INFORME DE PRÁCTICAS PREPROFESIONALES EN BASE DE DATOS

ESTUDIANTE: JIMBO JARAMILLO DARWIN RICARDO

ASIGNATURA: BASE DE DATOS

CICLO: TERCERO

PARALELO: A

PERIODO ACADÉMICO: ABRIL - AGOSTO DE 20025

DOCENTE SUPERVISOR: RENÉ GUAMAN-QUINCHE

FECHA DE ENTREGA: 30 DE JULIO DE 2025

INTRODUCCIÓN

Estas prácticas preprofesionales tienen como objetivo el fortalecer la información académica de los estudiantes a través de una vinculación con el entorno laboral, permitiéndonos aplicar los conocimientos adquiridos a lo largo del ciclo académico, desarrollando habilidades técnicas y contribuyendo al cumplimiento de objetivos institucionales.

El área donde se llevaron a cabo las prácticas preprofesionales fué principalmente el departamento en el que me encuentro viviendo, pero también se llevaron a cabo en ciertos momentos libres que hubieron en las horas de clase en el laboratorio 1 del bloque A2 y en el aula del bloque A4.

Durante el desarrollo de estas prácticas, las bases de datos empleadas desarrollaron un papel fundamental en el soporte de almacenamiento, la organización de datos y la consulta eficiente de la información. En el contexto de sistemas como la gestión de requisitos, la estructuración de las bases de datos garantiza la integridad, trazabilidad, seguridad y disponibilidad de la información, facilitando la colaboración entre los estudiantes y docentes.

OBJETIVOS

Objetivo General:

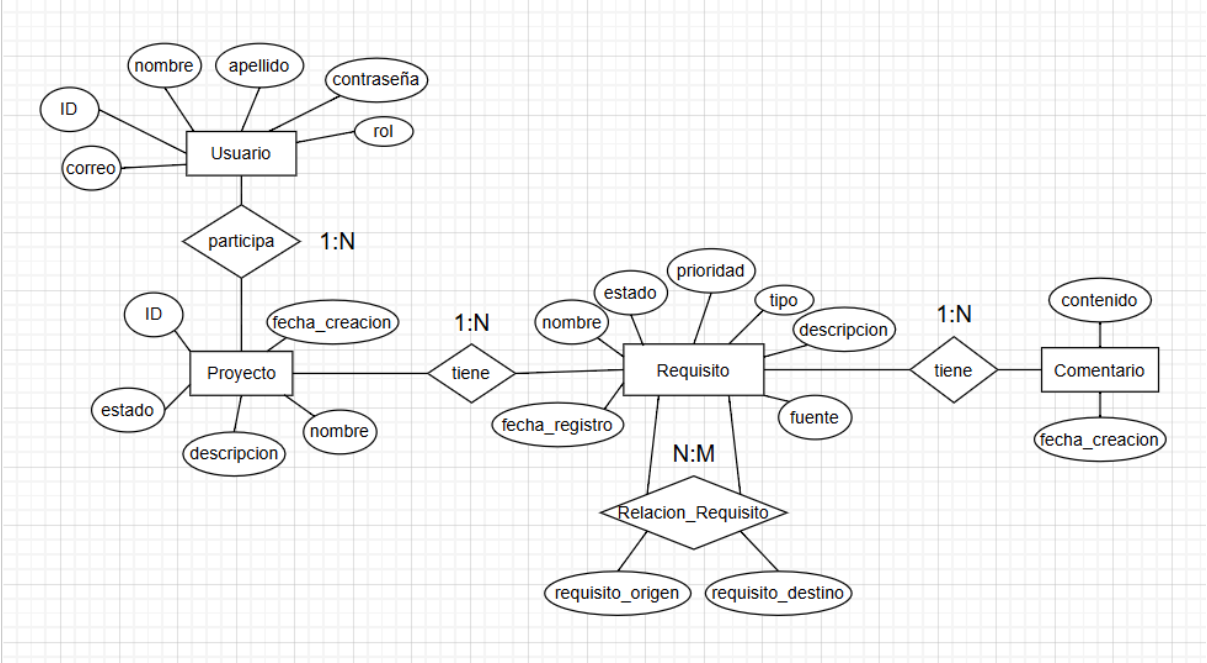
- Describir el propósito principal de las prácticas en el ámbito de bases de datos.

Objetivos Específicos:

- Analizar las necesidades de información para diseñar esquemas de base de datos coherentes.
- Desarrollar modelos entidad-relación y modelos relacionales que representen correctamente las entidades y relaciones del sistema gestionado.
- Implementar la base de datos MySQL, asegurando integridad y uso adecuado de claves primarias y foráneas.

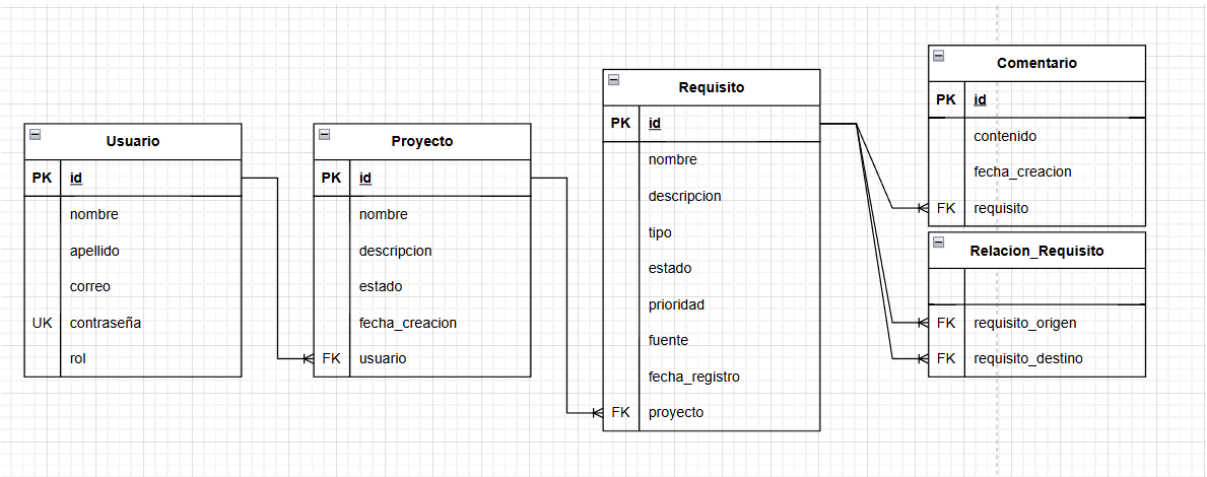
ACTIVIDADES REALIZADAS

Modelo Entidad-Relación:



El diagrama Entidad-Relación presenta un sistema de gestión de requisitos (SGR) para proyectos de software donde se incluyen las entidades principales como el Usuario, Proyecto, Requisito y Comentario. Su objetivo es permitir que los usuarios gestionen proyectos y definan, que relacionen y que comenten sobre los requisitos de dichos proyectos.

