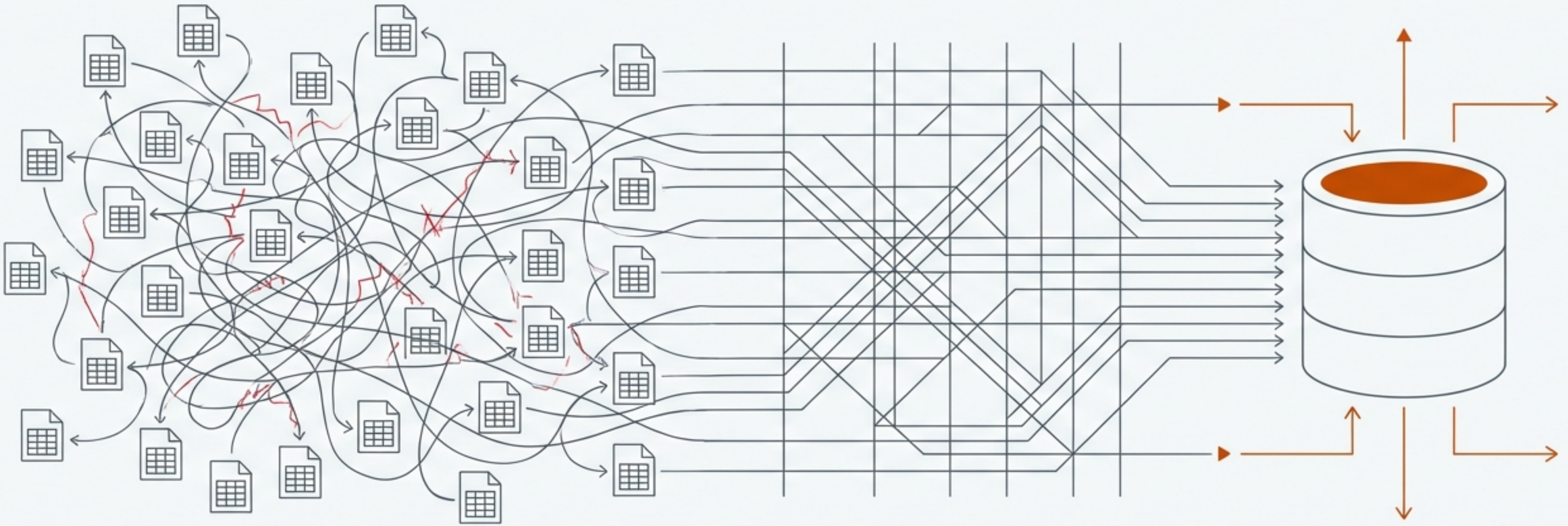
From Spaghetti to Schema

Building the Foundational Database for a Scaling Grocery Chain



Challenge 01 of the 'Grocery Retail – Driven by Data' Series: A journey from spreadsheets to strategy.

