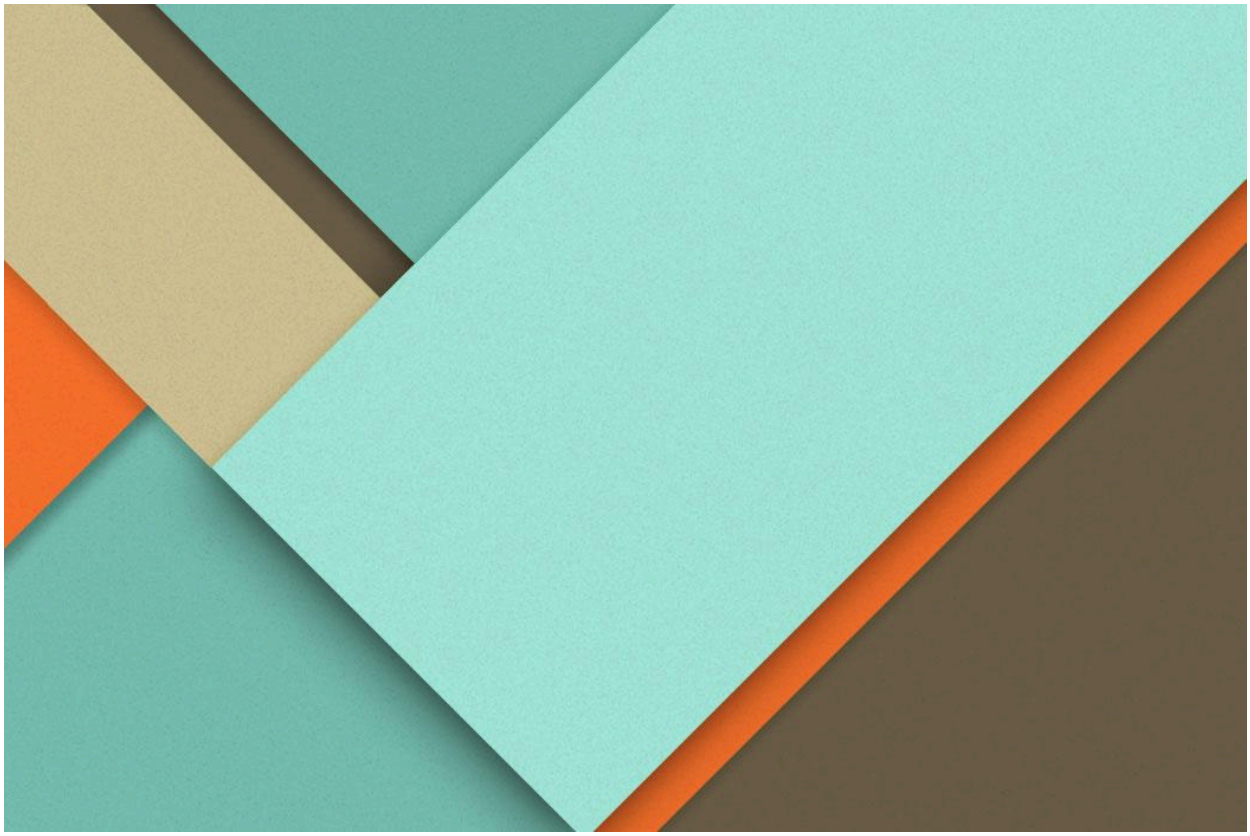


Stormforge Kingdoms

Juego MMORPG medieval

—

Fabrizio Fontanarrosa
Coderhouse - SQL
Noviembre 2025
Entrega final

Introducción

Este documento forma parte de la entrega final del curso de Bases de datos SQL de Coderhouse. En el presente proyecto me dediqué a diseñar el sistema de bases de datos para un modelo de videojuego multijugador masivo online de rol (MMORPG) ambientado en un mundo de fantasía medieval. Se modelaron sistemas de autentificación (username, email, password en texto plano), personajes, clases, estadísticas de combate, objetos, inventario, misiones, localizaciones, mascotas, entre otras entidades del mundo.