Modelo Relacional:



Detalle de la normalización de la base de datos:

1. **Todos los atributos deben ser atómicos y cada campo debe tener un solo valor**
   Todos los campos como por ejemplo: nombre, correo, estado, entre otros, son anatómicos. No existen campos multivaluados ni listas de valores, por lo que cumple el primer criterio.

2. **Todos los atributos no clave dependen totalmente de una clave primaria.**
   Todas las tablas tienen claves primarias simples, a excepción de "Relacion_Requisito", que se puede ver como una clave compuesta, sin embargo, no tiene otros atributos además de las claves, por lo que cumple con el segundo criterio.

3. **No existen dependencias transitivas entre atributos no clave.**
   Ninguna de las tablas presenta dependencias transitivas, por lo que cumple con el tercer criterio.

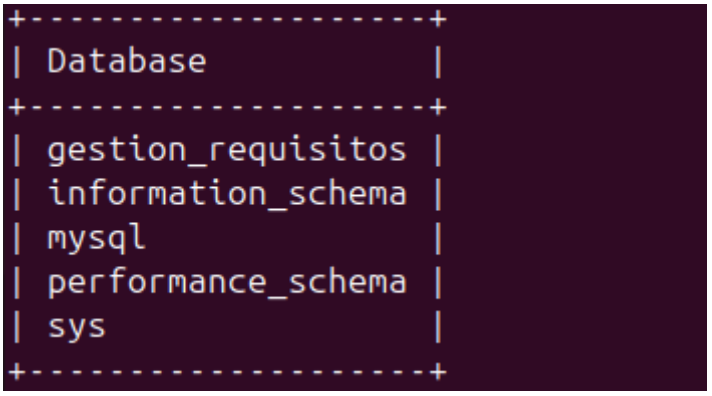
Diseño Físico (MySQL):

La base de datos en MySQL se creó directamente con el comando:

CREATE DATABASE gestion_requisitos;

y se pudo visualizar con el comando:

SHOW DATABASES;



Acto seguido, se crearon las tablas por medio de los siguientes comandos:

```
CREATE TABLE Usuario(
    id INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(50) NOT NULL,
    apellido VARCHAR(50) NOT NULL,
```

correo VARCHAR(30) NOT NULL UNIQUE,
contraseña VARCHAR(100) NOT NULL,
rol ENUM('Estudiante', 'Docente') NOT NULL
);

Se observa con el comando:

DESC Usuario;

```
+-------------+-------------------------------+------+-----+---------+----------------+
| Field       | Type                          | Null | Key | Default | Extra          |
+-------------+-------------------------------+------+-----+---------+----------------+
| id          | int                           | NO   | PRI | NULL    | auto_increment |
| nombre      | varchar(50)                   | NO   |     | NULL    |                |
| apellido    | varchar(50)                   | NO   |     | NULL    |                |
| correo      | varchar(30)                   | NO   | UNI | NULL    |                |
| contraseña  | varchar(100)                  | NO   |     | NULL    |                |
| rol         | enum('Estudiante','Docente') | NO   |     | NULL    |                |
+-------------+-------------------------------+------+-----+---------+----------------+
6 rows in set (0,01 sec)
```

CREATE TABLE Proyecto(
id INT PRIMARY KEY AUTO_INCREMENT,
nombre VARCHAR(50) NOT NULL,
descripcion VARCHAR(1000) NOT NULL,
estado ENUM('Activo', 'Finalizado') NOT NULL,
fecha_creacion DATE NOT NULL,
usuario INT NOT NULL,
FOREIGN KEY (usuario) REFERENCES Usuario(id)
);

Se observa con el comando:

DESC Proyecto;

```
+----------------+-------------------------------+------+-----+---------+----------------+
| Field          | Type                          | Null | Key | Default | Extra          |
+----------------+-------------------------------+------+-----+---------+----------------+
| id             | int                           | NO   | PRI | NULL    | auto_increment |
| nombre         | varchar(50)                   | NO   |     | NULL    |                |
| descripcion    | varchar(1000)                 | NO   |     | NULL    |                |
| estado         | enum('Activo','Finalizado') | NO   |     | NULL    |                |
| fecha_creacion | date                          | NO   |     | NULL    |                |
| usuario        | int                           | NO   | MUL | NULL    |                |
+----------------+-------------------------------+------+-----+---------+----------------+
6 rows in set (0,01 sec)
```

CREATE TABLE Requisito(
id INT PRIMARY KEY AUTO_INCREMENT,

```
nombre VARCHAR(50) NOT NULL,
descripcion VARCHAR(1000) NOT NULL,
tipo ENUM('Funcional', 'No Funcional', 'Restricción') NOT NULL,
estado ENUM('Propuesto', 'Aprovado', 'Implementado', 'Rechazado') NOT NULL,
prioridad ENUM('Baja', 'Media', 'Alta') NOT NULL,
fuente ENUM('Docente', 'Cliente', 'Reglamento') NOT NULL,
fecha_registrio DATE NOT NULL,
proyecto INT NOT NULL,
FOREIGN KEY (proyecto) REFERENCES Proyecto(id)
);
```

Se observa con el comando:

DESC Requisito;

```
+---------------+-----------------------------------------------------------+------+-----+---------+----------------+
| Field         | Type                                                      | Null | Key | Default | Extra          |
+---------------+-----------------------------------------------------------+------+-----+---------+----------------+
| id            | int                                                       | NO   | PRI | NULL    | auto_increment |
| nombre        | varchar(50)                                               | NO   |     | NULL    |                |
| descripcion   | varchar(1000)                                             | NO   |     | NULL    |                |
| tipo          | enum('Funcional','No Funcional','Restricciòn')            | NO   |     | NULL    |                |
| estado        | enum('Propuesto','Aprovado','Implementado','Rechazado')   | NO   |     | NULL    |                |
| prioridad     | enum('Baja','Media','Alta')                               | NO   |     | NULL    |                |
| fuente        | enum('Docente','Cliente','Reglamento')                    | NO   |     | NULL    |                |
| fecha_registro | date                                                     | NO   |     | NULL    |                |
| proyecto      | int                                                       | NO   | MUL | NULL    |                |
+---------------+-----------------------------------------------------------+------+-----+---------+----------------+
9 rows in set (0,00 sec)
```

```
CREATE TABLE Comentario(
    id INT PRIMARY KEY AUTO_INCREMENT,
    contenido VARCHAR(1000) NOT NULL,
    fecha_creacion DATE NOT NULL,
    requisito INT NOT NULL,
    FOREIGN KEY (requisito) REFERENCES Requisito(id)
    );
```

Se observa con el comando:

DESC Comentario;

```
+----------------+---------------+------+-----+---------+----------------+
| Field          | Type          | Null | Key | Default | Extra          |
+----------------+---------------+------+-----+---------+----------------+
| id             | int           | NO   | PRI | NULL    | auto_increment |
| contenido      | varchar(1000) | NO   |     | NULL    |                |
| fecha_creacion | date          | NO   |     | NULL    |                |
| requisito      | int           | NO   | MUL | NULL    |                |
+----------------+---------------+------+-----+---------+----------------+
4 rows in set (0,00 sec)
```

```
CREATE TABLE Relacion_Requisito(
        requisito_origen INT NOT NULL,
        requisito_destino INT NOT NULL,
        FOREIGN KEY (requisito_origen) REFERENCES Requisito(id),
        FOREIGN KEY (requisito_destino) REFERENCES Requisito(id)
        );
```

