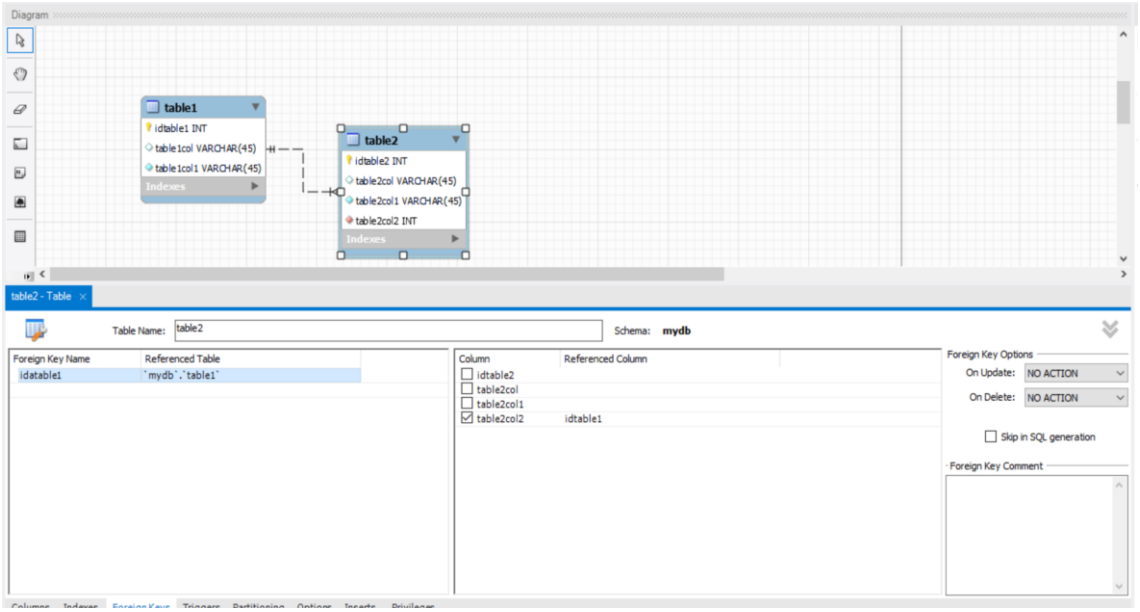
- ¿Cómo se genera una Clave Primaria, Candidata o Foránea cuando está compuesta por más de un campo?
  PrimayKey



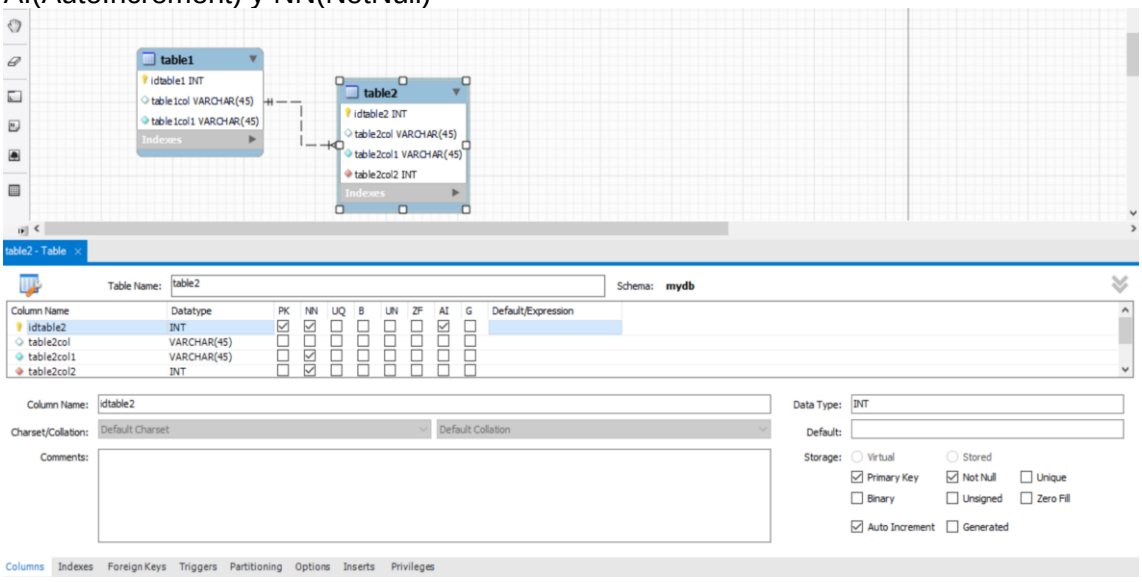
Candidata ("UNIQUE" se refiere a una restricción en una columna de una tabla en una base de datos. Cuando se aplica la restricción "UNIQUE" a una columna, significa que los valores en esa columna deben ser únicos en todas las filas de la tabla. Esto significa que no puede haber duplicados en esa columna).



