Final - EsTim

Hecho por: Centurión, Franco; Idañez, Lucía; Lamas, Chabela

Enunciado

Final DDS 20230225 - EsTim.pdf

Dudas

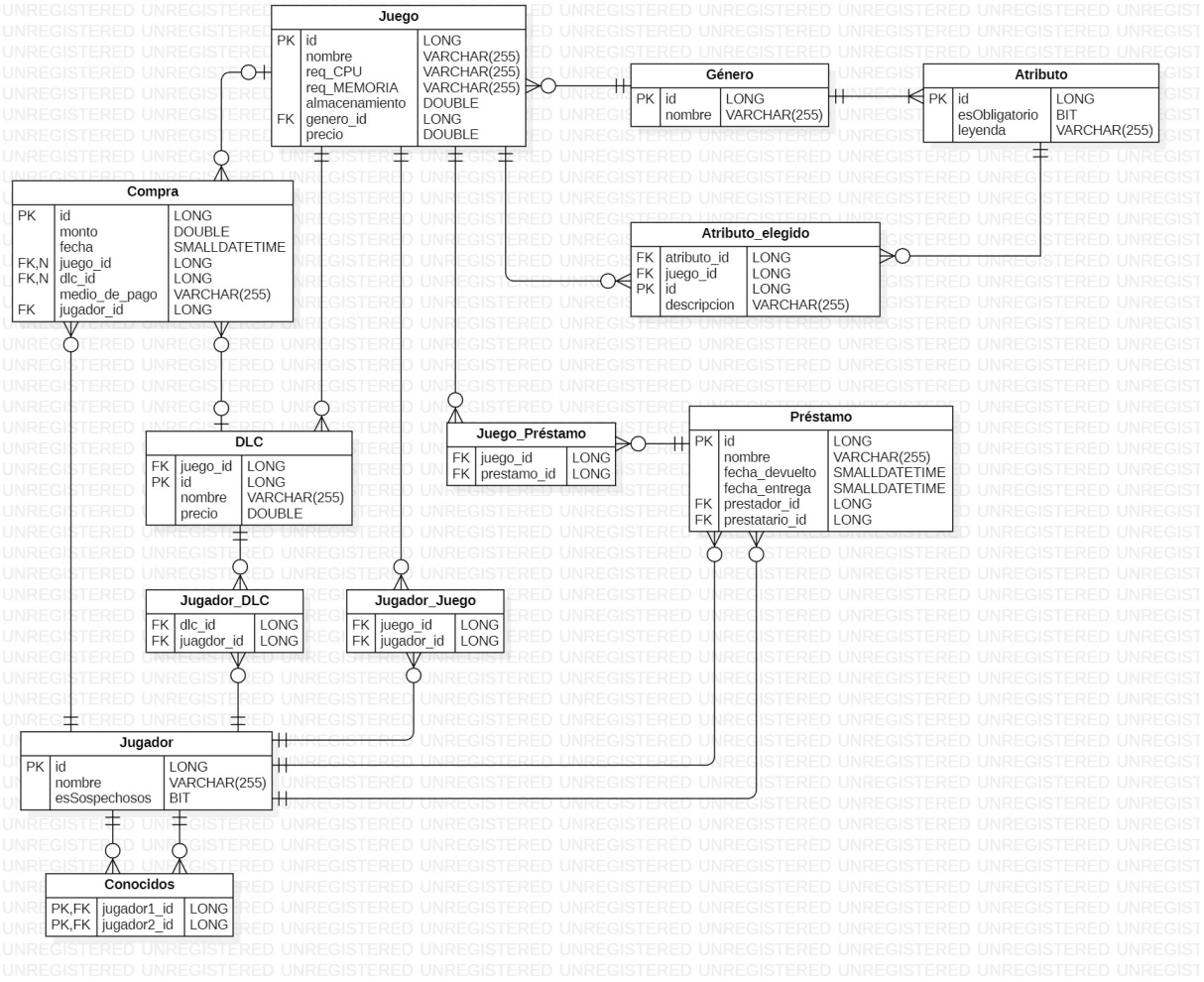
- [ ] Duda con la respuesta brindada a la pregunta 1 de arquitectura