Se observa con el comando:

DESC Relacion_Requisito;

```
+-------------------+------+------+-----+---------+-------+
| Field             | Type | Null | Key | Default | Extra |
+-------------------+------+------+-----+---------+-------+
| requisito_origen  | int  | NO   | MUL | NULL    |       |
| requisito_destino | int  | NO   | MUL | NULL    |       |
+-------------------+------+------+-----+---------+-------+
2 rows in set (0,01 sec)
```

Después de haber creado las tablas, se prosiguió a insertar valores en cada uno de las tablas correspondientes.
Se empezó con los datos de la tabla Usuario:

INSERT INTO Usuario(nombre, apellido, correo, contraseña, rol) VALUES ('Esmeralda', 'Muro', 'muroesmeralda@unl.edu.ec', 'v8r1yLQkG*', 'Docente');

INSERT INTO Usuario(nombre, apellido, correo, contraseña, rol) VALUES ('Albano', 'Carrera', 'carreraalbano@unl.edu.ec', 'X1KDzIz0^Z', 'Docente');

INSERT INTO Usuario(nombre, apellido, correo, contraseña, rol) VALUES ('Lina', 'Iniesta', 'iniestalina@unl.edu.ec', '+*76pIfeG(', 'Docente');

INSERT INTO Usuario(nombre, apellido, correo, contraseña, rol) VALUES ('Trinidad', 'Costa', 'costatrinidad@unl.edu.ec', '*38Rk^Px*P', 'Estudiante');

INSERT INTO Usuario(nombre, apellido, correo, contraseña, rol) VALUES ('Julia', 'Quero', 'querojulia@unl.edu.ec', '4m1XQ4kp_U', 'Docente');

INSERT INTO Usuario(nombre, apellido, correo, contraseña, rol) VALUES ('Ale', 'Heredia', 'herediaale@unl.edu.ec', 'L3G6C&Lo(V', 'Docente');

INSERT INTO Usuario(nombre, apellido, correo, contraseña, rol) VALUES ('Iván', 'Montoya', 'montoyaivan@unl.edu.ec', 'v9q5ZFTv!&', 'Estudiante');

INSERT INTO Usuario(nombre, apellido, correo, contraseña, rol) VALUES ('Timoteo', 'Costa', 'costatimoteo@unl.edu.ec', '#^1ZlqbtI+', 'Docente');

Se observa los valores con el comando:

SELECT * FROM Usuario;

```
+----+-----------+----------+--------------------------+-------------+------------+
| id | nombre    | apellido | correo                   | contraseña  | rol        |
+----+-----------+----------+--------------------------+-------------+------------+
|  1 | Esmeralda | Muro     | muroesmeralda@unl.edu.ec | v8r1yLQkG*  | Docente    |
|  2 | Albano    | Carrera  | carreraalbano@unl.edu.ec | X1KDzIz0^Z  | Docente    |
|  3 | Lina      | Iniesta  | iniestalina@unl.edu.ec   | +*76pIfeG(  | Docente    |
|  4 | Trinidad  | Costa    | costatrinidad@unl.edu.ec | *38Rk^Px*P  | Estudiante |
|  5 | Julia     | Quero    | querojulia@unl.edu.ec    | 4m1XQ4kp_U  | Docente    |
|  6 | Ale       | Heredia  | herediaale@unl.edu.ec    | L3G6C&Lo(V  | Docente    |
|  7 | Iván      | Montoya  | montoyaivan@unl.edu.ec   | v9q5ZFTv!&  | Estudiante |
|  8 | Timoteo   | Costa    | costatimoteo@unl.edu.ec  | #^1ZlqbtI+  | Docente    |
+----+-----------+----------+--------------------------+-------------+------------+
8 rows in set (0,00 sec)
```

Se continuó con la tabla Proyecto:

INSERT INTO Proyecto(nombre, descripcion, estado, fecha_creacion, usuario) VALUES ('Optimized 3rdgeneration website', 'Sitio web de tercera generación optimizado para velocidad, accesibilidad, experiencia del usuario y compatibilidad multiplataforma, utilizando tecnologías web modernas y mejores prácticas de desarrollo.', 'Activo', '2024-12-23', 3);

INSERT INTO Proyecto(nombre, descripcion, estado, fecha_creacion, usuario) VALUES ('Profound well-modulated firmware', 'Firmware bien estructurado y cuidadosamente modulado, diseñado para ofrecer alto rendimiento, fácil mantenimiento y escalabilidad, garantizando un control preciso y eficiente del hardware.', 'Activo', '2024-12-29', 2);

INSERT INTO Proyecto(nombre, descripcion, estado, fecha_creacion, usuario) VALUES ('Robust systematic superstructure', 'Superestructura robusta y sistemática que proporciona una base sólida, organizada y escalable para el desarrollo e integración de componentes complejos dentro del sistema.', 'Finalizado', '2025-06-08', 5);

INSERT INTO Proyecto(nombre, descripcion, estado, fecha_creacion, usuario) VALUES ('Universal holistic archive', 'Archivo universal y holístico que centraliza, organiza e integra diversos tipos de datos e información, facilitando su acceso, gestión y análisis desde una perspectiva integral.', 'Finalizado', '2025-02-01', 6);

INSERT INTO Proyecto(nombre, descripcion, estado, fecha_creacion, usuario) VALUES ('User-friendly coherent knowledgebase', 'Base de conocimientos coherente y fácil de usar, diseñada para ofrecer información clara, estructurada y accesible, mejorando la comprensión y la experiencia del usuario.', 'Activo', '2025-01-20', 1);

INSERT INTO Proyecto(nombre, descripcion, estado, fecha_creacion, usuario) VALUES ('Pre-emptive zero administration', 'Sistema proactivo con administración cero, capaz de autoconfigurarse, autogestionarse y resolver problemas automáticamente sin intervención del usuario o del administrador.', 'Activo', '2025-05-10', 7);

Se observa la tabla con el comando:

SELECT * FROM Proyecto;

id	nombre	descripcion	estado	fecha_creacion	usuario
1	Optimized 3rdgeneration website	Sitio web de tercera generación optimizado para velocidad, accesibilidad, experiencia del usuario y compatibilidad multiplataforma, utilizando tecnologías web modernas y mejores prácticas de desarrollo.	Activo	2024-12-23	3
2	Profound well-modulated firmware	Firmware bien estructurado y cuidadosamente modulado, diseñado para ofrecer alto rendimiento, fácil mantenimiento y escalabilidad, garantizando un control preciso y eficiente del hardware.	Activo	2024-12-29	2
3	Robust systematic superstructure	Superestructura robusta y sistemática que proporciona una base sólida, organizada y escalable para el desarrollo e integración de componentes complejos dentro del sistema.	Finalizado	2025-06-08	5
4	Universal holistic archive	Archivo universal y holístico que centraliza, organiza e integra diversos tipos de datos e información, facilitando su acceso, gestión y análisis desde una perspectiva integral.	Finalizado	2025-02-01	6
5	User-friendly coherent knowledgebase	Base de conocimientos coherente y fácil de usar, diseñada para ofrecer información clara, estructurada y accesible, mejorando la comprensión y la experiencia del usuario.	Activo	2025-01-20	1
6	Pre-emptive zero administration	Sistema proactivo con administración cero, capaz de autoconfigurarse, autogestionarse y resolver problemas automáticamente sin intervención del usuario o del administrador.	Activo	2025-05-10	7

