Phase 1: Polish Your Foundations

SQL - Get Really Solid

You know intermediate SQL already, but let's make sure you're *really* comfortable. **Quick assessment**: Can you write queries with:

Multiple JOINs (3+ tables)
Window functions (ROW_NUMBER, RANK, LAG, LEAD) CTEs (Common Table Expressions)
Subqueries in SELECT, WHERE, and FROM

If any of those feel shaky, practice here:

Mode SQL Tutorial - Advanced section
LeetCode SQL Problems - Do 10-15 Medium problems

Practice project: Download the DVD Rental Database and write queries to answer: Which customers have rented the most in the last 30 days?
What are the top 5 films by revenue in each category? Calculate running totals of rentals by day

Phase 2: Build Your First Real Pipeline

Project: Automated ETL Pipeline

This is where it gets fun. Build something that runs on its own.

Choose your data source (pick what interests you):

- **Cricket scores**: Cricbuzz unofficial API or scrape ESPNcricinfo
- **Stock market**: Alpha Vantage - Free API for stock prices
- **Weather tracking**: OpenWeatherMap - Multiple cities
- **Social media trends**: Reddit API - Track popular posts
- **Your own idea**: Anything that has data you care about

What to build:

Step 1: Plan Your Schema (spend time here)

- What data are you collecting?
- What tables do you need?
- What are the primary keys?
- Draw it out on paper or use dbdiagram.io

Step 2: Extract & Transform

- Write Python script to fetch data
- Clean it (handle missing values, format dates, etc.)
- Transform it into the shape you need

Step 3: Load

- Create your PostgreSQL tables
- Insert data (handle duplicates properly)
- Add data validation (check for nulls, unexpected values)

Step 4: Error Handling

- What if API fails? Log it and continue
- What if database connection drops? Retry logic
- What if data format changes? Graceful failure

Step 5: Logging

- Log when pipeline starts
- Log how many records processed
- Log any errors
- Save logs to a file

Step 6: Automation

- Use cron (Linux/Mac) or Task Scheduler (Windows)
- Schedule to run daily/hourly
- Let it run for a week, monitor it

Step 7: Make It Clean

- Refactor into functions (extract, transform, load functions)
- Use a config file for API keys, DB credentials
- Add comments explaining complex parts

Resources:

- Real Python - Building an ETL Pipeline
- Automate with Python - Logging
- Cron tutorial

YouTube recommendations:

- Data Engineering Project from Scratch by Karolina Sowinska
- Python ETL Pipeline Tutorial by TechWithTim

Phase 3: Level Up - Documentation & Containerization

Make It Production-Ready

Part 1: Documentation Write a killer README for your project.

Include:

- What the project does
- Architecture diagram (draw it in draw.io or Excalidraw)
- How to set it up (step by step)
- Dependencies (requirements.txt)
- How to run it
- Sample output/screenshots
- Challenges you faced and how you solved them

Resources:

- How to write a great README
- Markdown guide

Part 2: Docker Basics Containerize your pipeline so it can run anywhere.

Learn Docker:

- Docker Getting Started - Official tutorial
- Docker for Beginners by TechWorld with Nana (YouTube)

What to do:

1. Install Docker Desktop
2. Create a Dockerfile for your pipeline
3. Create docker-compose.yml (include PostgreSQL container)
4. Run your entire pipeline in containers
5. Test that it works the same way

Phase 4: Workflow Orchestration with Airflow

Apache Airflow

This is a big one. Airflow is used by every serious data team.

Learn the basics:

- Apache Airflow Documentation
- Airflow Tutorial for Beginners by Coder2j (YouTube)
- Data Engineering Zoomcamp - Airflow Section - Free, detailed

Setup:

- Install Airflow using Docker (easier than local install)
- Running Airflow in Docker

Understand these concepts:

- DAGs (Directed Acyclic Graphs)
- Operators (PythonOperator, BashOperator, etc.)
- Tasks and Task dependencies
- Scheduling (cron expressions)
- XComs (passing data between tasks)
- Sensors (waiting for conditions)

Project: Convert Your Pipeline to Airflow

Take your Phase 2 pipeline and rebuild it as an Airflow DAG:

Tasks to create:

1. `extract_data` - Fetch from API
2. `validate_data` - Check data quality (not null, right format)
3. `transform_data` - Clean and transform
4. `load_to_staging` - Load to staging table
5. `load_to_production` - Move to production table
6. `data_quality_check` - Verify load was successful
7. `send_notification` - Email on success/failure

Set up:

- Task dependencies (extract → validate → transform → load)

- Retry logic (3 retries with 5 min delay)
- Email alerts on failure
- Schedule to run daily

Go deep here: Airflow has a lot of features. Play with them. Try different operators. Break the DAG and fix it

.

