Nashville Housing Data Warehouse – Technical Specification

1 Project Overview

The Nashville Housing Data Warehouse is an end-to-end analytics stack that ingests raw CSV sales data, stores it in Postgres, transforms it with **dbt**, and surfaces an automated lineage/catalog site. The stack is containerised with **Docker Compose** and version-controlled in **GitHub**, with CI that executes dbt builds and republishes docs on every push.

2 Objectives

- **Ingest** county-assessor CSV into a relational database reproducibly.
- **Clean & standardise** data (staging layer) and expose analytics-ready **fact/dim tables** (mart layer).
- **Validate data** via automated tests (null, range, relationships).
- **Document** lineage and business metadata in an interactive site.
- **Automate** builds/tests/docs in GitHub Actions.

3 Architectural Components

Layer	Technology	Responsibility
Source	CSV file (`Nashville_housing_data_2013_2016.csv`)	Raw assessor exports
Ingestion	Python 3.12 + `pandas`, wrapped in `` container	Load CSV → Postgres table `public.nashville_housing`
Storage	Postgres 15 (Docker)	OLTP database, persistent volume `pgdata_custom`
Transform	**dbt 1.10.3** (Postgres adapter 1.9.0)	Create staging views, dimension & fact tables, run tests
Orchestration	Docker Compose	Brings up DB, pgAdmin, ingester
Documentation	`dbt docs` static site	Lineage graph, catalog, test results
CI/CD	GitHub Actions	Lint/`dbt build`, push docs to GitHub Pages

4 Docker Compose Services

```
version: "3.9"
services:
  db:
    image: postgres:15
    container_name: postgres_container
    env_file: .env              # PG_* vars
    ports: ["5432:5432"]
    volumes:
      - pgdata_custom:/var/lib/postgresql/data
      - ./pg_hba.conf:/etc/postgresql/pg_hba.conf
    command: >
      postgres -c hba_file=/etc/postgresql/pg_hba.conf

  pgadmin:
    image: dpage/pgadmin4
    container_name: pgadmin_container
    environment:
      PGADMIN_DEFAULT_EMAIL: admin@admin.com
      PGADMIN_DEFAULT_PASSWORD: admin123
    ports: ["5050:80"]
    depends_on: [db]

  ingester:
    build:
      context: ./ingester
      dockerfile: Dockerfile.ingester
    container_name: data_ingester
    volumes:
      - .:/app
    working_dir: /app/scripts
    depends_on: [db]

volumes:
  pgdata_custom:
```

5 dbt Project Structure

```
dbt/
 ├─ dbt_project.yml
 ├─ models/
 │   ├─ staging/
 │   │   └─ stg_nashville_housing.sql
 │   └─ marts/
```

```
│       ├─ dims/
│       │   ├─ dim_date.sql
│       │   └─ dim_property.sql
│       └─ facts/
│           ├─ fct_property_sale.sql
│           ├─ fct_monthly_sales.sql
│           └─ fct_yearly_sales.sql
├─ tests/
│   └─ stg_nashville_housing_tests.yml
├─ macros/
└─ target/            # ignored by Git
```

5.1 Naming & schemas

- **staging** models → schema `staging` (views)
- **mart** models → schema `mart` (tables)

5.2 Tests

- **Generic**: `not_null`, `unique`, `accepted_values`, `accepted_range`, `expression_is_true`
- **Severity**: critical failures vs warnings configured per test.

6 Data Model (Star-Schema)

```
public.staging.stg_nashville_housing
          │
          ├─ mart.dim_date
          ├─ mart.dim_property
          └─ mart.fct_property_sale
                    │
                    ├─ mart.fct_monthly_sales
                    └─ mart.fct_yearly_sales
```

Facts

Table	Grain	Measures
`fct_property_sale`	parcel × sale date	`sale_price`, counts
`fct_monthly_sales`	calendar month	`transactions`, `total_sales`, `avg_sale_price`
`fct_yearly_sales`	calendar year	same as monthly

Dimensions

Table	Key	Highlights
`dim_date`	`date_id`	full calendar attributes
`dim_property`	`property_key`	physical attributes, land-use
`dim_owner` *	`owner_key`	snapshot of owners (`UNKNOWN` placeholder)

7 Data Quality Workflow

1. `` runs every model + tests locally and in CI.
2. Critical failures abort pipeline; warnings surface anomalies.
3. Failure rows can be persisted by enabling `+store_failures`.

8 Documentation & Lineage

- Command: `dbt docs generate && dbt docs serve` 🤝local preview.
- Static site in `dbt/target/` is deployed via GitHub Pages.
- Column descriptions (+ models) persisted to Postgres (`+persist_docs`).

9 CI/CD Pipeline (GitHub Actions)

```
name: dbt Build
on: [push, pull_request]
jobs:
  build:
    runs-on: ubuntu-latest
    env:
      DBT_PROFILES_DIR: ./ci_profiles
    steps:
      - uses: actions/checkout@v4
      - uses: actions/setup-python@v5
        with: {python-version: '3.12'}
      - run: pip install dbt-postgres==1.10.3
      - run: dbt deps --project-dir dbt
      - run: dbt build --project-dir dbt --fail-fast
      - run: dbt docs generate --project-dir dbt
      - uses: peaceiris/actions-gh-pages@v4
        with: {publish_dir: ./dbt/target}
```

Secrets `PG_HOST`, `PG_USER`, `PG_PASSWORD` are stored in **Actions Secrets**.

10 Security & Credential Strategy

Item	Storage
Postgres user/pass	`.env` locally | GitHub Secrets in CI
dbt profiles	`profiles.yml` references environment variables

11 Setup Instructions (Local)

```
# 1. clone
git clone https://github.com/baheldeepti/housing-data-warehouse-.git
cd housing-data-warehouse-

# 2. configure env
cp .env.example .env   # edit passwords

# 3. bring up services
docker compose up -d   # db, pgAdmin, ingester

# 4. run transformations
dbt deps --project-dir dbt
dbt build --project-dir dbt

# 5. serve docs
dbt docs serve --project-dir dbt
```

12 Future Enhancements

- Incremental models for append-only fact.
- Airflow or Dagster orchestration.
- Snapshot SCD-2 owner dimension.
- Unit tests for `load_to_postgres.py` using pytest-docker.
- Monitoring dashboard from test results (dbt Artifacts).

© 2025 D. Bahel – MIT Licence