6 rows in set (0,00 sec)

(el diseño de la tabla se ve afectado debido al tamaño de la descripción)

Se continuó con la tabla Requisito:

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Facilitate user engagement', 'Diseñar funcionalidades interactivas e intuitivas que promuevan la participación activa del usuario en el sitio web.', 'Funcional', 'Aprovado', 'Alta', 'Docente', '2025-02-01', 2);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Implement encryption features', 'Incorporar mecanismos de cifrado para garantizar la confidencialidad e integridad de los datos en el firmware.', 'No Funcional', 'Propuesto', 'Media', 'Reglamento', '2025-01-15', 1);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Improve latency rate', 'Optimizar la arquitectura para reducir la latencia del sistema y mejorar el rendimiento general de respuesta.', 'Funcional', 'Implementado', 'Alta', 'Cliente', '2025-06-20', 3);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Restrict unauthorized access', 'Establecer medidas de control que impidan accesos no autorizados a la superestructura del sistema.', 'Restricción', 'Rechazado', 'Media', 'Docente', '2025-03-25', 5);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Enable data portability', 'Permitir la exportación e importación de datos

entre sistemas de manera eficiente y conforme a normativas.', 'Funcional', 'Propuesto', 'Alta', 'Reglamento', '2025-04-11', 2);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Audit system logs', 'Implementar auditoría de registros del sistema para asegurar trazabilidad y facilitar el diagnóstico.', 'No Funcional', 'Aprovado', 'Baja', 'Docente', '2025-03-12', 6);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Create UI mockups', 'Diseñar prototipos visuales de la interfaz para facilitar validación temprana de usabilidad y estructura.', 'Funcional', 'Aprovado', 'Media', 'Cliente', '2025-05-19', 1);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Monitor performance metrics', 'Implementar monitoreo continuo de métricas clave del sistema para evaluar rendimiento y detectar anomalías.', 'No Funcional', 'Implementado', 'Media', 'Cliente', '2025-06-10', 4);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Automate test coverage', 'Desarrollar mecanismos automáticos de pruebas para validar componentes de forma continua y eficiente.', 'Funcional', 'Propuesto', 'Alta', 'Docente', '2025-06-28', 5);

INSERT INTO Requisito(nombre, descripcion, tipo, estado, prioridad, fuente, fecha_registro, proyecto) VALUES ('Ensure scalability', 'Diseñar la arquitectura del archivo para permitir su crecimiento progresivo sin comprometer el rendimiento.', 'No Funcional', 'Implementado', 'Alta', 'Reglamento', '2025-07-01', 6);

Se observa la tabla con el comando:

SELECT * FROM Requisito;

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro	proyecto
1	Facilitate user engagement	Diseñar funcionalidades interactivas e intuitivas que promuevan la participación activa del usuario en el sitio web.	Funcional	Aprovado	Alta	Docente	2025-02-01	2
2	Implement encryption features	Incorporar mecanismos de cifrado para garantizar la confidencialidad e integridad de los datos en el firmware.	No Funcional	Propuesto	Media	Reglamento	2025-01-15	1
3	Improve latency rate	Optimizar la arquitectura para reducir la latencia del sistema y mejorar el rendimiento general de respuesta.	Funcional	Implementado	Alta	Cliente	2025-06-20	3
4	Restrict unauthorized access	Establecer medidas de control que impidan accesos no autorizados a la superestructura del sistema.	Restricciòn	Rechazado	Media	Docente	2025-03-25	5
5	Enable data portability	Permitir la exportación e importación de datos entre sistemas de manera eficiente y conforme a normativas.	Funcional	Propuesto	Alta	Reglamento	2025-04-11	2
6	Audit system logs	Implementar auditoría de registros del sistema para asegurar trazabilidad y facilitar el diagnóstico.	No Funcional	Aprovado	Baja	Docente	2025-03-12	6
7	Create UI mockups	Diseñar prototipos visuales de la interfaz para facilitar validación temprana de usabilidad y estructura.	Funcional	Aprovado	Media	Cliente	2025-05-19	1
8	Monitor performance metrics	Implementar monitoreo continuo de métricas clave del sistema para evaluar rendimiento y detectar anomalías.	No Funcional	Implementado	Media	Cliente	2025-06-10	4
9	Automate test coverage	Desarrollar mecanismos automáticos de pruebas para validar componentes de forma continua y eficiente.	Funcional	Propuesto	Alta	Docente	2025-06-28	5
10	Ensure scalability	Diseñar la arquitectura del archivo para permitir su crecimiento progresivo sin comprometer el rendimiento.	No Funcional	Implementado	Alta	Reglamento	2025-07-01	6

10 rows in set (0,00 sec)

(el diseño de la tabla se ve afectado por el tamaño de la descripción de los requisitos)

Se continuó con la tabla Comentario:

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Buen enfoque, pero debe afinarse la seguridad.', '2025-05-10', 1);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Considerar uso de herramientas ágiles.', '2025-05-15', 2);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Muy útil para la trazabilidad.', '2025-05-20', 3);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Agregar justificación técnica.', '2025-05-22', 4);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Requiere revisión de viabilidad.', '2025-05-25', 5);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Implementar con control de versiones.', '2025-05-28', 6);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Validar con cliente antes de aprobar.', '2025-06-01', 7);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Excelente funcionalidad.', '2025-06-05', 8);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Cumple con estándares.', '2025-06-10', 9);

INSERT INTO Comentario(contenido, fecha_creacion, requisito) VALUES ('Debe incluir pruebas automatizadas.', '2025-06-12', 10);

Se puede observar la tabla con el comando:

SELECT * FROM Comentario;

```
+----+-------------------------------------------------+----------------+-----------+
| id | contenido                                       | fecha_creacion | requisito |
+----+-------------------------------------------------+----------------+-----------+
|  1 | Buen enfoque, pero debe afinarse la seguridad. | 2025-05-10     |         1 |
|  2 | Considerar uso de herramientas ágiles.          | 2025-05-15     |         2 |
|  3 | Muy útil para la trazabilidad.                  | 2025-05-20     |         3 |
|  4 | Agregar justificación técnica.                  | 2025-05-22     |         4 |
|  5 | Requiere revisión de viabilidad.                | 2025-05-25     |         5 |
|  6 | Implementar con control de versiones.           | 2025-05-28     |         6 |
|  7 | Validar con cliente antes de aprobar.           | 2025-06-01     |         7 |
|  8 | Excelente funcionalidad.                        | 2025-06-05     |         8 |
|  9 | Cumple con estándares.                          | 2025-06-10     |         9 |
| 10 | Debe incluir pruebas automatizadas.             | 2025-06-12     |        10 |
+----+-------------------------------------------------+----------------+-----------+
10 rows in set (0,00 sec)
```