A Scaling Business is Drowning in Its Own Data

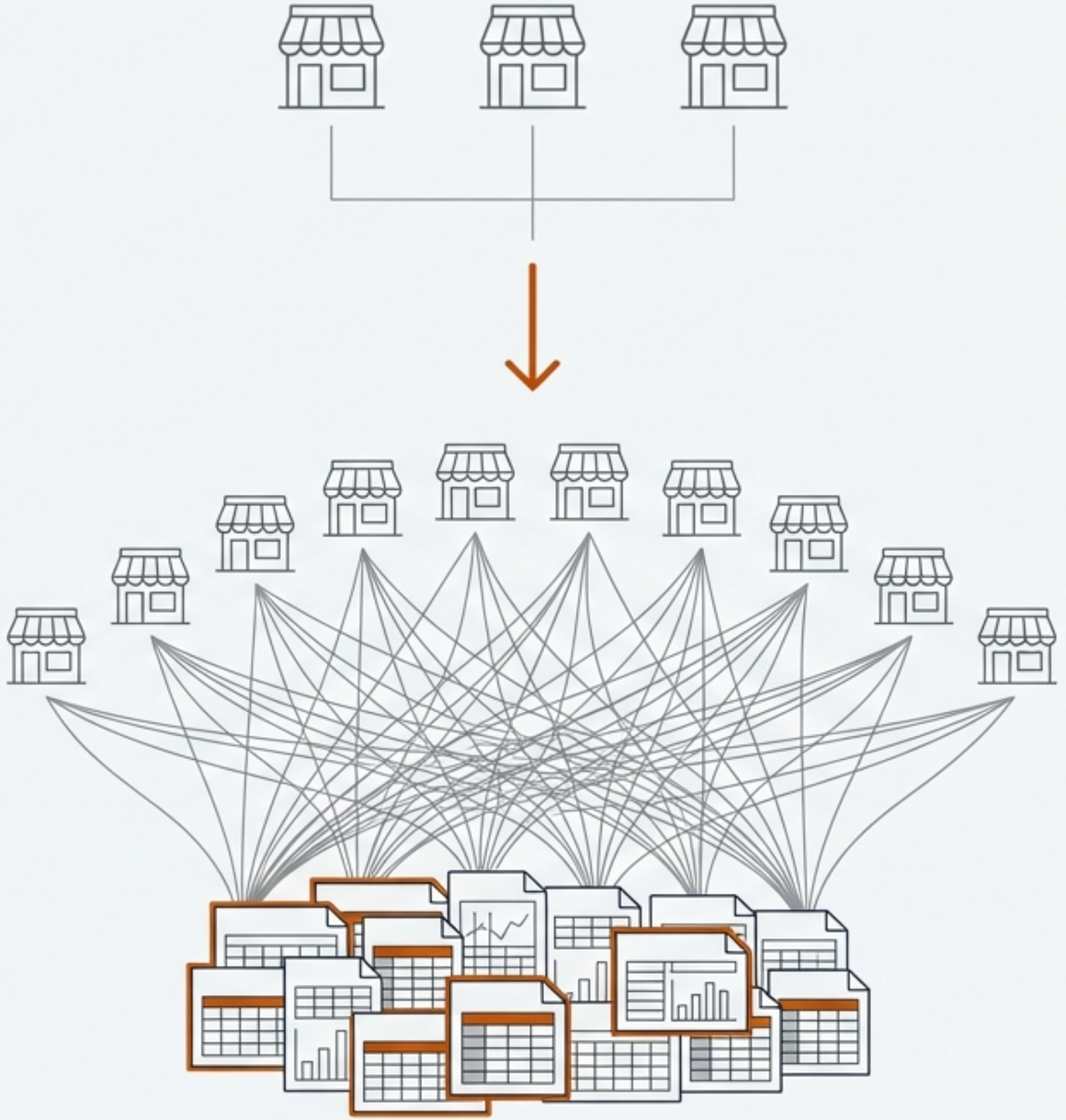
The Business Context

A health-oriented grocery chain is successfully expanding from 3 to 10 stores. This rapid growth is putting immense pressure on their operational infrastructure.

The Core Tension

The business runs on one year of 'Excel spaghetti'—a collection of disparate spreadsheets for sales, items, vendors, and more. What worked for a small operation is now a critical liability.

"Growth without a solid data foundation isn't scaling; it's creating bigger problems."



The Anatomy of ‘Excel Spaghetti’

The reliance on spreadsheets has led to severe data integrity issues that block any attempt at reliable analytics. We identified four primary symptoms of the chaos:

Date	SKU	QTY

Transaction_Date	Product_SKU	Quantity

Duplicated Headers: Inconsistent file structures required manual, error-prone cleaning for every analysis.

Unit_Price
1.99
2.50
N/A
1.75

Mixed Data Types: Columns contained both numbers and text, causing calculations to fail silently.

Date
01/15/23
2023-01-16
17 Jan 2023

Inconsistent Date Formats: Dates were stored in multiple, ambiguous formats (e.g., `MM/DD/YY`, `YYYY-MM-DD`), making time-series analysis impossible.

Category
produce
Produce
prod.

Category Naming Drift: The same product category was often spelled differently across files (e.g., ‘produce,’ ‘Produce,’ 'prod.'), preventing accurate aggregation.

A Deliberate Plan to Engineer a Reliable Data Foundation

To solve the problem permanently, we executed a four-stage process to transform the raw spreadsheets into a robust, analytics-ready database.

1  → 2  → 3  → 4 

Reverse-Engineer the Blueprint

Analyze the spreadsheets to identify core business entities and their relationships, creating a conceptual model.

Architect for Integrity

Design a normalized (3NF) Entity-Relationship Diagram (ERD) and implement it in a Postgres database with strict constraints.

Build a Repeatable Pipeline

Develop an idempotent ETL script to reliably load, clean, and structure the data.

