Tabla usuarios

La tabla usuarios es fundamental para gestionar la información de los usuarios en el sistema, permitiendo registrar sus credenciales, datos de contacto, tipo de suscripción, rol y otros detalles relevantes. Esta tabla es esencial para la administración de cuentas de usuarios y para la gestión de permisos en la plataforma.

Descripción de tabla:

- usuario_id (BIGINT PRIMARY KEY AUTO_INCREMENT):
- Descripción: Identificador único de cada usuario en la base de datos.
- Justificación: Se utiliza BIGINT para permitir un amplio rango de identificadores, soportando un gran número de usuarios en la plataforma. El auto-incremento garantiza que cada usuario tenga un ID único asignado automáticamente.

- nombre_usuario (VARCHAR(255)):
- Descripción: Nombre de usuario único para identificar a los usuarios.
- Justificación: Cambiado de TEXT a VARCHAR(255) para limitar la longitud y optimizar el rendimiento. VARCHAR es más adecuado para campos con un límite de longitud conocido.

- email (VARCHAR(255) UNIQUE):
- Descripción: Dirección de correo electrónico del usuario, utilizada como identificador único.
- Justificación: Se cambió de TEXT a VARCHAR(255) para establecer una restricción de unicidad, asegurando que cada email sea único en la base de datos, lo cual es vital para la autenticación y la prevención de cuentas duplicadas.

- password_usuario (TEXT):
- Descripción: Contraseña del usuario en formato hash.
- Justificación: Se utiliza TEXT para permitir almacenar contraseñas cifradas de cualquier longitud, aunque generalmente el tamaño es limitado tras el cifrado.

- fecha_registro (TIMESTAMP DEFAULT CURRENT_TIMESTAMP):
- Descripción: Fecha y hora en que se registró el usuario.
- Justificación: Utilizar CURRENT_TIMESTAMP como valor por defecto permite registrar automáticamente la fecha de creación del usuario, ayudando en el seguimiento de registros.

- tipo_suscripcion (VARCHAR(50) DEFAULT 'basica'):
- Descripción: Tipo de suscripción del usuario, con un valor por defecto de 'básica'.
- Justificación: Permite categorizar a los usuarios según su nivel de suscripción (p. ej., básica, premium). Usar un valor predeterminado garantiza que todos los registros tengan un tipo asignado si no se especifica.

- rol (ENUM('usuario', 'admin') DEFAULT 'usuario'):
- Descripción: Rol del usuario en el sistema, que puede ser 'usuario' o 'admin'.
- Justificación: El uso de ENUM permite limitar las opciones de roles y establecer el rol predeterminado como 'usuario'. Esto es útil para el control de acceso y la gestión de permisos.

- ingresos (DECIMAL(10, 2) NULL):
- Descripción: Ingresos mensuales reportados por el usuario.
- Justificación: Utilizar DECIMAL(10, 2) asegura precisión en la representación de cifras monetarias, permitiendo hasta 10 dígitos con 2 decimales. Este campo es opcional (NULL).

Esta estructura está diseñada para proporcionar un equilibrio entre flexibilidad y optimización, asegurando que los datos clave de los usuarios estén bien definidos y que la tabla pueda escalar para soportar una base de datos grande.

```sql
-- Tabla de usuarios
CREATE TABLE usuarios (
  usuario_id BIGINT PRIMARY KEY AUTO_INCREMENT,
  nombre_usuario VARCHAR(255), -- Cambiado de TEXT a VARCHAR(255)
  email VARCHAR(255) UNIQUE, -- Cambiado de TEXT a VARCHAR(255) para usar UNIQUE
  password_usuario TEXT,
  fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  tipo_suscripcion VARCHAR(50) DEFAULT 'basica', -- Valor por defecto "básica"
  rol ENUM('usuario', 'admin') DEFAULT 'usuario', -- Rol predeterminado como "usuario"
  ingresos DECIMAL(10, 2) NULL -- Ingreso mensual del usuario
);
```

Tabla categorías_metas

La tabla categorias_metas se utiliza para almacenar las categorías de metas o propósitos que pueden estar asociadas a otros elementos en el sistema. Sirve para clasificar y organizar metas de manera estructurada, facilitando la gestión y la búsqueda de metas específicas por categorías.