Por último, se terminó con la tabla Relacion_Requisito:

INSERT INTO Relacion_Requisito (requisito_origen, requisito_destino) VALUES (1, 2);

INSERT INTO Relacion_Requisito (requisito_origen, requisito_destino) VALUES (3, 5);

INSERT INTO Relacion_Requisito (requisito_origen, requisito_destino) VALUES (4, 1);

INSERT INTO Relacion_Requisito (requisito_origen, requisito_destino) VALUES (6, 7);

Se puede observar la tabla con el comando:

SELECT * FROM Relacion_Requisito;

```
+-------------------+--------------------+
| requisito_origen | requisito_destino |
+-------------------+--------------------+
|                 1 |                  2 |
|                 3 |                  5 |
|                 4 |                  1 |
|                 6 |                  7 |
+-------------------+--------------------+
4 rows in set (0,01 sec)
```

Consultas y Reportes:

Después de haber llenado las tablas con los datos anteriores, se procedió a realizar diversas consultas en álgebra relacional:

1. **Mostrar todos los requisitos de prioridad alta y estado aprobado**
   σ prioridad == 'Alta' (σ estado == 'Aprovado' (requisito))

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro
2	Registro de usuarios	Formulario de registro con validaciones	Funcional	Aprovado	Alta	Cliente	2025-07-02
8	Notificaciones	Alertas al modificar requisitos	Funcional	Aprovado	Alta	Docente	2025-07-10
4	Base de datos segura	Encriptación de contraseñas	No Funcional	Aprovado	Alta	Docente	2025-07-06

2. **Mostrar los proyectos junto al nombre del responsable**
   σ id == usuario (ρ id ➡ id_proyecto (proyecto) ⋈ ρ nombre ➡ nombre_usuario (usuario))

id	nombre_usuario	apellido	correo	contrasenia	rol	id_proyecto	nombre
6	Andrea	Vega	andrea.vega@unl.edu.ec	1234	Estudiante	4	App de Tareas
3	Juan	Jiménez	juan.jimenez@unl.edu.ec	1234	Estudiante	2	Plataforma Virtual
1	Carlos	Ramírez	carlos.ramirez@unl.edu.ec	1234	Estudiante	6	SGR Ingeniería Requisitos
5	Pedro	Martínez	pedro.martinez@unl.edu.ec	1234	Estudiante	3	Gestor de Biblioteca
7	Sofía	Pineda	sofia.pineda@unl.edu.ec	1234	Estudiante	5	Optimized 3rdgeneration we
1	Carlos	Ramírez	carlos.ramirez@unl.edu.ec	1234	Estudiante	1	Sistema Académico

3. **Mostrar los comentarios realizados sobre los requisitos en estado "Implementado"**
   π descripcion, contenido, estado (σ requisito == id (σ estado == 'Implementado' (ρ id ➡ id_comentario (comentario) ⋈ requisito)))

descripcion	contenido	estado
Descarga de reportes en PDF	Incluir exportación en Excel	Implementado
Descarga de reportes en PDF	Falta formato del PDF	Implementado
Importar CSV para usuarios	Mejorar la interfaz de carga	Implementado

4. **Obtener los proyectos con requisitos funcionales aprobados**
   π nombre_proyecto (σ estado == 'Aprovado' (σ tipo == 'Funcional' (σ id == proyecto (ρ id ➡ id_requisito (requisito) ⋈ π id, nombre_proyecto (ρ nombre ➡ nombre_proyecto (proyecto))))))

nombre_proyecto
SGR Ingeniería Requisitos
Sistema Académico

5. **Obtener los usuarios con proyectos que tienen un requisito de restricción**
   π nombre_usuario, nombre, tipo (σ tipo == 'Restricciòn' (σ id_proyecto == proyecto (π proyecto, tipo (requisito) ⋈ σ id == usuario (ρ id ➡ id_proyecto (proyecto) ⋈ π id, nombre_usuario (ρ nombre ➡ nombre_usuario (usuario))))))

nombre_usuario	nombre	tipo
Andrea	App de Tareas	Restricciòn

6. **Mostrar los requisitos del proyecto "Optimized 3rdgeneration website"**

π nombre, descripcion (σ nombre == 'Optimized 3rdgeneration website' (σ id == proyecto (π id, nombre (proyecto) ⋈ π descripcion, proyecto (requisito))))

nombre	descripcion
Optimized 3rdgeneration website	Importar CSV para usuarios

7. **Obtener los requisitos con su respectivo comentario**
   π nombre, contenido (σ id == requisito (π requisito, contenido (comentario) ⋈ requisito))

nombre	contenido
Registro de usuarios	Agregar validaciones de correo
Inicio de sesión	Agregar validación al login
Accesibilidad	Añadir opciones de accesibilidad
Notificaciones	Agregar roles en notificaciones
Inicio de sesión	Se debe considerar la seguridad
Generación de reportes	Incluir exportación en Excel
Base de datos segura	Revisar estándar OWASP
Carga masiva de datos	Mejorar la interfaz de carga
Generación de reportes	Falta formato del PDF
Soporte multiplataforma	Documentar restricciones

8. **Mostrar los proyectos que no tienen requisitos registrados**
   π nombre, descripcion (σ proyecto== None (proyecto ⋈ π proyecto (requisito)))

nombre	descripcion

9. **Obtener los requisitos que dependen de otros requisitos (requisitos de origen)**
   π nombre (σ id == requisito_origen (relacion_requisito ⋈ requisito))

nombre
Generación de reportes
Inicio de sesión
Base de datos segura
Carga masiva de datos

10. **Mostrar los requisitos con estado "propuesto"**
    σ estado == 'Propuesto' (requisito)

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro
1	Inicio de sesión	Permitir el inicio de sesión de usuarios	Funcional	Propuesto	Alta	Docente	2025-07-01
7	Auditoría del sistema	Registro de actividad	No Funcional	Propuesto	Baja	Cliente	2025-07-09

11. **Obtener los nombres de los usuarios que han creado proyectos finalizados**
    π nombre, nombre_proyecto, estado (σ usuario == id_usuario (ρ id ➡ id_usuario (usuario) ⋈ σ estado == 'Finalizado' (ρ nombre ➡ nombre_proyecto (proyecto))))

nombre	nombre_proyecto	estado
Pedro	Gestor de Biblioteca	Finalizado
Sofía	Optimized 3rdgeneration website	Finalizado

12. **Mostrar los requisitos funcionales de prioridad alta y estado implementado**
    σ estado == 'Implementado' (σ prioridad == 'Alta' (σ tipo == 'Funcional' (requisito)))

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro
6	Carga masiva de datos	Importar CSV para usuarios	Funcional	Implementado	Alta	Docente	2025-07-08

