

Give your AI a filing cabinet

A database starting point for service-business owners.

An AI assistant needs the right business records before it can help with a job, invoice or customer follow-up. A database gives those records a consistent structure that software can read and update.

The binder and the cabinet

The binder: guidance about your business - services, policies, tone and instructions.

The cabinet: records of what is happening - customers, jobs, invoices and recorded AI actions.

Four tables to start with

Table	What it holds	Question it supports
customers	Identity and contact details	Who is this customer?
jobs	Work, dates, technician and status	What did we do for them?
invoices	Amount, due date and payment status	Which invoices need attention?
ai_action_log	Recorded action, time and result	What did the assistant do?

A foundation, then a workflow

The lesson demonstrates a sample database with Claude and Neon. Creating tables does not automatically send reminders, synchronize a CRM or record AI actions. Those tasks need connected, tested workflows.

Rules help catch specified errors; they do not guarantee true records or correct AI answers. Start with fictional data and one practical question, then inspect the result.

Your starter prompt

Adapt the business details. Ask for a separate demo project before applying generated SQL.

I own a residential plumbing company in Houston, TX. We do water heater installs and repairs, drain cleaning, repipes, faucet and fixture repair, and sewer line inspections. Customers reach us by phone, text, and a website form. A job goes: first contact, quoted, scheduled, done, invoiced, paid.

Build me a simple database that an AI assistant can read and write to. Give me exactly four tables:

- customers (who they are and how to reach them)
- jobs (what we did, when, which tech, for how much)
- invoices (amount, due date, paid or not)
- ai_action_log (every text, email, or call the AI makes, with the result)

For each table:

- (1) Explain the columns in plain English, one line each.
- (2) Give the exact SQL to create the table in Neon (Postgres) with sensible rules, like a date column that only accepts dates and a status column limited to scheduled, in_progress, done, or cancelled.
- (3) Give me 8 rows of realistic fake Houston customers and matching jobs, invoices, and log entries as INSERT statements, with at least two unpaid invoices.

Then give me three questions I can ask the database this week as SQL:

- Who owes me money and how late are they?
- Who has not been serviced in over a year?
- What did the AI send out this week?

Keep every explanation at a 5th-grade reading level.

Check the result

Review table relationships, field rules and the intended project before applying changes. Ask which validations were actually implemented. The action log must be written by the workflow; an empty table cannot report real actions.

Keep your CRM if it works for you. Decide which system owns each record and how updates will move between systems.

Continue with [ComfortGrowth AI on YouTube](#).