Descripción de tabla:

categoria_meta_id (BIGINT PRIMARY KEY AUTO_INCREMENT):
Descripción: Identificador único de cada categoría de meta.
Justificación: Se usa BIGINT para soportar un amplio rango de identificadores y garantizar la unicidad. El AUTO_INCREMENT asegura que cada nueva categoría de meta tenga un ID único asignado automáticamente, simplificando la gestión de registros.

nombre_categoria (VARCHAR(255)):
Descripción: Nombre descriptivo de la categoría de la meta.
Justificación: Se cambió de TEXT a VARCHAR(255) para limitar la longitud y mejorar el rendimiento de las consultas y el almacenamiento. VARCHAR es una opción óptima para campos que tienen un tamaño predecible y limitado, asegurando un uso más eficiente del espacio de almacenamiento y de los índices.

El propósito de esta tabla es proporcionar una manera clara y organizada de clasificar las metas en el sistema. Al tener las categorías definidas en una tabla separada, se permite una mejor normalización de la base de datos y se facilita la creación de relaciones con otras tablas que requieran asociarse a categorías específicas de metas.

```sql
-- Tabla de categorías para metas de ahorro
CREATE TABLE categorias_metas (
    categoria_meta_id BIGINT PRIMARY KEY AUTO_INCREMENT,
    nombre_categoria VARCHAR(255) -- Cambiado de TEXT a VARCHAR(255)
);
```

Tabla categorías_gastos

La tabla categorias_gasto está diseñada para almacenar las distintas categorías de gasto, facilitando la clasificación y la organización de los gastos en el sistema. Esta tabla es útil para analizar y gestionar los gastos de forma estructurada, permitiendo la categorización de las transacciones o registros relacionados.

Descripción de tabla:

categoria_gasto_id (BIGINT PRIMARY KEY AUTO_INCREMENT):
Descripción: Identificador único para cada categoría de gasto.
Justificación: Se usa BIGINT para permitir un amplio rango de identificadores únicos, asegurando que la tabla pueda escalar y soportar un número elevado de categorías. El AUTO_INCREMENT facilita la asignación automática de un ID único a cada nueva categoría.

nombre_categoria (VARCHAR(255)):
Descripción: Nombre descriptivo de la categoría de gasto.
Justificación: Se cambió de TEXT a VARCHAR(255) para limitar la longitud del campo y optimizar el rendimiento en las consultas y el almacenamiento. VARCHAR(255) es adecuado para campos con una longitud de texto previsible, y también mejora la eficiencia del uso de índices.

El propósito de esta tabla es proporcionar una forma estructurada de categorizar los gastos. Esto permite una mejor organización y un análisis detallado de los tipos de gastos, ayudando a los usuarios o al sistema a clasificar y gestionar las finanzas de manera más eficiente.

```sql
CREATE TABLE categorias_gasto (
    categoria_gasto_id BIGINT PRIMARY KEY AUTO_INCREMENT,
    nombre_categoria VARCHAR(255) -- Cambiado de TEXT a VARCHAR(255)
);
```

Tabla gastos

La tabla gastos se utiliza para registrar los gastos asociados a los usuarios, incluyendo detalles como el monto, la categoría del gasto, la fecha y una descripción opcional. Esta tabla permite un seguimiento detallado de los gastos individuales y facilita su categorización y análisis.

Descripción de tabla:

id_gasto (BIGINT PRIMARY KEY AUTO_INCREMENT):
Descripción: Identificador único de cada registro de gasto.
Justificación: El uso de BIGINT asegura un amplio rango de valores para soportar una gran cantidad de registros de gastos. El AUTO_INCREMENT permite que cada registro tenga un ID único generado automáticamente.

usuario_id (BIGINT):
Descripción: Identificador del usuario que realizó el gasto.
Justificación: Relaciona el gasto con un usuario específico en la tabla usuarios mediante una clave foránea. Esto asegura que los gastos puedan ser rastreados y filtrados por usuario.

monto (DECIMAL(10, 2)):
Descripción: Monto del gasto con dos decimales de precisión.
Justificación: El tipo DECIMAL(10, 2) es ideal para representar cifras monetarias, ya que proporciona precisión en la representación y manipulación de valores numéricos.

categoria_gasto_id (BIGINT):
Descripción: Identificador de la categoría del gasto.
Justificación: Relaciona el gasto con una categoría específica en la tabla categorias_gasto mediante una clave foránea, lo cual permite clasificar los gastos de forma ordenada y facilitar su análisis.

fecha (TIMESTAMP DEFAULT CURRENT_TIMESTAMP):
Descripción: Fecha y hora en que se registró el gasto.
Justificación: Usar CURRENT_TIMESTAMP como valor por defecto garantiza que el registro tenga automáticamente la marca temporal de creación, útil para el seguimiento y ordenación de los gastos por fecha.

descripcion (TEXT):
Descripción: Detalles adicionales sobre el gasto.
Justificación: El uso de TEXT permite almacenar descripciones de longitud variable, ofreciendo flexibilidad en la cantidad de detalles que se pueden incluir.