13. Obtener los requisitos de los que dependen otros requisitos (requisitos de destino)

π nombre (σ id == requisito_destino (relacion_requisito ⋈ requisito))

nombre
Auditoría del sistema
Inicio de sesión
Registro de usuarios
Soporte multiplataforma

14. Mostrar los requisitos que provienen del cliente y están aprobados

σ estado == 'Aprovado' (σ fuente == 'Cliente' (requisito))

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro
2	Registro de usuarios	Formulario de registro con validaciones	Funcional	Aprovado	Alta	Cliente	2025-07-02

15. Mostrar los proyectos que tienen al menos un requisito rechazado

π nombre, descripcion (σ proyecto == id (proyecto ⋈ π proyecto (σ estado == 'Rechazado' (requisito))))

nombre	descripcion
App de Tareas	Aplicación para gestionar tareas
SGR Ingeniería Requisitos	Sistema de Gestión de Requisitos

16. Obtener los proyectos con requisitos dependientes

π nombre_proyecto, nombre (σ id == proyecto (ρ nombre ➡ nombre_proyecto (proyecto) ⋈ π nombre, proyecto (σ id == requisito_origen (relacion_requisito ⋈ (requisito)))))

nombre_proyecto	nombre
Gestor de Biblioteca	Base de datos segura
Sistema Académico	Inicio de sesión
Optimized 3rdgeneration website	Carga masiva de datos
Plataforma Virtual	Generación de reportes

17. Mostrar los proyectos con requisitos no funcionales propuestos

σ estado == 'Propuesto' (σ tipo == 'No Funcional' (requisito))

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro
7	Auditoría del sistema	Registro de actividad	No Funcional	Propuesto	Baja	Cliente	2025-07-09

18. **Mostrar el nombre de los usuarios y el estado de los proyectos que tienen requisitos dependientes**

    π nombre_usuario, nombre_proyecto, estado (σ id_proyecto == proyecto (π id, nombre, proyecto (σ id == requisito_origen (relacion_requisito ⋈ (requisito))) ⋈ σ usuario == id_usuario (π id_usuario, nombre_usuario (ρ id ➡ id_usuario (ρ nombre ➡ nombre_usuario (usuario))) ⋈ π usuario, id_proyecto, nombre_proyecto, estado (ρ nombre ➡ nombre_proyecto (ρ id ➡ id_proyecto (proyecto))))))

nombre_usuario	nombre_proyecto	estado
Juan	Plataforma Virtual	Activo
Pedro	Gestor de Biblioteca	Finalizado
Sofía	Optimized 3rdgeneration website	Finalizado
Carlos	Sistema Académico	Activo

19. **Obtener los nombres de los usuarios que tienen un proyecto con requisitos aprobados**

    π nombre, nombre_proyecto (σ estado_requisito == 'Aprovado' (σ id == proyecto (π estado_requisito, proyecto (ρ estado ➡ estado_requisito (requisito)) ⋈ σ usuario == id_usuario (ρ nombre ➡ nombre_proyecto (proyecto) ⋈ (π id_usuario, nombre (ρ id ➡ id_usuario (usuario)))))))

nombre	nombre_proyecto
Carlos	Sistema Académico
Carlos	SGR Ingeniería Requisitos
Pedro	Gestor de Biblioteca

20. **Mostrar todos los usuarios y sus proyectos activos con requisitos funcionales**

    π nombre, nombre_proyecto, estado, nombre_requisito (σ proyecto == id (π nombre_requisito, proyecto (ρ nombre ➡ nombre_requisito (requisito)) ⋈ π id, nombre, nombre_proyecto, estado (σ estado == 'Activo' (σ id_usuario == usuario (ρ nombre ➡ nombre_proyecto (proyecto) ⋈ π id_usuario, nombre (ρ id ➡ id_usuario (usuario)))))))

nombre	nombre_proyecto	estado	nombre_requisito
Carlos	SGR Ingeniería Requisitos	Activo	Auditoría del sistema
Carlos	SGR Ingeniería Requisitos	Activo	API externa
Juan	Plataforma Virtual	Activo	Generación de reportes
Carlos	SGR Ingeniería Requisitos	Activo	Accesibilidad
Carlos	SGR Ingeniería Requisitos	Activo	Notificaciones
Carlos	Sistema Académico	Activo	Inicio de sesión
Carlos	Sistema Académico	Activo	Registro de usuarios
Andrea	App de Tareas	Activo	Soporte multiplataforma

Views:

Después de realizar las consultas en álgebra relacional, se prosiguió a convertirlas en SQL y guardarlas como VIEWS (vistas):

1. **Mostrar todos los requisitos de prioridad alta y estado aprobado**
   CREATE VIEW RequisitosAltaAprobados AS SELECT * FROM Requisito WHERE prioridad = 'Alta' AND estado = 'Aprovado';

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro	proyecto
1	Facilitate user engagement	Diseñar funcionalidades interactivas e intuitivas que promuevan la participación activa del usuario en el sitio web.	Funcional	Aprovado	Alta	Docente	2025-02-01	2

1 row in set (0,01 sec)

2. **Mostrar los proyectos junto al nombre del responsable**
   CREATE VIEW ProyectosConResponsables AS SELECT P.id, P.nombre, P.descripcion, P.estado, U.nombre AS nombre_usuario, U.apellido FROM Proyecto P JOIN Usuario U ON P.usuario = U.id;

id	nombre	descripcion	estado	nombre_usuario	apellido
1	Optimized 3rdgeneration website	Sitio web de tercera generación optimizado para velocidad, accesibilidad, experiencia del usuario y compatibilidad multiplataforma, utilizando tecnologías web modernas y mejores prácticas de desarrollo.	Activo	Lina	Iniesta
2	Profound well-modulated firmware	Firmware bien estructurado y cuidadosamente modulado, diseñado para ofrecer alto rendimiento, fácil mantenimiento y escalabilidad, garantizando un control preciso y eficiente del hardware.	Activo	Albano	Carrera
3	Robust systematic superstructure	Superestructura robusta y sistemática que proporciona una base sólida, organizada y escalable para el desarrollo e integración de componentes complejos dentro del sistema.	Finalizado	Julia	Quero
4	Universal holistic archive	Archivo universal y holístico que centraliza, organiza e integra diversos tipos de datos e información, facilitando su acceso, gestión y análisis desde una perspectiva integral.	Finalizado	Ale	Heredia
5	User-friendly coherent knowledgebase	Base de conocimientos coherente y fácil de usar, diseñada para ofrecer información clara, estructurada y accesible, mejorando la comprensión y la experiencia del usuario.	Activo	Esmeralda	Muro
6	Pre-emptive zero administration	Sistema proactivo con administración cero, capaz de autoconfigurarse, autogestionarse y resolver problemas automáticamente sin intervención del usuario o del administrador.	Activo	Iván	Montoya

6 rows in set (0,00 sec)