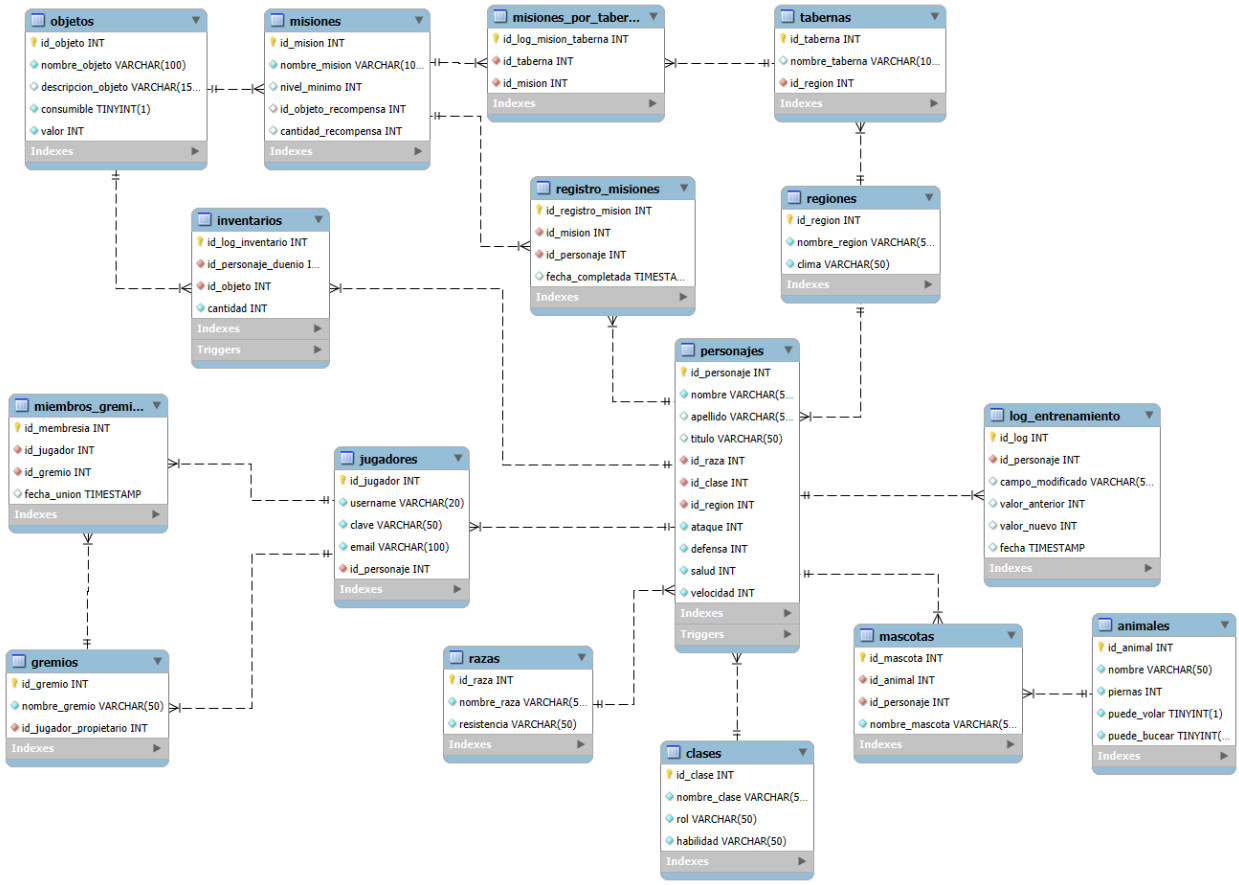
Objetivo

El objetivo del proyecto es poner en práctica todo lo aprendido a lo largo del curso, desde la creación básica de tablas, claves primarias y foráneas, hasta la implementación de procedimientos almacenados, transacciones y herramientas de análisis de datos. También aprender a documentar un sistema y poder justificar y defender las decisiones tomadas en el diseño del mismo.