Phase 5: Data Warehousing

Understanding Data Modeling

Learn dimensional modeling:

- Kimball Dimensional Modeling - Read the basics
- Star Schema vs Snowflake by Seattle Data Guy (YouTube)

Key concepts:

- Fact tables (measures, metrics, events)
- Dimension tables (who, what, where, when, why)
- Star schema design
- Slowly Changing Dimensions (SCD Type 1, Type 2)

Project: Build a Data Warehouse

Take your pipeline data and design a proper warehouse:

Example for Weather Pipeline:

- **Fact table**: `fact_weather_readings` (temperature, humidity, wind_speed, reading_timestamp, city_id, weather_condition_id)
- **Dimension tables**:
  - `dim_cities` (city_id, city_name, country, latitude, longitude)
  - `dim_weather_conditions` (condition_id, condition_name, description)
  - `dim_date` (date_id, date, day, month, year, quarter, is_weekend)

What to do:

1. Design your star schema (draw it out)
2. Create dimension and fact tables in PostgreSQL
3. Modify your Airflow pipeline to populate these tables

4. Write analytical queries:
    - Average temperature by month
    - Wettest cities in last 30 days
    - Temperature trends over time
    - Compare cities

Optional: Learn dbt

- dbt Tutorial
- dbt for Beginners by Kahan Data Solutions (YouTube)
- Use dbt to create transformations instead of writing raw SQL

Move on when: You understand why warehouses are structured this way and have built one.

Phase 6: Cloud Deployment (AWS)

Move to the Cloud

AWS Free Tier Setup:

- Create AWS Free Tier account
- Set up billing alerts (so you don't accidentally spend money)

Learn AWS basics:

- AWS for Beginners by Simplilearn (YouTube)
- AWS Data Engineering by Darshil Parmar (YouTube)

Services to understand:

- **S3**: Object storage (store files, data)
- **RDS**: Managed PostgreSQL (your database in the cloud)
- **EC2**: Virtual servers (run your Airflow here)
- **Glue**: AWS ETL service (optional to explore)
- **Athena**: Query S3 data with SQL (optional)

Project: Cloud Pipeline

What to build:

1. Create RDS PostgreSQL instance
2. Migrate your data warehouse to RDS

3. Store raw data files in S3
4. Run Airflow on EC2 (or use Docker on EC2)
5. Configure security groups properly
6. Set up automated backups

Optional additions:

- Use AWS Glue for transformations
- Query S3 data using Athena
- Set up CloudWatch for monitoring

Resources:

- AWS RDS Tutorial
- Deploying Airflow on AWS by Data Engineering Simplified (YouTube)

Phase 7: Advanced Tools

Option A: Apache Spark (Big Data Processing)

When you need it: Processing large datasets (GBs to TBs)

Learn:

- PySpark Tutorial - Spark by Examples
- Spark for Data Engineers by Data Engineering Simplified (YouTube)

Project idea:

- Download large CSV dataset from Kaggle (100MB+)
- Process it with PySpark
- Compare performance to Pandas
- Do aggregations, joins, transformations

Option B: Stream Processing (Real-time Data)

When you need it: Handling real-time events/data streams

Learn Kafka basics:

- Kafka in 100 Seconds by Fireship (YouTube)
- Kafka Tutorial - Official docs
- Kafka for Beginners by Confluent (YouTube)

Project idea:

- Set up local Kafka
- Create producer that streams data (API calls, sensor data simulation)
- Create consumer that processes and stores in database
- Build simple real-time dashboard

Option C: Data Quality & Testing

Learn Great Expectations:

- Great Expectations Tutorial
- Data Quality Testing by Seattle Data Guy (YouTube)

Project:

- Add data validation to your pipeline
- Create expectations (data tests)
- Automate quality checks
- Generate data quality reports

Phase 8: Portfolio Project (The Big One)

Build Something Impressive

This is your showcase project. Pick something you genuinely find interesting.