3. **Mostrar los comentarios realizados sobre los requisitos en estado “Implementado”**
   CREATE VIEW ComentariosRequisitosImplementados AS SELECT R.descripcion, C.contenido, R.estado FROM Comentario C JOIN Requisito R ON C.requisito = R.id WHERE R.estado = 'Implementado';

descripcion	contenido	estado
Optimizar la arquitectura para reducir la latencia del sistema y mejorar el rendimiento general de respuesta.	Muy útil para la trazabilidad.	Implementado
Implementar monitoreo continuo de métricas clave del sistema para evaluar rendimiento y detectar anomalías.	Excelente funcionalidad.	Implementado
Diseñar la arquitectura del archivo para permitir su crecimiento progresivo sin comprometer el rendimiento.	Debe incluir pruebas automatizadas.	Implementado

3 rows in set (0,00 sec)

4. **Obtener los proyectos con requisitos funcionales aprobados**

CREATE VIEW ProyectosConRequisitosFuncionalesAprobados AS SELECT DISTINCT P.nombre AS nombre_proyecto FROM Proyecto P JOIN Requisito R ON R.proyecto = P.id WHERE R.tipo = 'Funcional' AND R.estado = 'Aprovado';

```
+-----------------------------------+
| nombre_proyecto                   |
+-----------------------------------+
| Profound well-modulated firmware  |
| Optimized 3rdgeneration website   |
+-----------------------------------+
2 rows in set (0,01 sec)
```

5. **Obtener los usuarios con proyectos que tienen un requisito de restricción**
   CREATE VIEW UsuariosConRequisitosRestriccion AS SELECT DISTINCT U.nombre AS nombre_usuario, P.nombre AS nombre_proyecto, R.tipo FROM Usuario U JOIN Proyecto P ON P.usuario = U.id JOIN Requisito R ON R.proyecto = P.id WHERE R.tipo = 'Restricción';

```
+----------------+---------------------------------------+--------------+
| nombre_usuario | nombre_proyecto                       | tipo         |
+----------------+---------------------------------------+--------------+
| Esmeralda      | User-friendly coherent knowledgebase | Restricciòn  |
+----------------+---------------------------------------+--------------+
1 row in set (0,00 sec)
```

6. **Mostrar los requisitos del proyecto “Optimized 3rdgeneration website”**
   CREATE VIEW RequisitosDeProyectoOptimizado AS SELECT R.nombre, R.descripcion FROM Requisito R JOIN Proyecto P ON R.proyecto = P.id WHERE P.nombre = 'Optimized 3rdgeneration website';

```
+---------------------------+-----------------------------------------------------------------------------------------------------------+
| nombre                    | descripcion                                                                                               |
+---------------------------+-----------------------------------------------------------------------------------------------------------+
| Implement encryption features | Incorporar mecanismos de cifrado para garantizar la confidencialidad e integridad de los datos en el firmware. |
| Create UI mockups         | Diseñar prototipos visuales de la interfaz para facilitar validación temprana de usabilidad y estructura.      |
+---------------------------+-----------------------------------------------------------------------------------------------------------+
2 rows in set (0,00 sec)
```

7. **Obtener los requisitos con su respectivo comentario**
   CREATE VIEW RequisitosConComentarios AS SELECT R.nombre, C.contenido FROM Requisito R JOIN Comentario C ON C.requisito = R.id;

```
+------------------------------+-----------------------------------------------+
| nombre                       | contenido                                     |
+------------------------------+-----------------------------------------------+
| Facilitate user engagement   | Buen enfoque, pero debe afinarse la seguridad. |
| Implement encryption features | Considerar uso de herramientas ágiles.        |
| Improve latency rate         | Muy útil para la trazabilidad.                |
| Restrict unauthorized access | Agregar justificación técnica.                |
| Enable data portability      | Requiere revisión de viabilidad.              |
| Audit system logs            | Implementar con control de versiones.         |
| Create UI mockups            | Validar con cliente antes de aprobar.         |
| Monitor performance metrics  | Excelente funcionalidad.                      |
| Automate test coverage       | Cumple con estándares.                        |
| Ensure scalability           | Debe incluir pruebas automatizadas.           |
+------------------------------+-----------------------------------------------+
10 rows in set (0,00 sec)
```

8. **Mostrar los proyectos que no tienen requisitos registrados**
   CREATE VIEW ProyectosSinRequisitos AS SELECT P.nombre, P.descripcion FROM Proyecto P LEFT JOIN Requisito R ON R.proyecto = P.id WHERE R.id IS NULL;

```
Empty set (0,00 sec)
```

9. **Obtener los requisitos que dependen de otros requisitos (requisitos de origen)**
   CREATE VIEW RequisitosOrigen AS SELECT R.nombre FROM Requisito R JOIN Relacion_Requisito RR ON R.id = RR.requisito_origen;

```
+------------------------------+
| nombre                       |
+------------------------------+
| Facilitate user engagement   |
| Improve latency rate         |
| Restrict unauthorized access |
| Audit system logs            |
+------------------------------+
4 rows in set (0,00 sec)
```

10. **Mostrar los requisitos con estado “propuesto”**
    CREATE VIEW RequisitosPropuestos AS SELECT * FROM Requisito WHERE estado = 'Propuesto';

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro	proyecto
2	Implement encryption features	Incorporar mecanismos de cifrado para garantizar la confidencialidad e integridad de los datos en el firmware.	No Funcional	Propuesto	Media	Reglamento	2025-01-15	1
5	Enable data portability	Permitir la exportación e importación de datos entre sistemas de manera eficiente y conforme a normativas.	Funcional	Propuesto	Alta	Reglamento	2025-04-11	2
9	Automate test coverage	Desarrollar mecanismos automáticos de pruebas para validar componentes de forma continua y eficiente.	Funcional	Propuesto	Alta	Docente	2025-06-28	5

3 rows in set (0,00 sec)

11. **Obtener los nombres de los usuarios que han creado proyectos finalizados**
    CREATE VIEW UsuariosConProyectosFinalizados AS SELECT U.nombre, P.nombre AS nombre_proyecto, P.estado FROM Usuario U JOIN Proyecto P ON P.usuario = U.id WHERE P.estado = 'Finalizado';

nombre	nombre_proyecto	estado
Julia	Robust systematic superstructure	Finalizado
Ale	Universal holistic archive	Finalizado

2 rows in set (0,00 sec)

12. **Mostrar los requisitos funcionales de prioridad alta y estado implementado**
    CREATE VIEW RequisitosFuncionalesAltaImplementados AS SELECT * FROM Requisito WHERE tipo = 'Funcional' AND prioridad = 'Alta' AND estado = 'Implementado';

id	nombre	descripcion	tipo	estado	prioridad	fuente	fecha_registro	proyecto
3	Improve latency rate	Optimizar la arquitectura para reducir la latencia del sistema y mejorar el rendimiento general de respuesta.	Funcional	Implementado	Alta	Cliente	2025-06-20	3

1 row in set (0,00 sec)

