# Phase 4: Set Up a Database and Trend News Articles

**Goal:**

Store the **summarized trend articles** from data/final/ into a **PostgreSQL database** for persistent, date-based querying. Each category will have its **own table**. When users request trends for a specific date, the system will check the DB and return matching articles (or inform the user if none exist).

## Steps:

### 1. PostgreSQL Setup

Configured a PostgreSQL database and stored the connection string in the .env file for secure access. This allows the application to connect and interact with the database using environment variables.

### 2. Defined Schema for Each Category

For each predefined category, it is created a dedicated table in the database. Each table includes the following columns:

* id (auto-incremented primary key)
* date (the day the trend was detected)
* summary (the 2–5 sentence summary)
* title, url, publishedAt (original article data)

### 3. Built Database Utility Script (write\_trends\_on\_database.py)

Created a reusable Python script to handle all database operations. It:

* Connects to PostgreSQL
* Creates tables dynamically if they don’t exist
* Deletes existing rows for a specific date
* Inserts new trend articles into the respective category tables

### 4. Integrated Database Logic

After the summarization step, the script reads all CSVs from data/final/. For each category, it deletes any previous records for the current date in that category's table, and then inserts the new articles. This ensures only the latest version is stored if the script is run multiple times in a single day.