SQL – user\_income\_expense\_schedule creation

CREATE TABLE user\_income\_expense\_schedule (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID NOT NULL,

name TEXT NOT NULL, -- e.g., Rent, Paycheck, Daycare

type TEXT NOT NULL CHECK (type IN ('income', 'expense')),

amount NUMERIC NOT NULL,

frequency TEXT NOT NULL CHECK (frequency IN ('weekly', 'bi-weekly', 'monthly', 'quarterly', 'annually')),

start\_date DATE NOT NULL,

end\_date DATE, -- optional for fixed-period events

created\_at TIMESTAMP DEFAULT now()

);

I need a version of the SQL that creates user\_forecast\_calendar and user\_income\_expense\_schedule but separately created

**ChatGPT said:**

Great — here are **two separate SQL table creation scripts** to define:

1. user\_income\_expense\_schedule — stores recurring income/expense definitions (source schedules)
2. user\_forecast\_calendar — stores each generated forecasted event (per due date)

**📄 SQL 1: Create user\_income\_expense\_schedule**

sql

Copy code

CREATE TABLE user\_income\_expense\_schedule (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID NOT NULL,

name TEXT NOT NULL, -- e.g., Rent, Paycheck, Daycare

type TEXT NOT NULL CHECK (type IN ('income', 'expense')),

amount NUMERIC NOT NULL,

frequency TEXT NOT NULL CHECK (frequency IN ('weekly', 'bi-weekly', 'monthly', 'quarterly', 'annually')),

start\_date DATE NOT NULL,

end\_date DATE, -- optional for fixed-period events

created\_at TIMESTAMP DEFAULT now()

);

**📄 SQL 2: Create user\_forecast\_calendar**

sql

Copy code

CREATE TABLE user\_forecast\_calendar (

calendar\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID NOT NULL,

reference\_id UUID REFERENCES user\_income\_expense\_schedule(id),

due\_date DATE NOT NULL,

amount NUMERIC NOT NULL,

event\_type TEXT NOT NULL CHECK (event\_type IN ('income', 'expense')),

created\_at TIMESTAMP DEFAULT now()

);

**🔗 How They Work Together**

| **Table** | **Purpose** |
| --- | --- |
| user\_income\_expense\_schedule | Stores the user's recurring rules (e.g., $1500 bi-weekly) |
| user\_forecast\_calendar | Stores each actual due date created from those rules |

You can:

* Join the two using reference\_id = id
* Query forecast data by user, by date, or by source