MintTrack - Stage 2: Database Design

Conceptual Design (ERD)

For the MintTrack project, we created an **Entity-Relationship Diagram (ERD)** to represent the conceptual design of the database. The ERD diagram includes the essential entities and relationships for tracking shared expenses, bill splitting, and notifications.

Relationships

1. **User—Group (Many-to-Many)**
   - **Explanation**: A user can belong to multiple groups, and each group can have multiple users. The UserGroup join table links users and groups.
   - **Implementation**: UserGroup is the table that connects users to the groups they belong to.
   - **Cardinality**: Many-to-Many.
2. **Group—Transaction (One-to-Many)**
   - **Explanation**: Each group can have multiple transactions, but a transaction is associated with one group.
   - **Cardinality**: One-to-Many.
3. **Transaction—User (Many-to-Many)**
   - **Explanation**: Each transaction involves at least two users (a sender and a receiver), and each user can participate in multiple transactions.
   - **Cardinality**: Many-to-Many.
4. **Transaction—Spending (One-to-Many)**
   - **Explanation**: A transaction can involve multiple spending entries, as a transaction might include multiple categories of spending (e.g., food, travel).
   - **Cardinality**: One-to-Many.
5. **Transaction—CurrencyExchange (Many-to-One)**
   - **Explanation**: Each transaction may involve a currency exchange, but a currency exchange rate is valid for many transactions during a specific time window.
   - **Cardinality**: Many-to-One.

Assumptions for Each Entity

- **User**:

Assumptions: Each user must register in the system to participate in groups and manage expenses. Users can belong to multiple groups and will receive notifications about transactions via their registered phone number. The DateCreated attribute captures when the user registered in the system.

Reasoning: The User entity is required to manage individual users' details and track their participation in the system. The phone number is important for sending payment reminders and notifications, while DateCreated helps track account activity and history.

- **Group**:

Assumptions: Groups are created by users to organize shared expenses. Users can create or join multiple groups, and each group is identified by a unique GroupId. The CreatedBy attribute ensures that each group has a responsible creator, and the CreateAt and DeleteAt timestamps track the lifecycle of each group.

Reasoning: The Group entity allows users to manage expenses specific to a set of participants. Tracking the creator of a group and its creation/deletion times helps manage group dynamics and audit group activities.

- **Transaction**:

Assumptions: Transactions represent shared expenses within a group. Each transaction is associated with two users: the sender and the receiver, who are linked via SenderId and ReceiverId. The GroupId connects the transaction to a specific group, while Amount, CurrencyType, and Date capture the financial details of the expense.

Reasoning: The Transaction entity ensures accurate logging of expenses between users within a group. The inclusion of SenderId and ReceiverId helps track who initiated the transaction and who is responsible for the payment. Recording the currency type is essential for handling multi-currency transactions.

- **Spending**:

Assumptions: Each transaction is categorized into a specific type of spending (e.g., food, rent), captured by the Category attribute. The amount spent is recorded in a specific currency (CurrencyType). This entity breaks down each transaction into its spending components for more detailed financial tracking.

Reasoning: The Spending entity supports finer categorization and analysis of expenses within a transaction. This entity is important for generating reports and analytics based on different categories of spending, allowing users to understand their financial behavior better.

- **CurrencyExchange**:

Assumptions: Currency exchange rates are necessary for converting transaction amounts when expenses are logged in different currencies. The SourceCurrency and TargetCurrency attributes define the currency pair, and the Rate provides the exchange value at a particular Timestamp.

Reasoning: The CurrencyExchange entity is crucial for handling multi-currency transactions, ensuring accurate conversion between currencies at the time of the transaction. This enables the application to support users operating across different currencies.

- **Inflation**

Assumptions: Inflation data, captured through the Year, Month, and Rate attributes, is used to contextualize financial data over time. This information can be applied to financial reports or projections to adjust for the effect of inflation on expenses.

Reasoning: The Inflation entity helps to incorporate economic data into the application, allowing for more accurate long-term financial planning. Users can view their expenses in real terms, adjusted for inflation over time.

Normalization

Database Schema Normalization Process (3NF Proof)

In this section, we demonstrate the normalization process for the MintTrack database by listing the functional dependencies for each entity and proving that the schema adheres to the Third Normal Form (3NF).