Validate and Analyze

Perform sanity checks and build a baseline predictive model to prove the new foundation's value.

Discovering the Blueprint: Identifying Core Business Entities

Before any code was written, we deconstructed the spreadsheets to define the fundamental components of the grocery business. This conceptual model separates the data into two logical types: Dimensions (the ‘who, what, where, when’) and Facts (the events or transactions).

Dimensions (DIMs)

`store`: Physical store locations.

`product`: Individual items for sale.

`category`: Product hierarchies.

`vendor`: Suppliers of products.

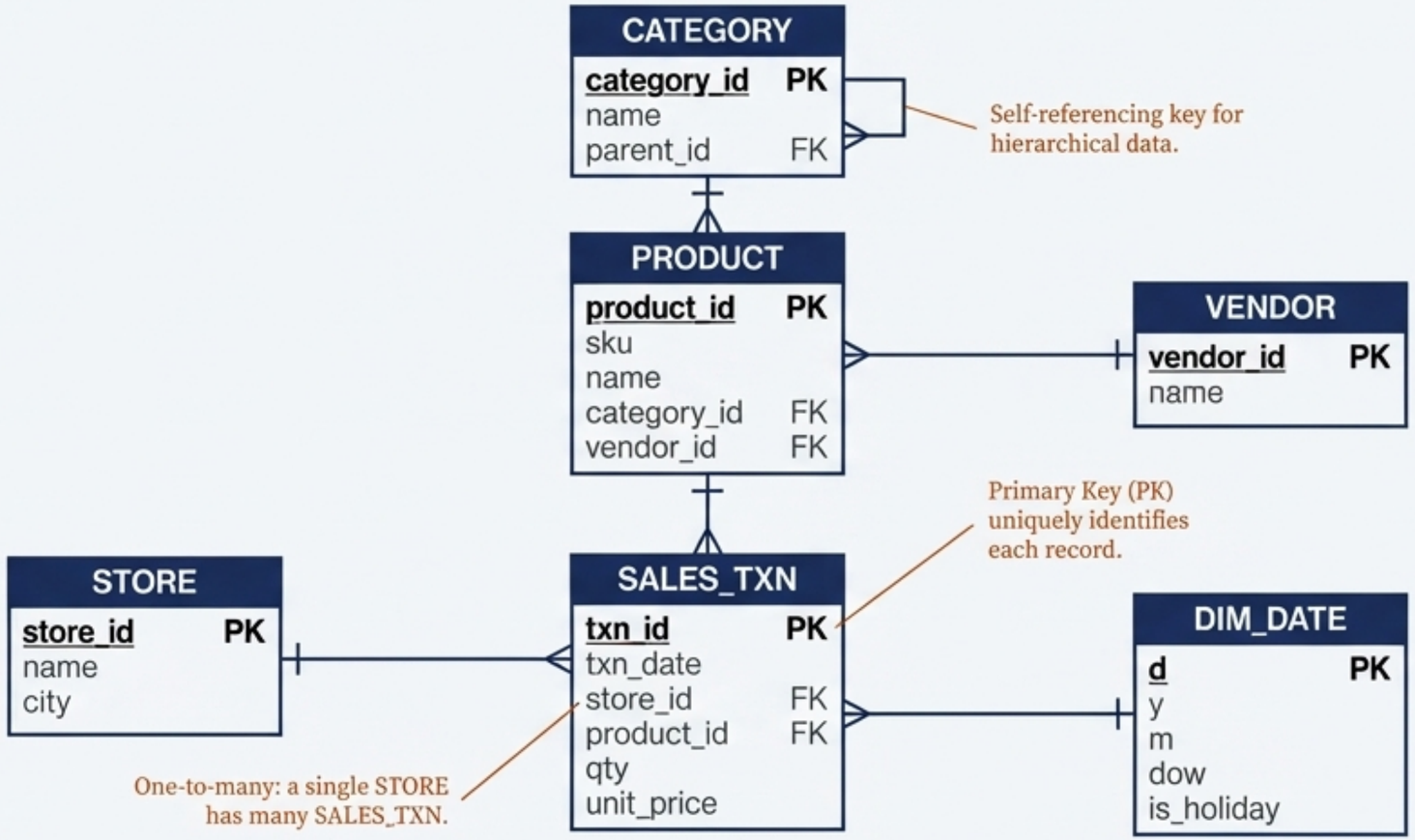
`date`: A comprehensive date table for time-based analysis.