Situación problemática

Partimos de la situación de querer implementar un sistema de bases de datos para un videojuego MMORPG llamado *Stormforge Kingdoms*. Como cualquier otro sistema de software, hay mucha información de índoles distintas que se necesita almacenar, tanto interna del juego (como los objetos y eventos que van sucediendo), como externa (datos de los usuarios). Y al ser un videojuego de carácter online, toda esta información debe guardarse en los servidores de la empresa desarrolladora (o servidores en la nube contratados por la misma), para garantizar el acceso óptimo de jugadores de todas partes del mundo a estos datos en tiempo real.

Diagrama Entidad-Relación



Listado de tablas

Descripción de cada tabla con sus respectivos campos y claves.

personajes

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_personaje	INT	X			X
nombre	VARCHAR(50)				
apellido	VARCHAR(50)			X	
titulo	VARCHAR(50)			X	
id_raza	INT		X		

id_clase	INT		X		
id_region	INT		X		
ataque	INT				
defensa	INT				
salud	INT				
velocidad	INT				

razas

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_raza	INT	X			X
nombre_raza	VARCHAR(50)				
resistencia	VARCHAR(50)			X	

clases

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_clase	INT	X			X
nombre_clase	VARCHAR(50)				
rol	VARCHAR(50)			X	
habilidad	VARCHAR(50)				

regiones

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_region	INT	X			X
nombre_region	VARCHAR(50)				
clima	VARCHAR(50)				

mascotas

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_mascota	INT	X			X
id_animal	INT		X		
id_personaje	INT		X		X
nombre_mascota	VARCHAR(50)				

animales

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_animal	INT	X			X
nombre	VARCHAR(50)				
piernas	INT				
puede_volar	BOOL				
puede_bucear	BOOL				

log_entrenamiento

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_log	INT	X			X
id_personaje	INT		X		
campo_modificado	VARCHAR(50)				
valor_anterior	INT			X	
valor_nuevo	INT			X	
fecha	TIMESTAMP			X	

jugadores

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_jugador	INT	X			X
username	INT				X
clave	VARCHAR(20)				
email	VARCHAR(50)				X
id_personaje	INT		X		

misiones

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_mision	INT	X			X
nombre_mision	VARCHAR(100)				
nivel_minimo	INT			X	
id_objeto_recompensa	INT		X	X	
cantidad_recompensa	INT			X	

registro_misiones

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_registro_mision	INT	X			X
id_mision	INT		X		
id_personaje	INT		X		
fecha_completada	TIMESTAMP				

objetos

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_objeto	INT	X			X

nombre_objeto	VARCHAR(100)				
descripcion_objeto	VARCHAR(150)			X	
consumible	BOOLEAN				
valor	INT				

inventarios

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_log_inventario	INT	X			X
id_personaje_duenio	INT		X		
id_objeto	INT		X		
cantidad	INT				

tabernas

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_taberna	INT	X			X
nombre_taberna	VARCHAR(50)				
id_region	INT		X		

misiones_por_taberna

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_log_mision_taberna	INT	X			X
id_taberna	INT		X		
id_mision	INT		X		

gremios

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_gremio	INT	X			X

nombre_gremio	VARCHAR(50)				X
id_jugador_propietario	INT		X		

miembros_gremios

Campo	Tipo de dato	PK	FK	NULL	UNIQUE
id_membresia	INT	X			X
id_jugador	INT		X		
id_gremio	INT		X		
fecha_union	TIMESTAMP				

Acerca de los objetos de la base de datos

A continuación se verán detalles de los distintos objetos de la base de datos: vistas, funciones, stored procedures, triggers y usuarios.

Vistas

detalle_personajes

Permite visualizar más fácilmente la información de los personajes, incluyendo su clase, región y raza. Muestra datos de las tablas *personajes*, *razas*, *regiones* y *clases*.

