Documentación de Vistas, Funciones, Stored Procedures y Triggers

1. Vistas

1.1. Vista: v_resumen_gastos_mensuales

Objetivo:
Proporciona un resumen de los gastos mensuales desglosados por usuario y categoría. Permite analizar cuánto se ha gastado en cada categoría dentro de un mes específico y cuántas transacciones se han realizado.

Tablas involucradas:

- **Gastos:** Contiene los registros de los gastos, con información sobre la fecha, el monto, el usuario que realizó el gasto y la categoría.
- **Categorias:** Almacena las categorías a las que pertenecen los gastos.
- **Usuarios:** Guarda los datos de los usuarios.

Datos generados:

- Año y mes del gasto.
- Categoría del gasto.
- Usuario que realizó el gasto.
- Total gastado por categoría y usuario.
- Cantidad de transacciones realizadas.

Ordenamiento:
Los datos se ordenan primero por año y mes, y luego por el total gastado en orden descendente.

1.2. Vista: v_balance_mensual

Objetivo:
Permite obtener un balance mensual por usuario, comparando los ingresos y gastos para determinar el saldo final de cada mes.

Tablas involucradas:

- **Ingresos:** Contiene los ingresos de cada usuario.
- **Gastos:** Contiene los gastos realizados por los usuarios.
- **Usuarios:** Permite asociar cada ingreso y gasto con un usuario específico.

Datos generados:

- Usuario.
- Año y mes de la operación.
- Total de ingresos en el mes.
- Total de gastos en el mes.
- Balance final (Ingresos - Gastos).

Ordenamiento:
Los resultados se ordenan por usuario, luego por año y mes.

1.3. Vista: v_metodos_pago_frecuentes

Objetivo:
Muestra qué métodos de pago son más utilizados por cada usuario, permitiendo analizar patrones de pago y hábitos financieros.

Tablas involucradas:

- **Gastos:** Contiene información sobre cada gasto realizado.
- **Usuarios:** Permite asociar cada gasto con el usuario correspondiente.
- **MetodosPago:** Contiene los distintos métodos de pago disponibles.

Datos generados:

- Usuario.
- Método de pago utilizado.
- Número de veces que se ha usado ese método.
- Total gastado con ese método de pago.
- Porcentaje de uso del método respecto al total de transacciones del usuario.

Ordenamiento:
Los resultados se ordenan por usuario y frecuencia de uso en orden descendente.

1.4. Vista: v_gastos_por_categoria

Objetivo:
Desglosa los gastos de cada usuario por categoría, permitiendo analizar en qué se está gastando más dinero.

Tablas involucradas:

- **Gastos:** Contiene los detalles de los gastos.
- **Usuarios:** Permite asociar los gastos con el usuario que los realizó.
- **Categorias:** Almacena las distintas categorías de gastos.

Datos generados:

- Usuario.
- Categoría de gasto.
- Total gastado en esa categoría.
- Porcentaje que representa ese gasto dentro del total de gastos del usuario.

Ordenamiento:
Los datos se ordenan por usuario y luego por el total gastado en cada categoría en orden descendente.

---

2. Funciones

2.1. Función: f_calcular_total_gastos_periodo

Objetivo:
Calcula el total de gastos de un usuario en un período determinado.

Tablas utilizadas:

- **Gastos:** Se filtran los registros por id_usuario y el rango de fechas.

Explicación:

- Se declara una variable total para almacenar la suma de los gastos.
- Se usa COALESCE(SUM(monto), 0) para evitar valores nulos.
- Se devuelve el total de gastos en el período especificado.

2.2. Función: f_calcular_balance_periodo

Objetivo:
Determina el balance financiero de un usuario en un período, calculando la diferencia entre ingresos y gastos.

Tablas utilizadas:

- **Ingresos:** Se suman los ingresos del usuario en el período.

- **Gastos:** Se suman los gastos del usuario en el período.

Explicación:

- Se declaran variables para almacenar los ingresos (total_ingresos) y los gastos (total_gastos).
- El balance se obtiene restando total_gastos de total_ingresos.

2.3. Función: f_porcentaje_gasto_categoria

Objetivo:
Calcula el porcentaje que representa el gasto en una categoría específica sobre el total de gastos de un usuario en un período determinado.

Tablas utilizadas:

- **Gastos:** Se filtran los registros por usuario, categoría y período.

Explicación:

- Se obtienen los gastos totales en la categoría (total_categoria).
- Se obtienen los gastos generales del usuario (total_general).
- Se calcula el porcentaje dividiendo total_categoria entre total_general y multiplicando por 100.
- Si el total general es 0, el porcentaje se establece en 0.

2.4. Función: f_meses_sin_gastos

Objetivo:
Devuelve una lista de los meses en los que un usuario no ha registrado gastos en un año determinado.

Tablas utilizadas:

- **Gastos:** Se filtran los registros por usuario y año.

Explicación:

- Se itera sobre los meses del 1 al 12.
- Para cada mes, se verifica si existen registros en la tabla Gastos.
- Si no hay gastos en un mes, se añade el nombre del mes a la lista.
- Se construye un string con los meses sin gastos separados por comas.

- Si el usuario ha registrado gastos en todos los meses, se devuelve "Ninguno".