13. **Obtener los requisitos de los que dependen otros requisitos (requisitos de destino)**
    CREATE VIEW RequisitosDestino AS SELECT R.nombre FROM Requisito R JOIN Relacion_Requisito RR ON R.id = RR.requisito_destino;

```
+---------------------------------+
| nombre                          |
+---------------------------------+
| Facilitate user engagement      |
| Implement encryption features   |
| Enable data portability         |
| Create UI mockups               |
+---------------------------------+
4 rows in set (0,00 sec)
```

14. **Mostrar los requisitos que provienen del cliente y están aprobados**
    CREATE VIEW RequisitosClienteAprobados AS SELECT * FROM Requisito WHERE fuente = 'Cliente' AND estado = 'Aprovado';

```
+----+------------------+-------------------------------------------------------------------------------------------------------+-----------+--
--------+-----------+---------+---------------+----------+
| id | nombre           | descripcion                                                                                           | tipo      | e
stado   | prioridad | fuente  | fecha_registro | proyecto |
+----+------------------+-------------------------------------------------------------------------------------------------------+-----------+--
--------+-----------+---------+---------------+----------+
|  7 | Create UI mockups | Diseñar prototipos visuales de la interfaz para facilitar validación temprana de usabilidad y estructura.   | Funcional | A
provado | Media     | Cliente | 2025-05-19     |        1 |
+----+------------------+-------------------------------------------------------------------------------------------------------+-----------+--
--------+-----------+---------+---------------+----------+
1 row in set (0,00 sec)
```

15. **Mostrar los proyectos que tienen al menos un requisito rechazado**
    CREATE VIEW ProyectosConRequisitosRechazados AS SELECT DISTINCT P.nombre, P.descripcion FROM Proyecto P JOIN Requisito R ON R.proyecto = P.id WHERE R.estado = 'Rechazado';

```
+-----------------------------------+-------------------------------------------------------------------------------------------------------
-------------------------------------------------------------+
| nombre                            | descripcion
                                                             |
+-----------------------------------+-------------------------------------------------------------------------------------------------------
-------------------------------------------------------------+
| User-friendly coherent knowledgebase | Base de conocimientos coherente y fácil de usar, diseñada para ofrecer información clara, estructurada y acce
sible, mejorando la comprensión y la experiencia del usuario.     |
+-----------------------------------+-------------------------------------------------------------------------------------------------------
-------------------------------------------------------------+
1 row in set (0,00 sec)
```

16. **Obtener los proyectos con requisitos dependientes**
    CREATE VIEW ProyectosConRequisitosDependientes AS SELECT DISTINCT P.nombre AS nombre_proyecto, R.nombre FROM Proyecto P JOIN Requisito R ON R.proyecto = P.id JOIN Relacion_Requisito RR ON R.id = RR.requisito_origen;

nombre_proyecto	nombre
Profound well-modulated firmware	Facilitate user engagement
Robust systematic superstructure	Improve latency rate
User-friendly coherent knowledgebase	Restrict unauthorized access
Pre-emptive zero administration	Audit system logs

4 rows in set (0,01 sec)

17. **Mostrar los proyectos con requisitos no funcionales propuestos**
    CREATE VIEW ProyectosConRequisitosNoFuncionalesPropuestos AS SELECT DISTINCT P * FROM Proyecto P JOIN Requisito R ON R.proyecto = P.id WHERE R.tipo = 'No Funcional' AND R.estado = 'Propuesto';

id	nombre	descripcion	estado	fecha_creacion	usuario
1	Optimized 3rdgeneration website	Sitio web de tercera generación optimizado para velocidad, accesibilidad, experiencia del usuario y compatibilidad multiplataforma, utilizando tecnologías web modernas y mejores prácticas de desarrollo.	Activo	2024-12-23	3

1 row in set (0,00 sec)

18. **Mostrar el nombre de los usuarios y el estado de los proyectos que tienen requisitos dependientes**
    CREATE VIEW UsuariosConProyectosDependientes AS SELECT DISTINCT U.nombre AS nombre_usuario, P.nombre AS nombre_proyecto, P.estado FROM Usuario U JOIN Proyecto P ON P.usuario = U.id JOIN Requisito R ON R.proyecto = P.id JOIN Relacion_Requisito RR ON RR.requisito_origen = R.id;

nombre_usuario	nombre_proyecto	estado
Albano	Profound well-modulated firmware	Activo
Julia	Robust systematic superstructure	Finalizado
Esmeralda	User-friendly coherent knowledgebase	Activo
Iván	Pre-emptive zero administration	Activo

4 rows in set (0,00 sec)

19. **Obtener los nombres de los usuarios que tienen un proyecto con requisitos aprobados**
    CREATE VIEW UsuariosConRequisitosAprobados AS SELECT DISTINCT U.nombre, P.nombre AS nombre_proyecto FROM Usuario U JOIN Proyecto P ON P.usuario = U.id JOIN Requisito R ON R.proyecto = P.id WHERE R.estado = 'Aprovado';

```
+--------+-----------------------------------+
| nombre | nombre_proyecto                   |
+--------+-----------------------------------+
| Albano | Profound well-modulated firmware  |
| Iván   | Pre-emptive zero administration   |
| Lina   | Optimized 3rdgeneration website   |
+--------+-----------------------------------+
3 rows in set (0,00 sec)
```

20. **Mostrar todos los usuarios y sus proyectos activos con requisitos funcionales**
    CREATE VIEW UsuariosProyectosActivosConFuncionales AS SELECT DISTINCT U.nombre, P.nombre AS nombre_proyecto, P.estado, R.nombre AS nombre_requisito FROM Usuario U JOIN Proyecto P ON P.usuario = U.id AND P.estado = 'Activo' JOIN Requisito R ON R.proyecto = P.id AND R.tipo = 'Funcional';

```
+-----------+-----------------------------------+--------+----------------------------+
| nombre    | nombre_proyecto                   | estado | nombre_requisito           |
+-----------+-----------------------------------+--------+----------------------------+
| Albano    | Profound well-modulated firmware  | Activo | Facilitate user engagement |
| Albano    | Profound well-modulated firmware  | Activo | Enable data portability    |
| Lina      | Optimized 3rdgeneration website   | Activo | Create UI mockups          |
| Esmeralda | User-friendly coherent knowledgebase | Activo | Automate test coverage  |
+-----------+-----------------------------------+--------+----------------------------+
4 rows in set (0,00 sec)
```

RESULTADOS OBTENIDOS

Los anexos de la práctica junto con este documento se encontrarán en el siguiente repositorio de GitHub:

https://github.com/Darwin-J5/Practica_Preprofesional

CONCLUSIONES Y RECOMENDACIONES

Durante estas prácticas preprofesionales se ha logrado consolidar de manera gratificante los conocimientos teóricos adquiridos en la asignatura de base de datos, fortaleciendo áreas como el diseño de las bases de datos, el desarrollo de consultas SQL, el análisis de requerimientos de información real y la adaptabilidad al entorno de la institución.

Como recomendaciones se tiene la posibilidad de manejar este tipo de prácticas y trabajos con bases de datos NoSQL y la documentación de diseños y técnicas diferentes para facilitar el mantenimiento de futuros sistemas

Estudiante

Docente Supervisor