Claves foráneas:
FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id):

Justificación: Enlaza el gasto con un usuario en la tabla usuarios, asegurando la integridad referencial y permitiendo un análisis detallado de los gastos por usuario.

FOREIGN KEY (categoria_gasto_id) REFERENCES categorias_gasto (categoria_gasto_id):
Justificación: Vincula el gasto con una categoría en la tabla categorias_gasto, lo que permite una categorización coherente y un análisis más específico.

```
-- Tabla de gastos
CREATE TABLE gastos (
    id_gasto BIGINT PRIMARY KEY AUTO_INCREMENT,
    usuario_id BIGINT,
    monto DECIMAL(10, 2),
    categoria_gasto_id BIGINT,
    fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    descripcion TEXT,
    FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id),
    FOREIGN KEY (categoria_gasto_id) REFERENCES categorias_gasto (categoria_gasto_id)
);
```

Tabla metas_de_ahorro

La tabla metas_de_ahorro se utiliza para gestionar y almacenar los objetivos de ahorro de los usuarios, incluyendo detalles como el nombre de la meta, el monto objetivo, la fecha límite, el estado actual, y la categoría de la meta. Esta estructura permite a los usuarios definir, seguir y cumplir sus metas financieras de manera organizada.

Descripción de tabla:

meta_id (BIGINT PRIMARY KEY AUTO_INCREMENT):
Descripción: Identificador único de cada meta de ahorro.
Justificación: Se usa BIGINT para asegurar un amplio rango de identificadores y garantizar la unicidad. El AUTO_INCREMENT facilita la asignación automática de IDs únicos.

usuario_id (BIGINT):
Descripción: Identificador del usuario al que pertenece la meta de ahorro.
Justificación: Se establece como clave foránea para relacionar la meta con un usuario específico en la tabla usuarios, garantizando la integridad referencial.

nombre_meta (VARCHAR(255)):
Descripción: Nombre descriptivo de la meta de ahorro.
Justificación: Cambiado de TEXT a VARCHAR(255) para limitar la longitud del campo, mejorar el rendimiento y hacer que el campo sea indexable.

monto_objetivo (DECIMAL(10, 2)):
Descripción: Monto objetivo de la meta de ahorro.
Justificación: Utilizar DECIMAL(10, 2) es ideal para representar cifras monetarias con precisión, permitiendo almacenar hasta 10 dígitos con 2 decimales.

fecha_limite (TIMESTAMP):
Descripción: Fecha límite para alcanzar la meta de ahorro.
Justificación: Usar TIMESTAMP permite almacenar fechas y horas, lo cual es útil para hacer seguimiento y generar recordatorios o alertas.

monto_actual (DECIMAL(10, 2) DEFAULT 0):
Descripción: Monto acumulado actual en la meta de ahorro.
Justificación: Inicializar el valor en 0 facilita el seguimiento del progreso desde el inicio y asegura que el campo esté definido al crear la meta.

descripcion (TEXT):
Descripción: Descripción detallada de la meta de ahorro.
Justificación: TEXT permite descripciones de longitud variable, proporcionando flexibilidad para que los usuarios ingresen detalles relevantes.

estado_de_meta (VARCHAR(50) DEFAULT 'en progreso'):
Descripción: Estado actual de la meta, como 'en progreso', 'completada' o 'cancelada'.
Justificación: Cambiado de TEXT a VARCHAR(50) para limitar la longitud y mejorar la consistencia del campo. Establecer un valor predeterminado de 'en progreso' ayuda a definir el estado inicial de la meta.

categoria_meta_id (BIGINT):
Descripción: Identificador de la categoría a la que pertenece la meta de ahorro.
Justificación: Clave foránea que relaciona la meta con una categoría específica en la tabla categorias_metas, permitiendo clasificar y organizar las metas.

