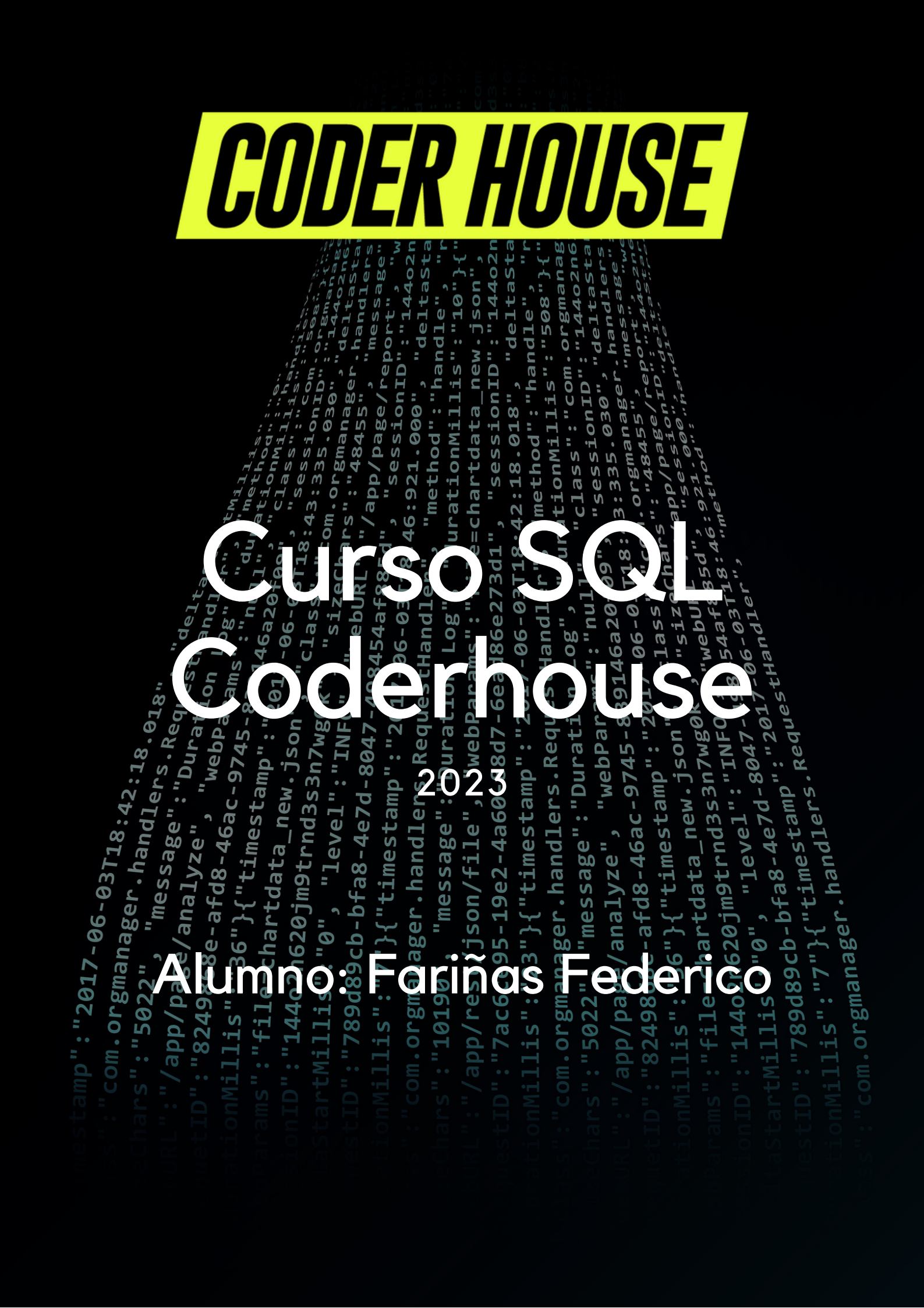


Curso SQL Coderhouse

2023

Alumno: Fariñas Federico

1. OBJETO

En este proyecto desarrollado en MySQL 8.0, se busca modelar el esquema de datos de un consultorio médico: desde el gestionamiento de sus pacientes y doctores, hasta la asignación de turnos y seguimiento de tratamientos.