Campo	Tipo de dato	Tabla	NULL	UNIQUE
id_personaje	INT	personajes		X
nombre	VARCHAR(50)	personajes		
apellido	VARCHAR(50)	personajes	X	
nombre_raza	VARCHAR(50)	razas		
nombre_region	VARCHAR(50)	regiones		
titulo	VARCHAR(50)	personajes	X	
nombre_clase	VARCHAR(50)	clases		

rol	VARCHAR(50)	clases		

regiones_magos

Sirve para ver un listado de las regiones de donde provienen los personajes que son magos en la base de datos. Muestra datos de la tabla *regiones*, trabaja con registros de las tablas *personajes* y *clases*.

Campo	Tipo de dato	Tabla	NULL	UNIQUE
id_region	INT	regiones		X
nombre_region	VARCHAR(50)	regiones		

tiene_dragon

Permite ver una lista de nombres y apellidos de los personajes que tienen un dragón como mascota. Muestra campos de *personajes*, trabaja también con *mascotas* y *animales*.

Campo	Tipo de dato	Tabla	NULL	UNIQUE
id_personaje	INT	personajes		X
nombre	VARCHAR(50)	personajes		
apellido	VARCHAR(50)	personajes	X	

mascotas_voladoras

Sirve para ver un listado de las mascotas que pueden volar, y sus respectivos dueños. Muestra datos de las tablas *personajes* y *mascotas,* trabaja también con *animales*.

Campo	Tipo de dato	Tabla	NULL	UNIQUE
id_mascota	INT	mascotas		X
nombre_mascota	VARCHAR(50)	mascotas		
nombre (AS nombre_dueño)	VARCHAR(50)	personajes	X	

apellido (AS apellido_dueño)	VARCHAR(50)	personajes		

daños_con_montura

Permite ver una lista de los personajes cuyo rol es de Daño y tienen una mascota, trabaja con las tablas *personajes, mascotas* y *animales*

Campo	Tipo de dato	Tabla	NULL	UNIQUE
id_personaje	INT	personajes		X
nombre	VARCHAR(50)	personajes		
apellido	VARCHAR(50)	personajes	X	
nombre_clase	VARCHAR(50)	clases		
nombre_mascota	VARCHAR(50)	mascotas		
nombre (AS animal)	VARCHAR(50)	animales		

clasificacion_personajes

Muestra una lista de los personajes junto a su *puntuación* (promedio de las cuatro estadísticas), y ordena los registros en función de ese campo, de manera descendente. Trabaja con las tablas *personajes* y *clases*, y utiliza la función *promedio_estadisticas(p_id_personaje INT)*.

Campo	Tipo de dato	Tabla	NULL	UNIQUE
id_personaje	INT	personajes		X
nombre	VARCHAR(50)	personajes		
apellido	VARCHAR(50)	personajes	X	
nombre_clase	VARCHAR(50)	clases		
promedio_estadisticas(id_personaje) (AS puntuacion)	VARCHAR(50)	N/A (función)		

inventarios_extendido

Muestra nombre del objeto y cantidad de cada objeto que posee cada personaje, además del nombre del personaje. Forma más cómoda de leer los inventarios de los personajes.

Campo	Tipo de dato	Tabla	NULL	UNIQUE
nombre	VARCHAR(50)	personajes		
nombre_objeto	VARCHAR(100)	objetos		
cantidad	INT	inventarios		

Funciones

promedio_estadisticas(p_id_personaje INT)

Recibe la primary key de un registro de la tabla *personajes* y devuelve el promedio de sus cuatro estadísticas: Ataque, Defensa, Salud y Velocidad.

cantidad_clase(p_id_clase INT)

Recibe la primary key de un registro de la tabla *clases* y devuelve la cantidad de personajes que hay de esa clase.