Foranea



- ¿Cómo se pueden agregar las propiedades AUTOINCREMENT y NOT NULL a un campo y qué significan estas dos propiedades?
  AI(AutoIncrement) y NN(NotNull)



5. Tipos de Datos MySQL.

MySQL admite una gran variedad de tipos de datos. Entre todos ellos se seleccionaron los que más se utilizan para que completes la tabla siguiente:

Tipo de dato	Grupo (Numérico, Fecha, Cadena)	Capacidad, formato...
BIT	Numérico	1bit
INT	Numérico	-2147483648 a 2147483647
FLOAT	Numérico	precisión Variable
DECIMAL (m,n)	Numérico	precisión fija (m total, n decimales)
DATE	Fecha	'YYYY-MM-DD'
DATETIME	Fecha	'YYYY-MM-DD HH:MI:SS'
TIME	Fecha	'HH:MI:SS'
CHAR (n)	Cadena	Cadena de longitud Fija
VARCHAR (n)	Cadena	Cadena de longitud variable
ENUM	Cadena	Lista de valores predefinidos
SET	Cadena	Conjunto de valores predefinidos

6. ¿Cuál es la diferencia entre los tipos de campo FLOAT y DECIMAL y cómo se interpretan los parámetros en este último tipo de campo? ¿Y entre CHAR y VARCHAR?

La elección entre FLOAT y DECIMAL depende de la precisión requerida para los valores numéricos, mientras que la elección entre CHAR y VARCHAR depende de si se necesita una longitud fija o variable para las cadenas de texto.

- FLOAT:
  - El tipo de dato FLOAT se utiliza para representar números de punto flotante (números con decimales).
  - Almacena números en formato de punto flotante, lo que significa que la precisión y la escala pueden variar.
  - Es adecuado para valores con decimales, pero la precisión puede ser limitada, lo que puede llevar a errores de redondeo en cálculos.
  - No requiere especificar la precisión o la escala al definir el campo.
- DECIMAL:
  - El tipo de dato DECIMAL se utiliza para representar números decimales con precisión fija.
  - Almacena números con una precisión específica, donde la precisión se refiere al número total de dígitos (antes y después del punto decimal) y la escala se refiere al número de dígitos después del punto decimal.
  - Es adecuado para valores que requieren una precisión precisa, como valores monetarios o científicos.

- Al definir un campo DECIMAL, debes especificar la precisión y la escala. Por ejemplo, DECIMAL (10, 2) significa que puede almacenar números con hasta 10 dígitos en total, con 2 de ellos después del punto decimal.

En cuanto a la diferencia entre CHAR y VARCHAR:

- CHAR:
  - El tipo de dato CHAR se utiliza para almacenar cadenas de longitud fija.
  - Requiere que todos los valores en ese campo tengan la misma longitud, lo que significa que se pueden rellenar con espacios en blanco si es necesario.
  - Es útil cuando se sabe que todas las cadenas tendrán la misma longitud, ya que puede ahorrar espacio de almacenamiento.
- VARCHAR:
  - El tipo de dato VARCHAR se utiliza para almacenar cadenas de longitud variable.
  - Permite que los valores tengan longitudes diferentes y no desperdicia espacio de almacenamiento en espacios en blanco.
  - Es adecuado para almacenar cadenas de longitud variable, como nombres de personas o descripciones.