Asimismo, a través de las distintas herramientas y funciones que nos brinda el Workbench, facilitarle a los operadores la navegación a través del esquema.

2. ENLACES

GitHub:

https://github.com/FariFede/SQL_43425

Draw.io:

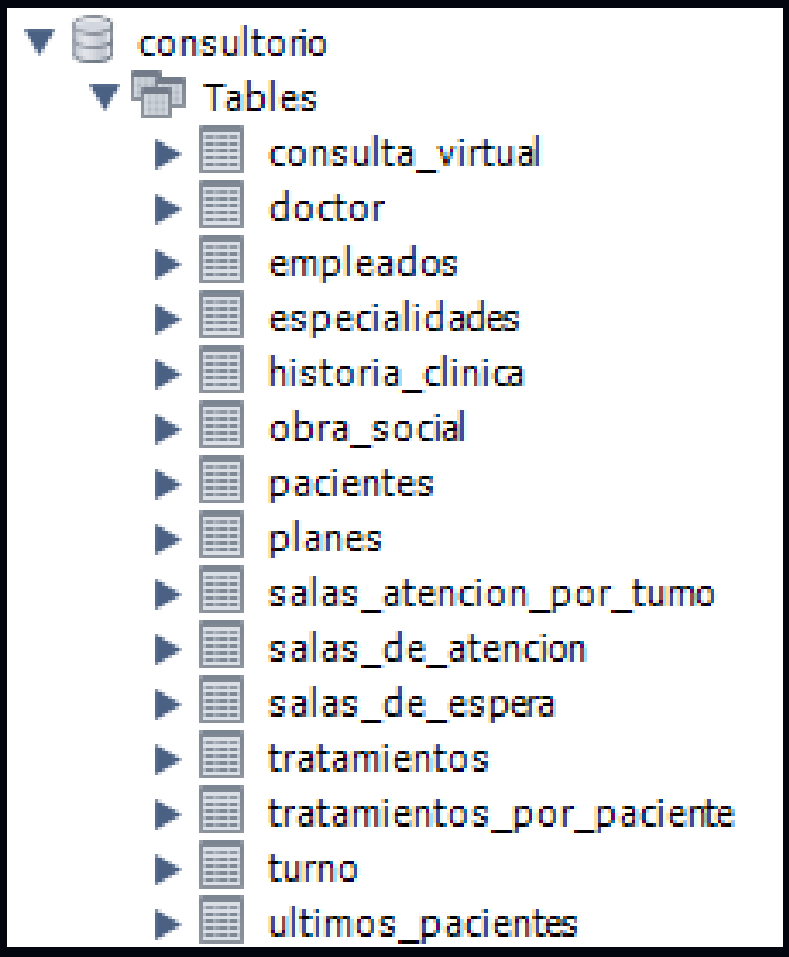
https://app.diagrams.net/#G1PiUo6tUlFMSgOsvIwJ7r3XaDyLJJAWRr

3. ESQUEMA Y TABLAS

Nuestro esquema cuenta con 15 tablas.

La selección de las mismas se basa en la necesidad de gestionar criteriosamente todo el proceso de atención de un paciente: desde su alta en el sistema, su asignación de turnos o consultas virtuales, su ubicación en la sala de espera y su vínculo con el doctor que tomará su caso y gestionará su tratamiento.

3. ESQUEMA Y TABLAS



4. DIAGRAMA ENTIDAD-RELACIÓN

En el siguiente diagrama se muestra el tipo de relación que toman entre sí las distintas columnas de nuestras tablas.

Es de suma importancia realizarlo antes de comenzar los proyectos, puesto que nos brindará una descripción visual y estructurada de nuestras tablas.

Debemos mantenerlo normalizado y actualizado para ayudar a nuestros colegas a comprender y mantener nuestra base de datos.