Project Ideas:

Option 1: Multi-Source Analytics Platform

Good if: You want to show integration skills

What to build:

- Combine 2-3 different data sources (APIs, scraping, files)
- Build dimensional data warehouse
- Use dbt for transformations
- Orchestrate with Airflow
- Deploy on AWS
- Create simple dashboard (Streamlit or Dash)

Example: E-commerce Analytics

- Scrape product prices from multiple sites
- Track price changes over time
- Analyze trends by category
- Alert when prices drop

Option 2: Real-time + Historical Pipeline

Good if: You learned Kafka and want to show streaming

What to build:

- Collect real-time data (Twitter API, stock prices, sports scores)
- Store streaming data with Kafka
- Process and store in data warehouse
- Calculate rolling metrics
- Create API to query the data

Example: Live Sports Analytics

- Stream cricket/football match data
- Calculate live statistics
- Store historical data
- Compare current match to historical averages

Option 3: End-to-End Data Product

Good if: You want to show full stack

What to build:

- Automated data collection
- Data warehouse with proper modeling
- dbt transformations
- Airflow orchestration
- Cloud deployment
- Interactive dashboard (Streamlit/Dash/Tableau)
- API for data access

Example: Personal Finance Tracker

- Import transactions from multiple sources
- Categorize spending automatically
- Track trends over time
- Budget vs actual analysis
- Forecasting

Option 4: Your Own Idea

Best option if: Something genuinely interests you

Think about:

- What data do you wish existed?
- What repetitive analysis could you automate?
- What insights would be valuable to you or others?

Requirements for any project:

- Use most of what you've learned
- Deploy to cloud
- Automate everything
- Proper data modeling
- Great documentation
- Clean, modular code

Documentation is critical:

- Detailed README with architecture diagram

- Blog post explaining what you built and why
- Screenshots/demo video
- Challenges faced and solutions
- What you learned

Resources for building dashboards:

- Streamlit Tutorial
- Dash by Plotly

.

Phase 9: Job Prep

Polish Your Presence

GitHub Portfolio:

- Pin your best 3 projects
- Clean up code, add comments
- Outstanding READMEs for each
- Consistent commit history (shows you're active)

Portfolio Website (optional but impressive):

- Use GitHub Pages (free)
- Showcase projects with screenshots
- Write about what you built
- Link to GitHub repos

Resume:

- Tailor for Data Engineering roles
- Highlight: SQL, Python, Airflow, AWS, Docker, dbt
- Feature your projects prominently
- Use action words: "Built", "Automated", "Optimized", "Deployed"
- Quantify impact: "Processed 1M+ records daily", "Reduced pipeline runtime by 40%"

LinkedIn:

- Professional photo

- Summary focused on data engineering
- Add all your skills
- Share your projects
- Connect with data engineers
- Join DE groups

Interview Preparation

SQL Practice (Most important):

- LeetCode Database Problems
- Do 30-40 Medium problems
- Focus on: JOINs, window functions, subqueries, optimization
- SQL Interview Questions by Alex the Analyst (YouTube)

Python Coding Practice:

- HackerRank Python Track
- Focus on: data structures, string manipulation, file processing
- Solve 20-30 medium problems

System Design:

- System Design Primer
- Practice designing data pipelines
- Learn to explain trade-offs
- Data Engineering System Design by Seattle Data Guy (YouTube)

Behavioral Questions:

- Prepare stories about your projects
- Use STAR method (Situation, Task, Action, Result)
- Practice explaining technical decisions
- Common questions:
    - "Tell me about a challenging project"
    - "How did you handle a data quality issue?"
    - "Explain your project architecture"

Mock Interviews:

- Practice with friends
- Record yourself explaining projects
- Use Pramp for free mock interviews
- Time yourself solving SQL problems

Resources Hub

Best YouTube Channels:

- Seattle Data Guy - Industry insights, career advice
- Data Engineering Simplified - Practical tutorials
- Darshil Parmar - End-to-end projects
- Karolina Sowinska - Portfolio projects
- TechWorld with Nana - DevOps/Docker/Cloud

Communities to Join:

- r/dataengineering (Reddit)
- DataTalks.Club (Slack/Discord)
- Data Engineering Discord servers
- LinkedIn Data Engineering groups

Free Courses:

- Data Engineering Zoomcamp by DataTalks.Club
- AWS Free Training
- Google Cloud Skills Boost (free tier)

Datasets for Practice:

- Kaggle Datasets
- Public APIs List
- AWS Open Data