Claves foráneas:

FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id):
Justificación: Relaciona la meta de ahorro con un usuario, manteniendo la integridad referencial y permitiendo el seguimiento de metas por usuario.
FOREIGN KEY (categoria_meta_id) REFERENCES categorias_metas (categoria_meta_id):
Justificación: Asocia la meta de ahorro con una categoría específica para facilitar la organización y el análisis por tipo de meta.

```sql
CREATE TABLE metas_de_ahorro (
  meta_id BIGINT PRIMARY KEY AUTO_INCREMENT,
  usuario_id BIGINT,
  nombre_meta VARCHAR(255), -- Cambiado de TEXT a VARCHAR(255)
  monto_objetivo DECIMAL(10, 2),
  fecha_limite TIMESTAMP,
  monto_actual DECIMAL(10, 2) DEFAULT 0, -- Valor inicial de 0
  descripcion TEXT,
  estado_de_meta VARCHAR(50) DEFAULT 'en progreso', -- Cambiado de TEXT a VARCHAR(50)
  categoria_meta_id BIGINT,
  FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id),
  FOREIGN KEY (categoria_meta_id) REFERENCES categorias_metas (categoria_meta_id)
);
```

Tabla recordatorios

La tabla recordatorios está diseñada para gestionar y almacenar recordatorios que los usuarios pueden configurar para eventos específicos, actividades o alertas importantes. Esta tabla permite a los usuarios programar recordatorios con descripciones y fechas personalizadas.

Descripción de tabla:

recordatorio_id (BIGINT PRIMARY KEY AUTO_INCREMENT):
Descripción: Identificador único para cada recordatorio.
Justificación: Se usa BIGINT para soportar un gran número de recordatorios y garantizar la unicidad. El AUTO_INCREMENT asegura que cada registro tenga un identificador único generado automáticamente.

usuario_id (BIGINT):
Descripción: Identificador del usuario que ha creado el recordatorio.
Justificación: Establecido como clave foránea para relacionar el recordatorio con un usuario específico en la tabla usuarios, manteniendo la integridad referencial y facilitando la organización de recordatorios por usuario.

descripcion (TEXT):
Descripción: Detalles del recordatorio.
Justificación: Se utiliza TEXT para permitir que los usuarios incluyan descripciones de longitud variable, ofreciendo flexibilidad para proporcionar detalles relevantes.

fecha_recordatorio (TIMESTAMP):
Descripción: Fecha y hora en que se debe activar o mostrar el recordatorio.
Justificación: Usar TIMESTAMP permite almacenar la fecha y hora precisa para el recordatorio, lo cual es útil para activar alertas y notificaciones en el momento adecuado.

Claves foráneas:
FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id):
Justificación: Relaciona el recordatorio con un usuario en la tabla usuarios, garantizando la integridad referencial y permitiendo la gestión de recordatorios específicos para cada usuario.

```sql
-- Tabla de recordatorios
CREATE TABLE recordatorios (
  recordatorio_id BIGINT PRIMARY KEY AUTO_INCREMENT,
  usuario_id BIGINT,
  descripcion TEXT,
  fecha_recordatorio TIMESTAMP,
  FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id)
);
```

La tabla recordatorios proporciona una manera eficiente de que los usuarios programen y gestionen recordatorios dentro del sistema. Esto es útil para mantener organizadas las tareas y actividades, asegurando que los usuarios reciban notificaciones y recordatorios a tiempo

Tabla reportes

La tabla Reportes se utiliza para almacenar información relacionada con reportes creados por los usuarios. Estos reportes pueden ser utilizados para notificar problemas, sugerencias, o cualquier tipo de incidencia dentro del sistema. La tabla permite registrar el título, la descripción y el estado del reporte, así como la fecha de creación y el usuario que lo creó.

Descripción de tabla:

reporte_id (INT AUTO_INCREMENT PRIMARY KEY):

Descripción: Identificador único de cada reporte.

Justificación: Se usa INT con AUTO_INCREMENT para asignar automáticamente un ID único a cada reporte, lo cual es esencial para la gestión y referencia de reportes en otras operaciones del sistema.

titulo (VARCHAR(255) NOT NULL):

Descripción: Título breve que resume el reporte.

Justificación: VARCHAR(255) es adecuado para almacenar títulos de longitud predecible y limitar el espacio de almacenamiento, mejorando la eficiencia de las consultas.

descripcion (TEXT NOT NULL):

Descripción: Detalles completos del reporte.

Justificación: TEXT permite almacenar descripciones de longitud variable, proporcionando la flexibilidad necesaria para que los usuarios ingresen información detallada.

fecha_creacion (TIMESTAMP DEFAULT CURRENT_TIMESTAMP):

Descripción: Fecha y hora en que se creó el reporte.