Facts (FACTs)

`sales_txn`: The core sales transaction data at the line-item grain.

(Also Identified for Future Use: `inventory_snapshot`, `promo`)

The Architecture of Order: A Normalized (3NF) Relational Schema



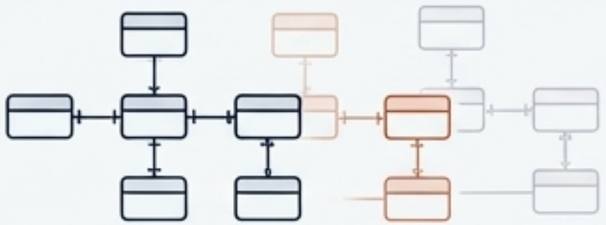
The schema is normalized to Third Normal Form (3NF) to eliminate data redundancy and ensure referential integrity. This structure is the bedrock of trustworthy analytics.

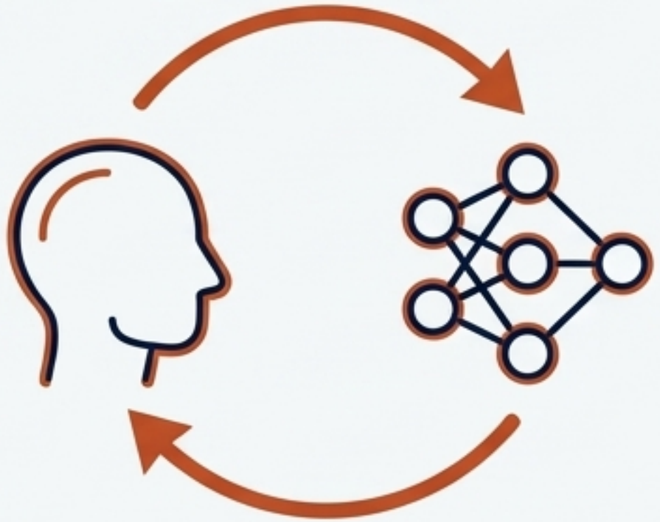
Enhancing Design and Documentation with AI Collaboration

We employed a human-in-the-loop AI workflow to accelerate development and improve the robustness of our solution. This approach combined domain expertise with AI's generative power.

Schema Pressure-Testing

LLMs generated alternative ERD variants based on the business context. These were compared against the human-designed schema to challenge assumptions and ensure comprehensive coverage of business rules.





Accelerated Documentation

An initial data dictionary was AI-generated from the schema and source files. This draft was then refined and validated by a human expert to produce the final, accurate documentation.

The final design remains a product of human expertise, validated and accelerated by AI.

From Blueprint to Reality: Implementing the Schema in Postgres

The ERD was translated into SQL DDL to create the physical database schema in PostgreSQL. The code enforces data integrity at the lowest level through carefully defined constraints and indexes.

Data Integrity

Use of `PRIMARY KEY`, `FOREIGN KEY`, and `NOT NULL` constraints ensures that every record is unique and all relationships between tables are valid.

Performance

Strategic `INDEXES` created on frequently joined columns like `store_id`, `product_id`, and `txn_date` to accelerate query performance.

