Segunda Entrega Proyecto

[Enlace del entregable anterior.]

Coderhouse SQL

Alumno: Jun

Criterios de la entrega:

ENTREGA DEL PROYECTO FINAL

Segunda entrega

Se debe entregar

- ✓ Listado de Vistas más una descripción detallada, su objetivo, y qué tablas las componen.
- ✓ Listado de Funciones que incluyan una descripción detallada, el objetivo para la cual fueron creadas y qué datos o tablas manipulan y/o son implementadas.
- ✓ Listado de Stored Procedures con una descripción detallada, qué objetivo o beneficio aportan al proyecto, y las tablas que lo componen y/o tablas con las que interactúa.
- ✓ Un archivo .sql que contenga:
  - ✓ Script de inserción de datos en las bases.
  - ✓ Si se insertan datos mediante importación, agregar el paso a paso de éste en el DOC PDF más los archivos con el contenido a importar, en el formato que corresponda.
  - ✓ Script de creación de Vistas, Funciones, Stored Procedures y Triggers.

CODERHOUSE

ENTREGA DEL PROYECTO FINAL

Segunda entrega

Formato

- ✓ Documento PDF con el nombre "Entrega2 + Apellido".

Sugerencias

- ✓ Activar la posibilidad de realizar comentarios en el archivo que subis como Entrega. Debe incluir el contenido del documento PDF de la primera entrega parcial, además de lo requerido en esta segunda entrega.

Podrás encontrar más información en la Guía de actividades hacia el Proyecto Final

CODERHOUSE

Listado de Objetos, Descripción...

Vista 1: Player_Items_View

- Objetivo: Esta vista permite visualizar los tres ítems virtuales más caros del juego, mostrando sus detalles como nombre, descripción, precio y categoría.
- Tablas Manipuladas:
    - Players: Para obtener información sobre los jugadores (ID y nombre).
    - Virtual_Items: Para obtener detalles sobre los ítems que poseen los jugadores.
- Detalles: Sobre el campo ID_Player de las tablas Players y Virtual_Items se realiza un left join para determinar los ítems que ha comprado el jugador.
- Datos: Selecciona el nombre del ítem, su descripción, precio y categoría, así como el nombre del jugador y su ID.

Vista 2: Item_Economic_Activity_View

- Objetivo: Esta vista presenta un resumen de la actividad económica de los ítems virtuales, incluyendo el número de transacciones y el total de ingresos generados por cada ítem.
- Tablas Manipuladas:
    - Virtual_Items: Para obtener los nombres de los ítems.
    - Financial_Transactions: Para contar el número de transacciones y sumar los montos generados por cada ítem.
- Detalles: Se realiza un left join de los campos ID_Item y ID_Player para obtener una lista relacionada entre los ítems virtuales y los jugadores.
- Datos: Muestra el nombre del ítem, la cantidad de transacciones y el total de ingresos, ordenando por ingresos totales.

Función 1: Get_Total_Spending

- Objetivo: Esta función calcula el total gastado por un jugador en todas sus transacciones.
- Tablas Manipuladas:
    - Financial_Transactions: Para sumar los montos de todas las transacciones del jugador especificado.
- Detalles: Se realiza la suma con la función SUM(), habiendo filtrado anteriormente por el ID del jugador para obtener todas las transacciones.
- Datos: DECIMAL. Retorna un valor decimal que representa el total gastado por el jugador.

Función 2: Get_Average_Spending_Status

- Objetivo: Esta función calcula el gasto promedio de un jugador y lo compara con el gasto promedio de todos los jugadores, retornando un mensaje de estado.
- Tablas Manipuladas:
    - Financial_Transactions: Para calcular el gasto promedio del jugador y del total de jugadores.
- Detalles: Se calculan los gastos promedios generales, y del jugador seleccionado, para a continuación, realizar una comparativa IF de estos valores.
- Datos: VARCHAR. Retorna un mensaje que indica si el jugador gasta más, menos o igual que el promedio general.

Procedimiento 1: Update_Support_Ticket_Status

- Objetivo: Este procedimiento permite actualizar el estado de un ticket de soporte y registrar la última fecha de actualización.
- Tablas Manipuladas:
    - Support_and_Customer_Service: Para actualizar el estado del ticket específico.
- Detalles: Se realiza un SET después de haber realizado un filtrado con WHERE.
- Datos: Se actualizan el campo Ticket_Status, y el campo Last_Updated_Date.

Procedimiento 2: Register_Transaction

- Objetivo: Este procedimiento registra una nueva transacción en el sistema. Antes de registrar esta nueva transacción, se valida la existencia del jugador y que el monto de la transacción no se trate de un valor negativo.
- Tablas Manipuladas:
    - Players: Para verificar si el jugador existe en el sistema.
    - Financial_Transactions: Para insertar la nueva transacción.
- Detalles: Se realiza un count(*) para verificar si el jugador existe. En caso de existir, procede con la creación de la transacción.
- Datos: Se insertan los siguientes datos en la transacción.
  ID_Player, Transaction_Type, Date_Time, Amount, Currency, Payment_Method, Transaction_Status.
- Nota: El parámetro “p_Payment_Method” tiene que ser los siguientes valores debido al último trigger ("Credit Card", "PayPal" y "Bank Transfer")

Trigger 1: Validate_Transaction_Amount

- Objetivo: Este trigger valida que el monto de una transacción no sea negativo antes de insertar el registro en la tabla de transacciones.
- Tablas Manipuladas:
    - Financial_Transactions: Se activa antes de la inserción de un nuevo registro.
- Detalles: Con la sentencia IF se hace la verificación de que el monto no sea negativo.
- Datos: VARCHAR. Devuelve una cadena de texto en caso de activación del trigger.

Trigger 2: Validate_Payment_Method

- Objetivo: Verifica si el método de pago proporcionado en la nueva transacción está en una lista de métodos permitidos (en este caso, "Credit Card", "PayPal" y "Bank Transfer").
- Tablas Manipuladas:
    - Financial_Transactions: Se activa antes de la inserción de un nuevo registro en esta tabla, y se verifica el método de pago.
- Detalles: Se utiliza un WHERE para verificar que el método de pago es uno de los permitidos. En caso de activación del trigger, se lanza un error.
- Datos: VARCHAR. Devuelve una cadena de texto en caso de activación del trigger.

INSTRUCCIONES

Para ejecutar el proyecto, es necesario copiar el contenido de los enlaces de GitHub: **Base Code**, **Population Code** y **SQL Objects Code**, y ejecutarlos en el orden indicado. De esta manera, todos los objetos estarán creados y listos para su respectivo testeo.

Base Code.

https://github.com/Cavo2/Entregable2/blob/main/BaseCode.sql

Population Code.

https://github.com/Cavo2/Entregable2/blob/main/Population.sql

SQL Objects Code.

https://github.com/Cavo2/Entregable2/blob/main/Objects.sql