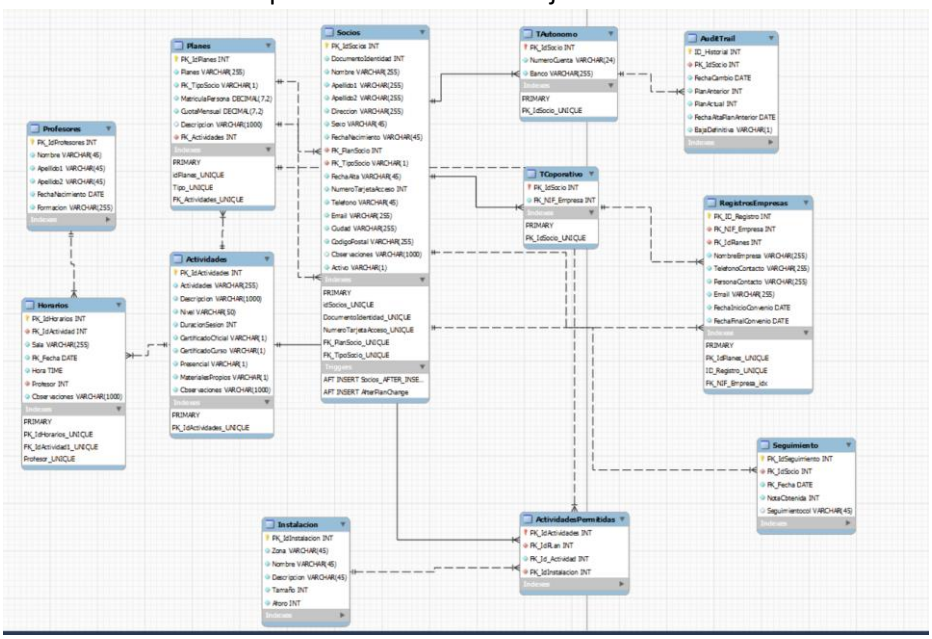
SECCIÓN B. PARTE PRÁCTICA

2.1 Instalar el servidor de Base de Datos MySQL y el Gestor de Bases de Datos MySQL Workbench.

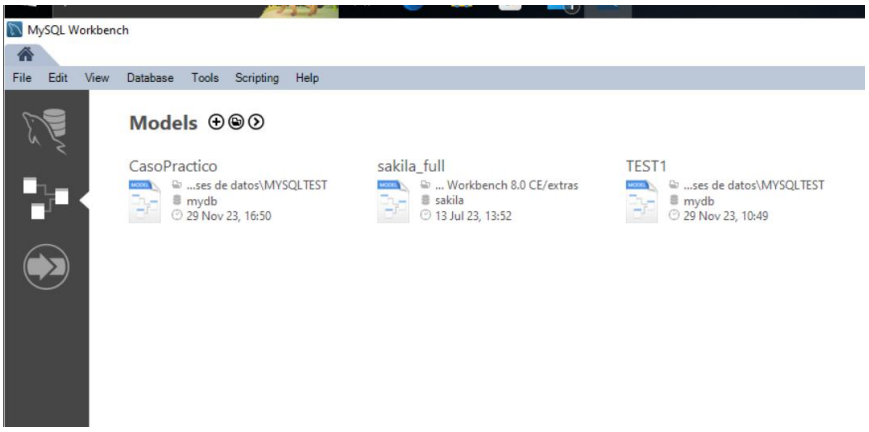
2.2 Realizar la transformación del diagrama entidad relación de la segunda actividad al modelo relacional (estructura de tablas) en MySQL Workbench.

Tienes que volver a crear de nuevo las tablas, campos, relaciones, pero ahora en Workbench y siguiendo el modelo relacional y asignando las Primary key y Foreign keys correctamente.

Creo que esta vez entendí mejor la actividad!

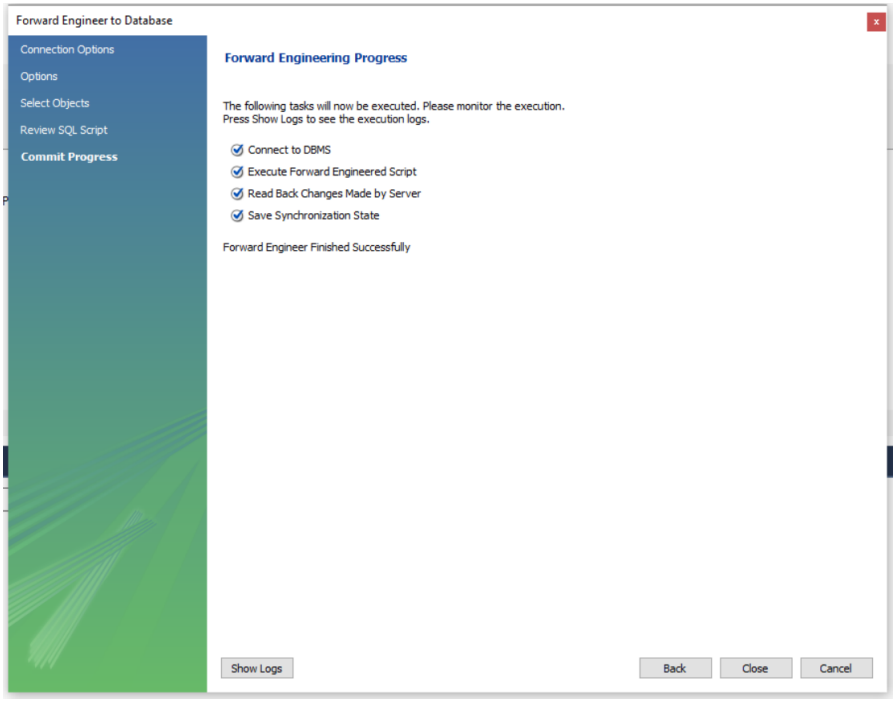


Traslada el ERD de la AA2 a MySQL Workbench y crea el Modelo Relacional con las pautas siguientes, que permitirán apreciar mejor las propiedades de los campos de las tablas y las relaciones:



2.3 Crear la Base de Datos en MySQL a partir del script en tu servidor local.

Utilizando la función Database - Forward Engineering de MySQL Workbench, genera la Base de Datos a partir del Modelo Relacional.
Cierra la ventana de modelos.



2.4 Generar un script .SQL exportando la Base de Datos recién creada.

Conecta con tu servidor local desde MySQL Workbench (clicando sobre Local instance…) y exporta la base de datos .SQL generada en el paso anterior (opción Server-Data Export).

2 Bibliografía.

2.1 Instalación y primera conexión MySQL Sever y Workbench
Video Tutorial.

2.2 Error Could not acquire managemment acces for adminitration
Video Tutorial.

2.3 Lista de Primeros pasos MySQL
Video Tutorial.