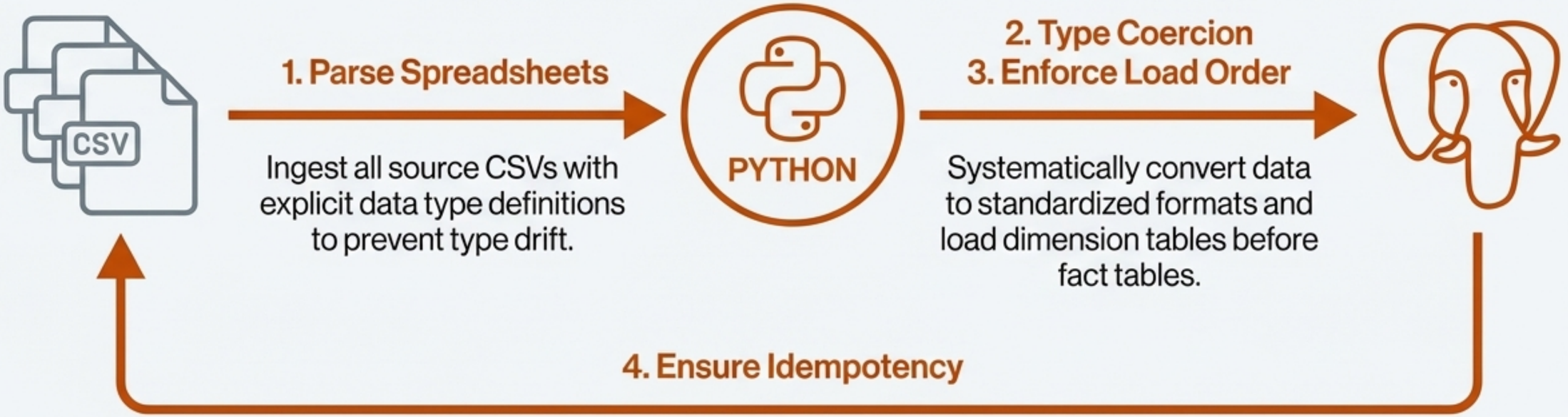
```sql
-- sql/ddl.sql snippet


CREATE TABLE sales_txn (
  txn_id BIGINT PRIMARY KEY,
  txn_date DATE NOT NULL,
  store_id INT NOT NULL,
  product_id INT NOT NULL,
  qty INT,
  unit_price NUMERIC(10, 2),

  -- Foreign Key Constraints
  CONSTRAINT fk_store
    FOREIGN KEY(store_id) REFERENCES store(store_id),
  CONSTRAINT fk_product
    FOREIGN KEY(product_id) REFERENCES product(product_id)
);
```

Building the Bridge: A Reliable and Repeatable ETL Pipeline

With the database structure in place, we developed a Python script (`scripts/load_data.py`) to extract, transform, and load the data from the source spreadsheets. The pipeline was designed to be robust and fully automated.



Verifying Success: The System Passes All Sanity Checks

Before using the data for analysis, we conducted a series of **validation** checks to confirm the ETL process was successful and the database was sound. The system met all acceptance criteria.

Row Counts Match

Verified that record counts in the database tables correspond to the source spreadsheets, accounting for any rejected rows.

Result: ≥ 95% of rows loaded successfully.

Referential Integrity Holds

Confirmed that all foreign key relationships are valid, with no orphaned records.

Result: All foreign key constraints are satisfied.

Duplicate Detection

Ran checks to ensure primary keys are unique and that no duplicate transactions were loaded.

Result: No duplicate primary keys found.

The database is confirmed to be a clean, complete, and reliable source of truth.