1. Functional Dependencies for Each Table:

1. **User Table:**
    Functional Dependency:
    UserId → Name, PhoneNumber, DateCreated
    Explanation: UserId is the primary key uniquely identifying each user. All non-key attributes (name, phone number, and registration date) are fully dependent on UserId.

2. **Group Table:**
    Functional Dependency:
    GroupId → GroupName, CreatedBy, CreateAt, DeleteAt
    Explanation: GroupId is the primary key. CreatedBy is a foreign key referencing the User table, ensuring that every group has a responsible creator. All other attributes are fully dependent on the primary key GroupId.

3. **Transaction Table:**
    Functional Dependency:
    TransactionId → GroupId, Amount, CurrencyType, Date, SenderId, ReceiverId
    Explanation: TransactionId is the primary key for each transaction. Each transaction is associated with a group through GroupId and has two participants—SenderId and ReceiverId. All other attributes, such as the amount, currency type, and date, are fully dependent on TransactionId.

4. **Spending Table:**
    Functional Dependency:
    SpendingId → CurrencyType, Category, Amount
    Explanation: SpendingId is the primary key. All attributes—currency type, category, and amount—are fully dependent on SpendingId.

5. **CurrencyExchange Table:**
    Functional Dependency:
    Timestamp → SourceCurrency, TargetCurrency, Rate
    Explanation: Timestamp is the primary key, and the currency pair along with the exchange rate is fully dependent on the timestamp.

6. **Inflation Table:**
    Functional Dependency:
    (Year, Month) → Rate
    Explanation: Year and Month together form a composite primary key, and the inflation rate is fully dependent on this composite key.

3NF Compliance Verification

To prove 3NF compliance, we must ensure:

1NF Verification:

All attributes in every table contain atomic values. For example, the PhoneNumber and Name fields in the User table contain individual, non-divisible values. There are no arrays or nested structures.

2NF Verification:

Every non-primary key attribute in each table is fully dependent on the primary key. For instance, in the Transaction table, the Amount attribute is dependent on TransactionId and not on any subset of attributes like SenderId or ReceiverId.

3NF Verification:

There are no transitive dependencies in any table. For example:

In the User table, PhoneNumber depends only on UserId, not on Name or any other non-key attribute.

In the Transaction table, Amount and CurrencyType are directly dependent on TransactionId and are not transitively dependent on any other non-key attributes like GroupId or SenderId.

3. Conclusion

Based on the analysis above, we conclude that the MintTrack database schema adheres to the Third Normal Form (3NF) requirements:

1NF: All attributes contain atomic values.

2NF: All non-primary key attributes are fully dependent on the primary key.

3NF: There are no transitive dependencies between non-primary key attributes.

Relational Schema

Here is the logical design of the relational schema based on the ERD:

1. User

User(UserId: INT [PK], Name: VARCHAR(100), PhoneNumber: VARCHAR(15), DateCreated: DATE)

2. Group

Group(GroupId: INT [PK], GroupName: VARCHAR(100), CreatedBy: INT [FK to User.UserId], CreateAt: DATE, DeleteAt: DATE)

3. Transaction

Transaction(TransactionId: INT [PK], GroupId: INT [FK to Group.GroupId], Amount: DECIMAL(10,2), CurrencyType: VARCHAR(3), Date: DATE, SenderId: INT [FK to User.UserId], ReceiverId: INT [FK to User.UserId])

4. Spending

Spending(SpendingId: INT [PK], CurrencyType: VARCHAR(3), Category: VARCHAR(50), Amount: DECIMAL(10,2))

5. CurrencyExchange

CurrencyExchange(Timestamp: DATETIME [PK], SourceCurrency: VARCHAR(3), TargetCurrency: VARCHAR(3), Rate: DECIMAL(10,6))

6. Inflation

Inflation(Year: Integer[PK], Month: Integer[PK], rate: Real)

Conclusion

This stage of the project includes a comprehensive database design with normalized tables and clearly defined relationships. The ERD captures all necessary components of the MintTrack system for managing shared expenses, bill splitting, and notifications. The schema adheres to normalization standards to ensure data integrity and efficiency.