|  |
| --- |
| **SPRINT 2: Repositorio de Código - Diseño Base de Datos** |

|  |  |
| --- | --- |
| **Identificación Proyecto** | |
| Nombre Proyecto: |  |
| Número Equipo: |  |
| **Integrantes del equipo** | |
| Rol  (Líder-Desarrollador – Cliente) | Nombre |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

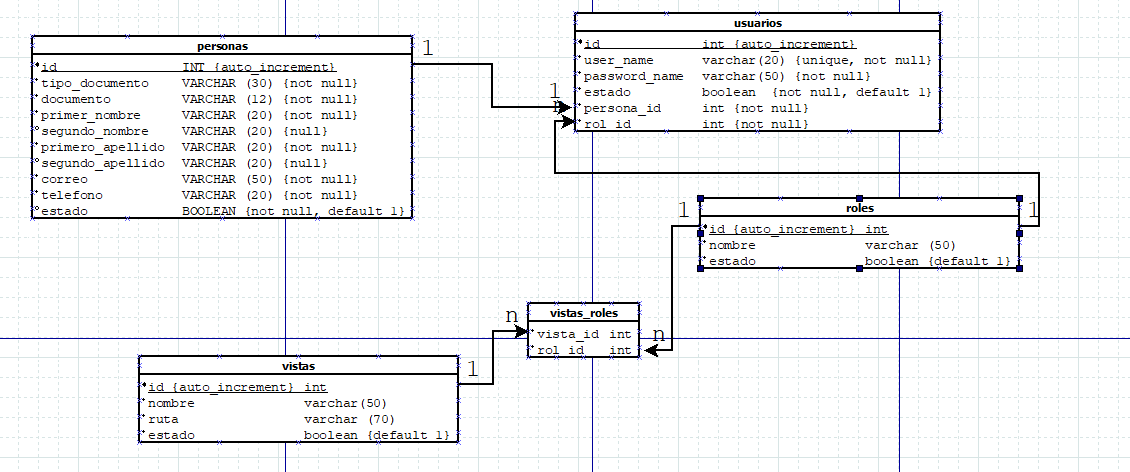
|  |
| --- |
| **Repositorio de Código GitLab o GitHub** |

Como evidencia del repositorio de código, creado con GitLab o GitHub, además de la URL del repositorio, se debe presentar capturas de pantalla donde se visualicen aspectos:

* Creación del proyecto del repositorio.
* Integrantes del equipo invitados.
* Evidencia de la realización de alguna actualización (commit), donde se visualice la actualización y el historial de actualizaciones (Versiones)

|  |
| --- |
| **Diseño de la Base de Datos (Proceso de normalización)** |

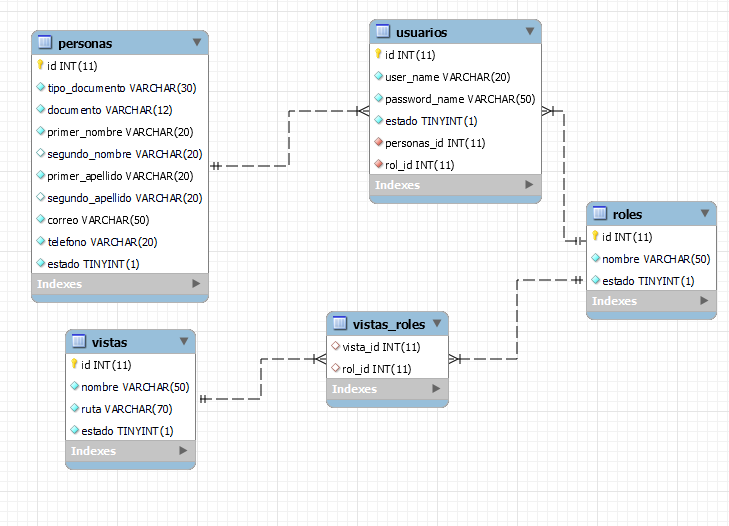
Como evidencia del Diseño de la Base de Datos, se debe presentar el proceso de normalización efectuado (Formas Normales).