The True Impact: Unlocking Predictive Analytics

REACTIVE

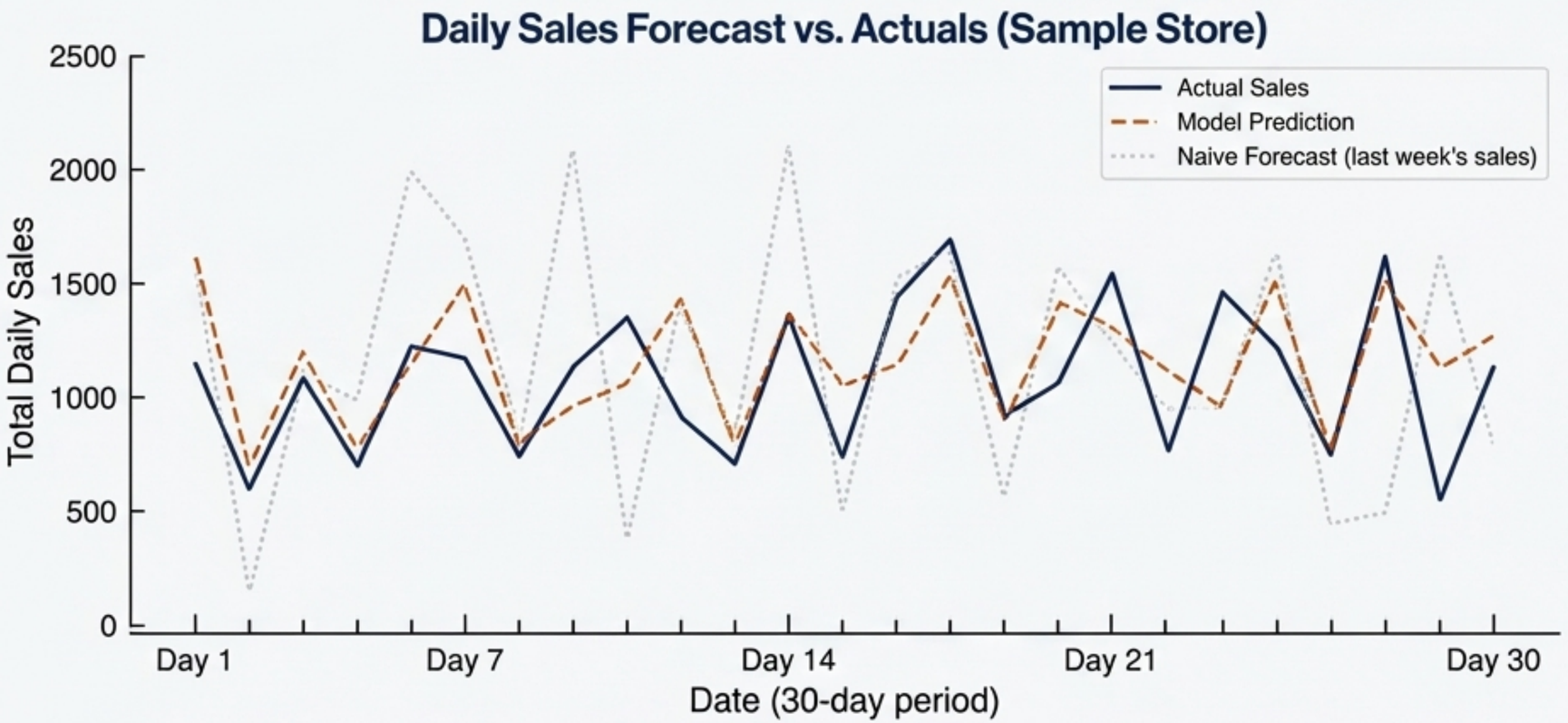
With data trapped in “Excel spaghetti,” the only possible analysis was reactive and historical. Forecasting was guesswork.

PREDICTIVE

The new, structured database enables proactive, data-driven decision-making. As a first proof of value, we built a baseline model to tackle a key business question: **Predict daily sales volume for each store.**

A simple regression model was trained on the clean, structured transaction data, using historical sales and date-based features.

Quantifying the Advantage: From Raw Data to Accurate Forecasts



Model Performance

MAPE: 8.5%
Mean Absolute Percentage Error

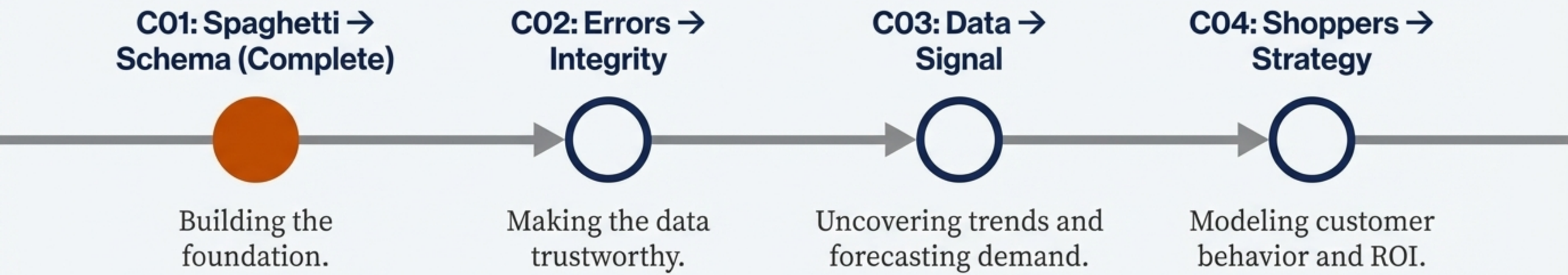
MAE: $215
Mean Absolute Error

Baseline Comparison: Significantly outperforms a naive forecast, demonstrating tangible predictive power.

The structured data foundation is not just clean; it is immediately useful for building predictive models that can inform business operations like staffing and inventory.

The Foundation is Set. The Journey Continues.

Challenge 01, “Spaghetti to Schema,” successfully transformed chaotic data into a strategic asset. This relational database is the essential bedrock upon which all future analytics and data science initiatives will be built.



With a solid data structure in place, we are now ready to build a truly data-driven retail organization.