Modelos conceptual y lógico para el proyecto

José Ignacio Botero Osorio

Instructor

Lyam Acosta Forero

Aprendiz

SENA – Servicio Nacional de Aprendizaje

Tecnología en análisis y Desarrollo de Software

3118526

Bogotá D.C.

Contenido

Modelos conceptual y lógico para el proyecto

En esta sección revisaremos diferentes modelos lógicos y conceptuales para nuestro videojuego, basados en modelos entidad-relación para una base de datos aquí consideraremos varias cosas como el modelo conceptual, modelos lógicos, bases de datos y la especificación del análisis junto con un diagrama visual.

Modelo Conceptual

El modelo conceptual representa entidades principales del sistema y sus relaciones de alto nivel, sin considerar aspectos técnicos de implementación.

1. **Usuario/Jugador:** Entidad Central que representa a los usuarios del sistema.
2. **Partida:** Sesión de juego individual de un usuario.
3. **Nivel:** Etapas o fases del juego.
4. **Puntaje:** Sistema de puntuación o logros.
5. **Configuración:** Preferencias y ajustes del usuario.
6. **Estadística:** Datos de rendimiento o progreso.

Modelo Lógico

El modelo lógico describe de forma detallada la estructura de las tablas, los campos, los tipos de datos y las relaciones necesarias para la implementación en una base de datos.

La primera tabla, **USUARIOS**, contiene la información de los jugadores o participantes. Sus campos son:

- **id_usuario (PK)**: un número entero que se incrementa automáticamente y actúa como identificador único.
- **nombre_usuario**: una cadena de texto de hasta 50 caracteres, con restricción de unicidad.
- **email**: dirección de correo electrónico del usuario, con un máximo de 100 caracteres y también única.
- **fecha_registro**: almacena la fecha y hora en que se registró el usuario.

- **estado_activo**: un valor booleano que indica si el usuario se encuentra activo o no.

La segunda tabla, **PARTIDAS**, almacena la información referente a las sesiones de juego. Incluye los siguientes campos:

- **id_partida (PK)**: identificador único de cada partida, generado automáticamente.
- **id_usuario (FK)**: clave foránea que hace referencia al identificador del usuario que jugó la partida.
- **fecha_inicio** y **fecha_fin**: registran el momento de inicio y finalización de cada partida.
- **puntaje_total**: almacena la puntuación obtenida en la sesión.
- **nivel_alcanzado**: indica el nivel más alto alcanzado por el jugador en esa partida.

Finalmente, la tabla **NIVELES** define las características de cada nivel disponible en el juego. Está compuesta por los siguientes campos:

- **id_nivel (PK)**: identificador único del nivel.
- **nombre_nivel**: nombre descriptivo del nivel, con un máximo de 100 caracteres.
- **dificultad**: establece el nivel de dificultad, que puede ser “Fácil”, “Medio” o “Difícil”.
- **tiempo_limite**: indica el tiempo máximo permitido para completar el nivel.
- **puntaje_objetivo**: define el puntaje necesario para superar el nivel.

Especificaciones del Análisis

Requisitos de datos

1. **Almacenamiento de usuarios:** Gestión de cuentas y perfiles.
2. **Registro de partidas:** Historial completo de sesión de juego.
3. **Sistema de Niveles:** Progresión estructurada del juego.
4. **Estadísticas en tiempo real:** Monitoreo de Rendimiento**.**
5. **Configuración personalizada:** Preferencias del usuario.

Integridad de datos

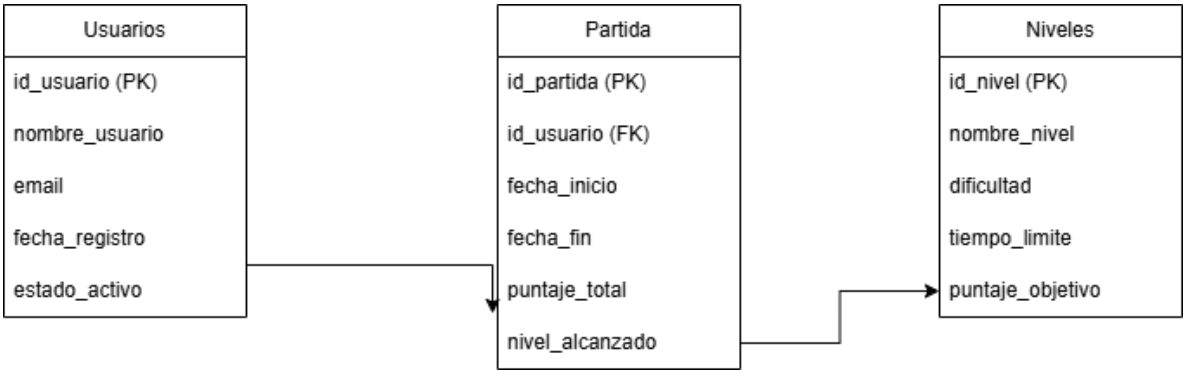
1. Claves primarias únicas para cada entidad.
2. Claves foráneas para mantener relación.
3. Restricción de integridad referencial.
4. Validación de datos de entrada.
5. Índice para optimización de consultas.

Tipos de Bases de Datos

1. **MySQL:** Base de datos relacional robusta, que es ideal para aplicaciones web con alta demanda.
2. **PostgreSQL:** Base de datos avanzadas con soporte para JSON y características modernas.
3. **MongoDB:** Base de datos NoSQL orientada a documentos, flexible para datos no estructurados.
4. **SQLite:** Base de datos ligera embebida, perfecta para aplicaciones Standalone.

Para nuestro videojuego la más recomendada es SQL debido a su amplia compatibilidad con Frameworks web, excelente rendimiento para aplicaciones de mediana escala, comunidad activa y documentación extensa, Herramientas de administración maduras y por último un costo/beneficio favorable

Diagrama



PK (Primary Key)

FK (Foreing Key)

Unique (Valor único de Tabla)

Podemos concluir que el modelo conceptual y lógico del proyecto propuesto es una base sólida para el desarrollo de nuestro videojuego y con eso aseguramos:

1. **Escalabilidad:** Estructura que puede crecer con el proyecto.
2. **Integridad:** Datos consistentes y relaciones bien definida.
3. **Rendimiento:** Optimización para consultas frecuentes.
4. **Mantenibilidad:** Estructura clara y documentada.
5. **Flexibilidad:** Capacidad de adaptación a nuevos requerimientos.