4. DIAGRAMA ENTIDAD-RELACIÓN

En el siguiente diagrama se muestra el tipo de relación que toman entre sí las distintas columnas de nuestras tablas.

Es de suma importancia realizarlo antes de comenzar los proyectos, puesto que nos brindará una descripción visual y estructurada de nuestras tablas antes de que nos pongamos a programarlas.

Debemos mantenerlo normalizado y actualizado para ayudar a nuestros colegas a comprender y mantener el esquema.

4. DIAGRAMA ENTIDAD-RELACIÓN

Enlace al diagrama:

https://drive.google.com/file/d/1PiUo6tUlFMSgOsvIwJ7r3XaDyLJJAWRr/view?usp=sharing

5. VISTAS

Las vistas nos permiten procesar combinaciones en cadena de consultas: de forma tal de obtener el nexo entre dos campos producto de varias consultas, en tan solo una.

Así simplificamos nuestra relación entre tablas, ahorrando tiempos de implementación y mantenimiento.

Además las vistas pueden reutilizarse, lo cual aumenta el rendimiento de la base de datos.

5. VISTAS

Contamos con 10 vistas, entre ellas:

- Vista de doctores por especialidad.
- Vista de doctores por turno.
- Vista de pacientes por historia clínica.
- Vista de pacientes por obra social.
- Vista de tratamientos por paciente.

Citamos un ejemplo para consultarlas:

```
SELECT nombre_paciente FROM
vista_nombre_paciente_por_historia WHERE
historia_id = '';
```

6. FUNCIONES

Las funciones nos ayudan a realizar cálculos y algoritmos entre distintos campos de nuestra tabla.

En nuestra aplicación, la utilizamos para simplificar el cálculo de la bonificación según obra social en cada tratamiento, o bien la ubicación de determinada sala de atención dentro del edificio.

Ejemplo para llamado de función:

```
SELECT calcular_costo_final(costo,
bonificacion) AS costo_final
FROM tratamientos, tratamientos_por_paciente
```

7. STORED PROCEDURES

Sin duda, los Stored Procedures han sido un foco central en la aplicación.

Además de combinar los beneficios de las vistas y las funciones (muchas veces integrándolas dentro del mismo SP), son utilizados para realizar comprobaciones de datos necesarias previamente a insertar en nuestras tablas.

También nos permiten verificar la consistencia de datos.

7. STORED PROCEDURES

Ejemplos de Stored Procedures:

- Insertar turno presencial.
- Insertar turno virtual.
- Procedimiento para agregar un paciente.
- Procedimiento para agregar un doctor.
- Elegir la sala de atención por turno.
- Ordenar datos de una tabla.

Ejemplos de SPs:

```
CALL OrdenarDatos ('obra_social',
'nombre_obra_social', 'ASC');
```

8. TRIGGERS

Los triggers poseen un gran potencial dado a que nos permiten realizar verificaciones automáticas, ya que a diferencia de los SPs, son eventos desencadenantes y no invocaciones manuales.

Se emplean para restringir, actualizar, automatizar, entre otras funciones.

Aunque en nuestro desarrollo les hemos dado poco protagonismo, suelen ser empleados con bastante frecuencia (a veces hasta en exceso).

9. PERMISOS

Con intenciones de mostrar la funcionalidad de los permisos, se crearon dos usuarios:

- Usuario para lectura únicamente.
- Operador de base de datos.

El lector sólo puede utilizar SELECT para encuestar las tablas, mientras que el operador puede utilizar SELECT, INSERT y UPDATE para manipular datos.

10. COMENTARIOS

Como resumen del proyecto, utilizo este apartado para aclarar que lo que se expone es una demostración de las capacidades que nos brinda la plataforma MySQL.

Es posible trabajar con mayor cantidad de tablas, datos, funciones automatizadas e incluso seguridad y respaldo.

Será cuestión de seguir avanzando en la incursión dentro de la tecnología, que con la Industria 5.0, seguirá requiriendo bases de datos completas y eficientes.