Notas

-

Modelado de datos

Entidades principales



Decisiones de diseño

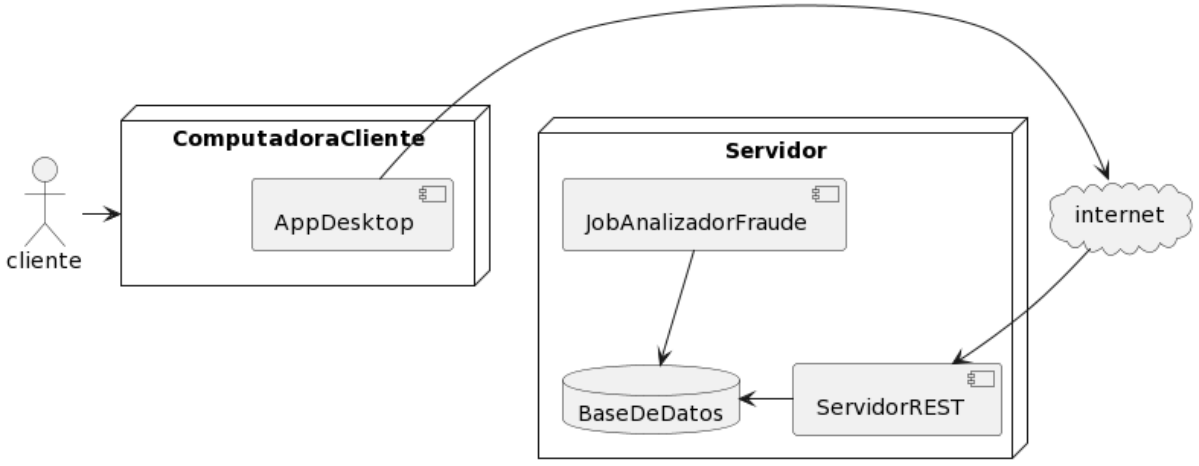
- En la parte de **Genero**, **Atributo_Elegido** y **Atributo** decidimos tomar el modelo de cuestionario de texto libre, esto quiere decir que **Atributo_Elegido** va a tener la referencia a su **Atributo** precargado por los administradores de acuerdo al género y un campo de texto/respuesta libre.
- En **Compra** decidimos realizar dos columnas, una para referenciar al Juego y otra al DLC, de esta manera uno u otro quedará nulo al momento de la compra.
- El medio de pago lo consideramos como un VARCHAR(255), no los predefinimos en la plataforma.
- Para mantener la consistencia de datos, decidimos hacer una desnormalización en **Compra**, ya que un Juego puede cambiar su precio a lo largo del tiempo. A su vez, dado que se trata de un proceso de compra e involucra dinero, marcamos la trazabilidad de la misma con las fechas.
- Trazabilidad en los prestamos, decidimos agregar al prestador (quien da el préstamo) y al prestatario (quien lo recibe).
- Como un jugador puede tener varios conocidos (que son jugadores) y un jugador puede estar entre los conocidos de varios jugadores, hacemos una tabla intermedia.

Impedance Mismatches

Tipo datos objetos	Tipo datos relacional
String	VARCHAR(255)
Bool	BIT
LocalDate	SMALLDATETIME

- Para las fechas usamos un converter que pase del tipo de dato LocalDate de JAVA a SMALLDATETIME del motor de base de datos.

Arquitectura



> 1. Los usuarios actualmente solo pueden visualizar el catálogo de juegos comprados y
> de forma online, lo cual imposibilita que de forma offline puedan visualizar los juegos comprados y jugarlos. Para solucionar este problema nos recomendaron utilizar una CDN , ¿Qué opinión tiene sobre esto? En caso de no estar de acuerdo plantear una solución al problema.

NOTA: *CDN. Content delivery network, red de distribución de contenidos*

RTA: Por lo que investigue una CDN no resolveria en caso de estar offline ya que igualmente se va a tener que ir a buscar a ese nuevo servidor y si uno esta offline no va a poder hacerlo, lo que pense fue unicamente almacenar en la appdesktop los juegos adquiridos por el usuario con el path al .exe para que pueda jugarlos, teniendo un cliente pesado con esa info no tendria que hacer ninguna request a nuestro servidor

> 2. La plataforma actualmente está al límite de su capacidad de atención de pedidos
> HTTP. Ya se ha mejorado varias veces la cantidad de memoria y núcleos del servidor, pero sigue siendo insuficiente. Además, se desea maximizar la

> disponibilidad de la plataforma. ¿Qué
> alternativas tenemos para tratar de resolver este problema?

RTA: Dado que ya se han realizado mejoras en la cantidad de memoria y núcleos del servidor y aún se encuentra al límite de su capacidad, podría ser más efectivo adoptar un enfoque de escalabilidad horizontal. Esto implica agregar más servidores a la plataforma para distribuir la carga y mejorar la capacidad de manejar pedidos HTTP mediante un balanceador de carga delante de los servidores para la redistribución de las solicitudes de HTTP, maximizando la disponibilidad de la plataforma.

> 3. Por último la plataforma los días de eventos importantes las peticiones de compras suelen demorar más del tiempo de espera máximo configurado en los clientes, logramos analizar que el cuello de botella se da al momento de facturar, ya que estas peticiones demoran mucho y en algunos casos incluso fallan. ¿Qué alternativas tenemos disponibles para tratar de resolver este problema?

RTA: La solución a este problema la podemos encarar desde el punto de vista del sincronismo/asincronismo. En este caso, al estar trabajando de forma sincrónica, se está generando el cuello de botella por el tiempo que tardan en procesarse las facturaciones.

Para resolver esta cuestión, recomendamos implementar una cola de mensajería en donde se procesen las facturaciones de forma asíncrona. De esta manera, al realizarse una compra, se cargará la cola con la información necesaria para realizar la factura y eventualmente el consumidor procesará la misma. Por consiguiente, se logra mejorar el rendimiento, la fiabilidad y la escalabilidad de nuestro sistema.

Otra posibilidad es persistir la información de facturación en la base de datos e implementar una Cron Task que ejecute un componente Facturador, por ejemplo, el cual recupera los registros de la base con la información de las facturas y los procesa de forma asíncrona. Cuando termine de procesarlos, marcará los elementos como procesados (podemos agregar un boolean, además de la información de facturación)