Stored Procedures

entrenar_ataque(p_id_personaje INT)

Recibe la primary key de un registro de la tabla *personajes* y aumenta su estadística de ataque en 5.

entrenar_defensa(p_id_personaje INT)

Recibe la primary key de un registro de la tabla *personajes* y aumenta su estadística de defensa en 5.

entrenar_salud(p_id_personaje INT)

Recibe la primary key de un registro de la tabla *personajes* y aumenta su estadística de salud en 5.

entrenar_velocidad(p_id_personaje INT)

Recibe la primary key de un registro de la tabla *personajes* y aumenta su estadística de velocidad en 5.

entrenar_fortaleza(p_id_personaje INT)

Recibe la primary key de un registro de la tabla *personajes* y aumenta su estadística más alta en 5.

entrenar_debilidad(p_id_personaje INT)

Recibe la primary key de un registro de la tabla *personajes* y aumenta su estadística más baja en 5.

ver_inventario(p_id_personaje INT)

Recibe la primary key de un registro de la tabla *personajes* y muestra los nombres de todos los objetos que posee en su inventario, con sus respectivas cantidades.

ver_misiones(p_id_taberna INT)

Recibe la primary key de un registro de la tabla *tabernas* y muestra los nombres de todas las misiones disponibles en esa taberna.

enviar_objetos(p_id_log_inventario INT, p_id_personaje_destino INT, p_cantidad INT)

Recibe la primary key de un registro de la tabla *inventarios*, la primary key de un registro de la tabla *personajes*, y un entero representando una cantidad. Envía la cantidad especificada del objeto del registro de la tabla inventarios al personaje de destino. Si se solicita una cantidad mayor a la que el personaje remitente tiene disponible, no se realiza el envío.

Triggers

log_cambio_estadistica

Cada vez que se modifica una estadística de un personaje, se logea la modificación en la tabla *log_entrenamiento*.

estadistica_max

Cada vez que se modifica una estadística de un personaje, se verifica que esta no haya superado el número 100, puesto que es el máximo que se puede obtener por estadística.

limpiar_registros_vacios_inventario

Cada vez que se modifica un registro de la tabla inventarios, se revisa el campo cantidad del mismo, y elimina el registro si la cantidad es 0.

Usuarios de la base de datos

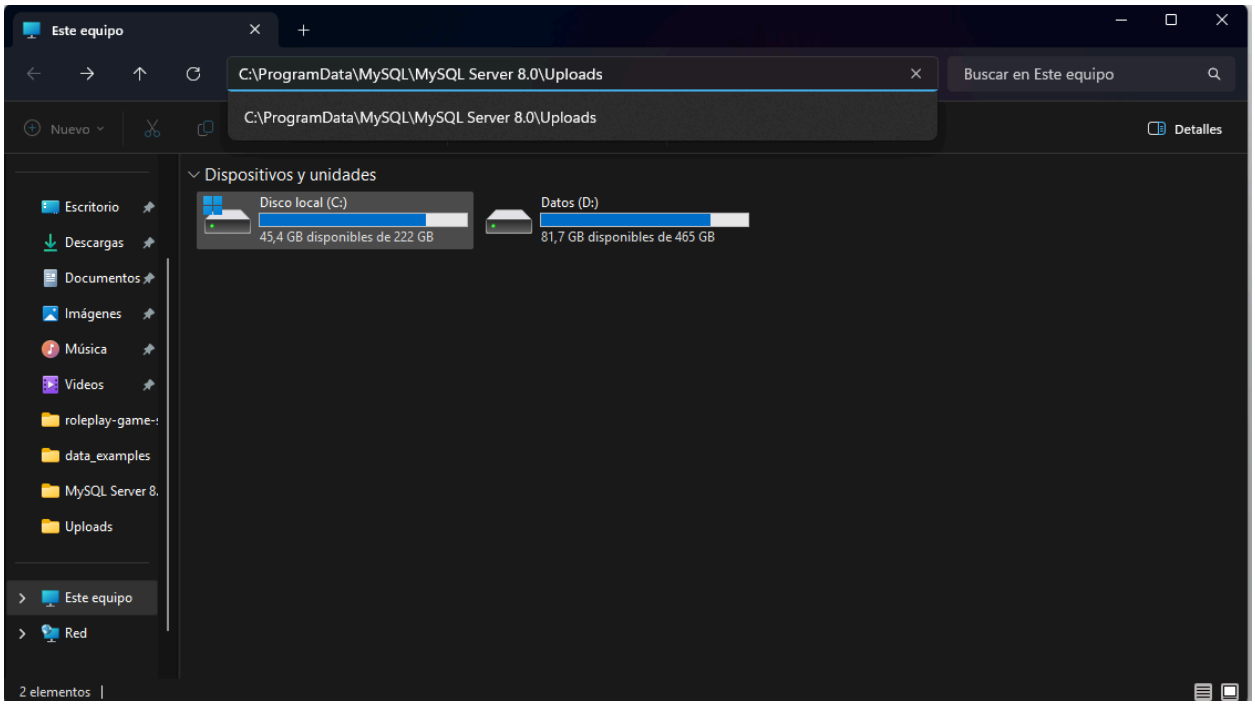
Username	Password	Accede desde	Permisos
visitante	visitante	localhost	Sólo permisos de lectura en toda la base de datos
moderador	moderador!321321	localhost	Permisos de lectura, inserción y actualización de datos en toda la DB
administrador	administrador!98323	localhost	Todos los permisos en toda la base de datos