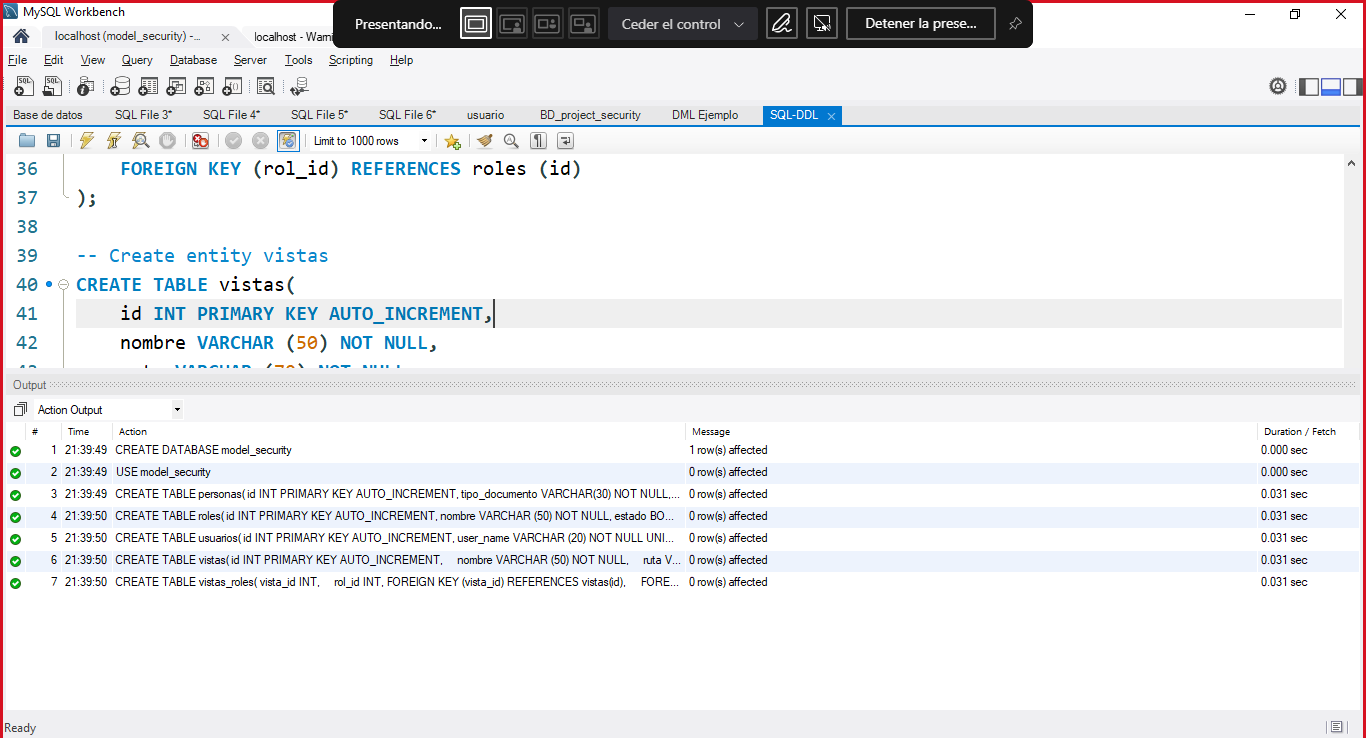
|  |
| --- |
| **Esquema de la Base de Datos (Código SQL)** |

Como evidencia del Esquema de la Base de Datos, se debe presentar el código SQL de creación de la base de datos.

Esquema de la base de datos



Sentencias SQL-DDL



-- Create data base model\_security

CREATE DATABASE model\_security;

USE model\_security;

-- Create entity personas

CREATE TABLE personas(

    id INT PRIMARY KEY AUTO\_INCREMENT,

    tipo\_documento VARCHAR(30) NOT NULL,

    documento VARCHAR(12) NOT NULL,

    primer\_nombre VARCHAR(20) NOT NULL,

    segundo\_nombre VARCHAR(20) NULL,

    primer\_apellido VARCHAR(20) NOT NULL,

    segundo\_apellido VARCHAR(20) NULL,

    correo VARCHAR(50) NOT NULL,

    telefono VARCHAR(20) NOT NULL,

    estado BOOLEAN NOT NULL DEFAULT 1

);

-- Create entity roles

CREATE TABLE roles(

    id INT PRIMARY KEY AUTO\_INCREMENT,

    nombre VARCHAR (50) NOT NULL,

    estado BOOLEAN NOT NULL DEFAULT 1

);

-- Create entity usuarios

CREATE TABLE usuarios(

    id INT PRIMARY KEY AUTO\_INCREMENT,

    user\_name VARCHAR (20) NOT NULL UNIQUE,

    password\_name VARCHAR (50) NOT NULL,

    estado BOOLEAN NOT NULL DEFAULT 1,

    persona\_id INT NOT NULL,

    rol\_id INT NOT NULL,

    CONSTRAINT fk\_persona\_id\_usuarios\_personas FOREIGN KEY (persona\_id) REFERENCES personas(id),

    CONSTRAINT fk\_rol\_id\_usuarios\_roles FOREIGN KEY (rol\_id) REFERENCES roles (id)

);

-- Create entity vistas

CREATE TABLE vistas(

    id INT PRIMARY KEY AUTO\_INCREMENT,

    nombre VARCHAR (50) NOT NULL,

    ruta VARCHAR (70) NOT NULL,

    estado BOOLEAN NOT NULL DEFAULT 1

);

-- Create entity vista\_roles

CREATE TABLE vistas\_roles(

    vista\_id INT,

    rol\_id INT,

    CONSTRAINT fk\_vista\_id\_vistas\_roles\_vistas FOREIGN KEY (vista\_id) REFERENCES vistas(id),

    CONSTRAINT fk\_rol\_id\_vistas\_roles\_roles FOREIGN KEY (rol\_id) REFERENCES roles(id)

);

|  |
| --- |
| **Evidencia JIRA (Seguimiento del proyecto)** |

Como evidencia del seguimiento del proyecto con la metodología ágil SCRUM, utilizando el software JIRA, se debe presentar capturas de pantalla donde se visualice la ejecución de los Sprint con las historias de usuario relacionadas con el repositorio de código y el diseño de la base de datos.

|  |
| --- |
| **Evidencias de las Reuniones de Equipo** |

Como evidencia de las reuniones que efectúa el equipo del proyecto, presentar capturas de pantalla de las reuniones efectuadas y si lo consideran pertinente algunas actas de las reuniones.