---

3. Stored Procedures

3.1. Stored Procedure: sp_insertar_gasto

Propósito:
Inserta un nuevo gasto en la base de datos, validando previamente la existencia de los datos relacionados.

Tablas involucradas:

- **Usuarios:** Verifica si el usuario existe.
- **Categorias:** Verifica si la categoría del gasto existe.
- **MetodosPago:** Verifica si el método de pago es válido.
- **Gastos:** Inserta el nuevo gasto si todas las validaciones son correctas.

Proceso:

1. Verifica que p_id_usuario exista en la tabla Usuarios.
2. Verifica que p_id_categoria exista en Categorias.
3. Verifica que p_id_metodo_pago exista en MetodosPago.
4. Valida que el monto sea mayor a cero.
5. Si todo es correcto, se inserta un nuevo registro en Gastos.
6. Retorna un mensaje con el resultado.

3.2. Stored Procedure: sp_actualizar_categoria

Propósito:
Actualiza el nombre de una categoría si no hay otra con el mismo nombre.

Tablas involucradas:

- **Categorias:** Valida la existencia de la categoría y actualiza el nombre.

Proceso:

1. Verifica si existe otra categoría con el mismo nombre.
2. Si ya existe, devuelve un error.

3. Si no existe, actualiza el nombre de la categoría en Categorias.
4. Retorna un mensaje con el resultado.

3.3. Stored Procedure: sp_generar_reporte_mensual

Propósito:
Genera un reporte mensual de ingresos y gastos de un usuario.

Tablas involucradas:

- **Usuarios:** Obtiene el nombre del usuario para el reporte.
- **Ingresos:** Calcula el total de ingresos del mes.
- **Gastos:** Calcula el total de gastos del mes y desglosa los gastos por categoría y método de pago.
- **Categorias:** Relaciona cada gasto con su categoría.
- **MetodosPago:** Relaciona cada gasto con el método de pago usado.

Proceso:

1. Calcula el rango de fechas del mes consultado.
2. Obtiene el nombre del usuario desde Usuarios.
3. Calcula:
    - Total de ingresos y promedio desde Ingresos.
    - Total de gastos y promedio desde Gastos.
4. Determina el balance (Ingresos - Gastos).
5. Muestra el desglose de gastos por categoría.
6. Muestra el desglose de gastos por método de pago.
7. Lista detalladamente cada gasto con su categoría y método de pago.

3.4. Stored Procedure: sp_transferir_gastos

Propósito:
Transfiere todos los gastos de un usuario a otro.

Tablas involucradas:

- **Usuarios:** Verifica que los usuarios de origen y destino existan.

- **Gastos:** Actualiza el id_usuario de los gastos del usuario de origen al usuario de destino.

Proceso:

1. Verifica si p_id_usuario_origen existe en Usuarios.
2. Verifica si p_id_usuario_destino existe en Usuarios.
3. Asegura que los usuarios no sean los mismos.
4. Actualiza en Gastos el id_usuario del usuario de origen con el del usuario destino.
5. Retorna la cantidad de registros transferidos.

---

4. Triggers

4.1. Trigger: trg_after_insert_gasto

Propósito:
Actualiza las estadísticas de categoría cuando se inserta un nuevo gasto.

Tablas involucradas:

- **Gastos:** Se inserta un nuevo gasto.
- **Estadisticas_Categorias:** Se actualiza la cantidad de transacciones, el total gastado y la fecha del último gasto para la categoría correspondiente.

Proceso:

1. Suma el monto del nuevo gasto al total_gastado.
2. Incrementa en 1 la cantidad_transacciones de la categoría.
3. Actualiza la ultimo_gasto con la fecha del nuevo gasto.

4.2. Trigger: trg_before_delete_categoria

Propósito:
Evita que se elimine una categoría si tiene gastos asociados.

Tablas involucradas:

- **Categorias:** Intento de eliminar una categoría.
- **Gastos:** Se verifica si existen gastos asociados a la categoría.

Proceso:

1. Cuenta cuántos gastos están asociados a la categoría (id_categoria).
2. Si hay gastos, se genera un error y se evita la eliminación.

4.3. Trigger: trg_after_update_usuario

Propósito:
Registra en un log los cambios realizados en los datos de los usuarios.

Tablas involucradas:

- **Usuarios:** Se actualizan los datos del usuario.
- **Log_Cambios_Usuario:** Se registra la información anterior y la nueva.

Proceso:

1. Inserta un registro en Log_Cambios_Usuario con:
    - id_usuario afectado.
    - Tipo de acción (UPDATE).
    - Usuario de la base de datos (CURRENT_USER()).
    - Datos antes y después del cambio (nombre y email).

4.4. Trigger: trg_before_delete_usuario

Propósito:
Registra la eliminación de un usuario y evita que se borre si tiene gastos o ingresos asociados.

Tablas involucradas:

- **Usuarios:** Se intenta eliminar un usuario.
- **Gastos:** Se verifica si tiene gastos asociados.
- **Ingresos:** Se verifica si tiene ingresos asociados.
- **Log_Cambios_Usuario:** Se registra la eliminación del usuario.

Proceso:

1. Registra la eliminación en Log_Cambios_Usuario con los datos del usuario eliminado.
2. Verifica si el usuario tiene gastos o ingresos asociados.

3. Si tiene registros, lanza un error y evita la eliminación.