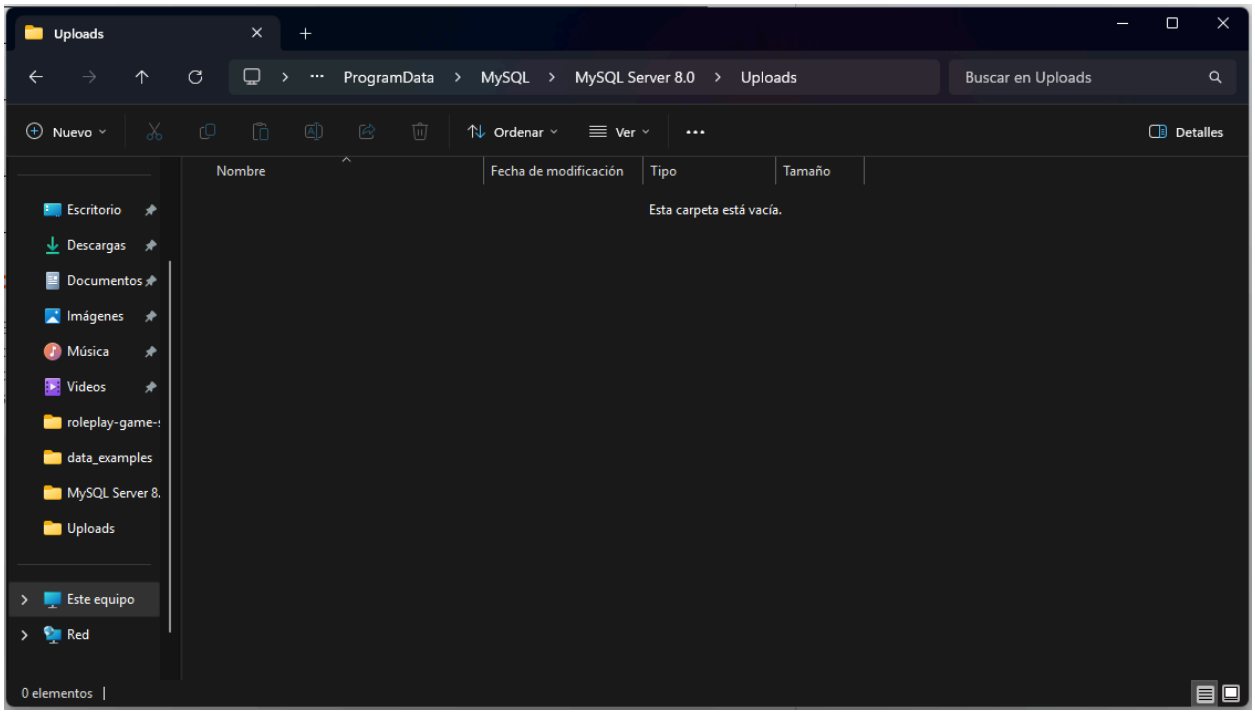
Procedimiento de importación de datos

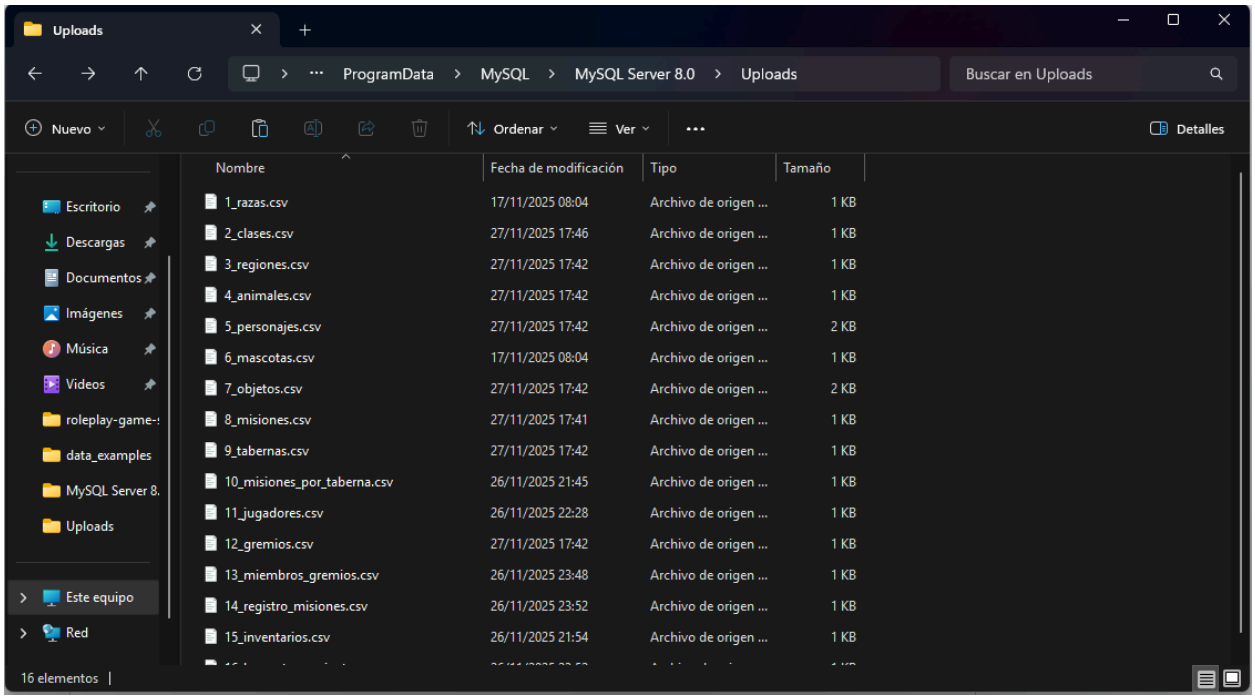
El proceso que se va a explicar es para usuarios que estén trabajando con Windows. En la carpeta data_examples se encuentran seis archivos .csv enumerados. Para importar correctamente los datos, hay que abrir una ventana del explorador de Windows y pegar la siguiente ruta en la barra de direcciones:

C:\ProgramData\MySQL\MySQL Server 8.0\Uploads

Luego, en la carpeta que se nos abre, hay que copiar los 16 archivos .csv de la carpeta *data_examples*:


Este equipo
C:\ProgramData\MySQL\MySQL Server 8.0\Uploads
Buscar en Este equipo
Nuevo
Detalles
Dispositivos y unidades
Disco local (C:)
45,4 GB disponibles de 222 GB
Datos (D:)
81,7 GB disponibles de 465 GB
Escritorio
Descargas
Documentos
Imágenes
Música
Videos
roleplay-game-
data_examples
MySQL Server 8.
Uploads
Este equipo
Red
2 elementos


Uploads
ProgramData
MySQL
MySQL Server 8.0
Uploads
Buscar en Uploads
Nuevo
Ordenar
Ver
Detalles
Nombre
Fecha de modificación
Tipo
Tamaño
Esta carpeta está vacía.
Escritorio
Descargas
Documentos
Imágenes
Música
Videos
roleplay-game-
data_examples
MySQL Server 8.
Uploads
Este equipo
Red
0 elementos

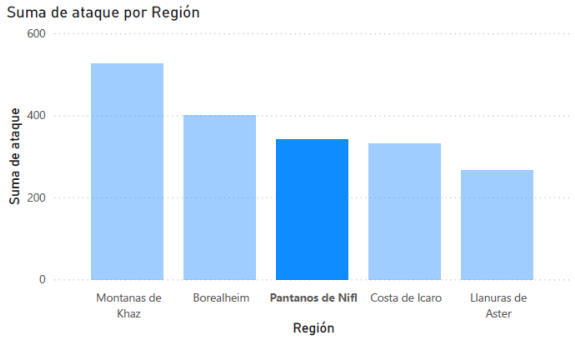


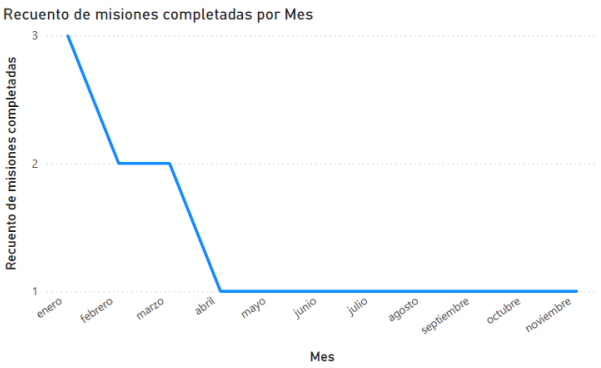
Puede ser que la dirección varíe si tiene una versión distinta de MySQL Server, pero siempre termina en *Uploads*.

Una vez hecho esto, abrimos MySQL Workbench con normalidad. Luego de ejecutar los scripts *tablas.sql* y *script.sql*, ejecutamos *importacion_datos.sql*, y el estado de ejecución de las solicitudes debería mostrarse en verde en la parte inferior de la interfaz de Workbench, indicando que los datos se importaron correctamente.

Informes de Business Intelligence

Mediante el uso del software Power BI, se hizo un análisis de los datos almacenados en las tablas del sistema, y se encontraron dos patrones interesantes: si sumamos la estadística de ataque de todos los personajes de cada región, encontraremos que Montañas de Khaz es la que más poder de ataque total posee. Esto indica altas posibilidades de que sus nativos se están preparando para una guerra. Por otro lado, yendo para el lado más relacionado con los jugadores reales, podemos observar que el número de misiones completadas por los jugadores en cada mes ha llegado a su máximo de 3 en el mes de enero de 2025, y a partir de ahí ha bajado hasta 1 misión por mes, que es el número que mantenemos desde abril del corriente año.





Herramientas y tecnologías utilizadas

Este proyecto se desarrolló con apoyo en las siguientes tecnologías y herramientas:

- MySQL para Windows como motor de base de datos
- MySQL Workbench como entorno gráfico para interactuar con la base de datos, y para diseñar el diagrama Entidad-Relación
- Visual Studio Code como editor de texto para los scripts SQL y los archivos csv
- Git y Github para el control de versiones del sistema
- Power BI para el análisis de los datos y los gráficos