Justificación: El uso de CURRENT_TIMESTAMP como valor por defecto permite que la fecha de creación se registre automáticamente, lo que es útil para el seguimiento cronológico de los reportes.

usuario_id (BIGINT):

Descripción: Identificador del usuario que creó el reporte.

Justificación: Se usa como clave foránea para enlazar el reporte con un usuario específico en la tabla usuarios, manteniendo la integridad referencial y permitiendo la trazabilidad del reporte por usuario.

estado (ENUM('Pendiente', 'Resuelto') DEFAULT 'Pendiente'):

Descripción: Estado actual del reporte, indicando si está pendiente o resuelto.

Justificación: Usar ENUM permite restringir los valores del estado a opciones específicas, asegurando la consistencia en los datos. El valor predeterminado es 'Pendiente', lo cual facilita identificar reportes que requieren atención.

Claves foráneas:

FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id):

Justificación: Vincula el reporte con un usuario en la tabla usuarios, lo cual es importante para identificar qué usuario generó el reporte y mantener la integridad de los datos.

```sql
CREATE TABLE Reportes (
    reporte_id INT AUTO_INCREMENT PRIMARY KEY,
    titulo VARCHAR(255) NOT NULL,
    descripcion TEXT NOT NULL,
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    usuario_id BIGINT,
    estado ENUM('Pendiente', 'Resuelto') DEFAULT 'Pendiente',
    FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id)
);
```

Tabla notificaciones

La tabla notificaciones se utiliza para almacenar notificaciones generadas para los usuarios. Estas notificaciones pueden ser de diferentes tipos, como relacionadas con metas, gastos u otros eventos importantes. La tabla permite que los usuarios reciban alertas personalizadas y puedan hacer un seguimiento de ellas.

Descripción de tabla:

notificacion_id (BIGINT PRIMARY KEY AUTO_INCREMENT):
Descripción: Identificador único de cada notificación.
Justificación: Se usa BIGINT para soportar una gran cantidad de notificaciones y garantizar la unicidad. El AUTO_INCREMENT permite que cada notificación tenga un ID único asignado automáticamente.

usuario_id (BIGINT):
Descripción: Identificador del usuario que recibe la notificación.
Justificación: Clave foránea que relaciona la notificación con un usuario específico en la tabla usuarios, lo que asegura la integridad referencial y permite asociar notificaciones con usuarios individuales.

tipo (VARCHAR(50)):
Descripción: Tipo de notificación, que puede ser algo como 'meta', 'gastos', etc.
Justificación: Utilizar VARCHAR(50) permite almacenar tipos de notificación de longitud predecible, ofreciendo flexibilidad para categorizar diferentes tipos de notificaciones.

mensaje (TEXT):
Descripción: Contenido o mensaje de la notificación.
Justificación: TEXT es ideal para almacenar mensajes de longitud variable, lo cual permite mayor flexibilidad en el contenido de la notificación.

fecha (TIMESTAMP DEFAULT CURRENT_TIMESTAMP):
Descripción: Fecha y hora en que se generó la notificación.
Justificación: CURRENT_TIMESTAMP como valor por defecto asegura que la fecha de creación se registre automáticamente, lo que es útil para ordenar y mostrar las notificaciones por fecha.

leida (BOOLEAN DEFAULT FALSE):
Descripción: Indica si la notificación ha sido leída o no.
Justificación: Usar un tipo de dato BOOLEAN permite representar de forma sencilla si una notificación ha sido revisada. El valor predeterminado FALSE marca las notificaciones nuevas como no leídas.

Claves foráneas:

FOREIGN KEY (usuario_id) REFERENCES usuarios (usuario_id):
Justificación: Relaciona la notificación con un usuario en la tabla usuarios, manteniendo la integridad referencial y permitiendo la gestión de notificaciones específicas por usuario.

```sql
CREATE TABLE notificaciones (
    notificacion_id BIGINT PRIMARY KEY AUTO_INCREMENT,
    usuario_id BIGINT,
    tipo VARCHAR(50), -- Puede ser 'meta', 'gastos', etc.
    mensaje TEXT,
    fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    leida BOOLEAN DEFAULT FALSE,
    FOREIGN KEY (usuario_id) REFERENCES usuarios(usuario_id)
);
```

Esta tabla está diseñada para almacenar y gestionar notificaciones de manera eficiente, permitiendo que los usuarios reciban alertas sobre eventos importantes. Esto mejora la experiencia del usuario al mantenerlos informados sobre actualizaciones y eventos relevantes relacionados con sus actividades en el sistema.