**Implementation Approach: Data Pipeline**

I built an ETL pipeline using Apache Spark and PostgreSQL that loads, cleans, and optimizes contact-entity data for searching.

## **Step-by-Step Implementation**

### **Step 1: Setting Up the Framework**

I chose Apache Spark for data processing due to its ability to handle large datasets efficiently. The implementation uses:

* PySpark for data transformation
* PostgreSQL for data storage
* JDBC connector for database integration

### **Step 2: Data Acquisition & Cleaning**

I implemented modular data cleaning functions for each data type

### **Step 3: Database Schema Creation**

The pipeline creates a properly structured database schema with appropriate relationships:

* Created tables for places, contacts, techstacks, customer\_mappings, and contact\_sync\_status
* Modified the contact\_sync\_status table to handle real-world sync challenges by removing the foreign key constraint
* Set up appropriate data types and relationships between tables

### **Step 4: Data Loading**

I implemented a batch loading approach to efficiently handle large datasets

### **Step 5: Search Optimization**

I implemented a PostgreSQL-based search solution using:

1. Strategic indexes for text fields and common filters
2. A data model that balances normalization with query performance
3. Materialized views for complex queries

### **Execution Results:**

* Places processed: 1,000 records
* Contacts processed: 182,745 records
* Techstacks processed: 2,300 records
* Customer mappings processed: 2,000 records
* Contact sync records processed: 396,046 records
* Total execution time: 15.88 seconds

**Example queries:**  
**1. Give me all entities where a contact has a title contains finance**

SELECT DISTINCT

p.place\_id,

p.display\_name AS entity\_name,

p.state\_abbr,

p.pop\_estimate\_2022 AS population,

string\_agg(DISTINCT t.name, ', ') AS technologies

FROM

places p

JOIN

contacts c ON p.place\_id = c.place\_id

LEFT JOIN

techstacks t ON p.place\_id = t.place\_id

WHERE

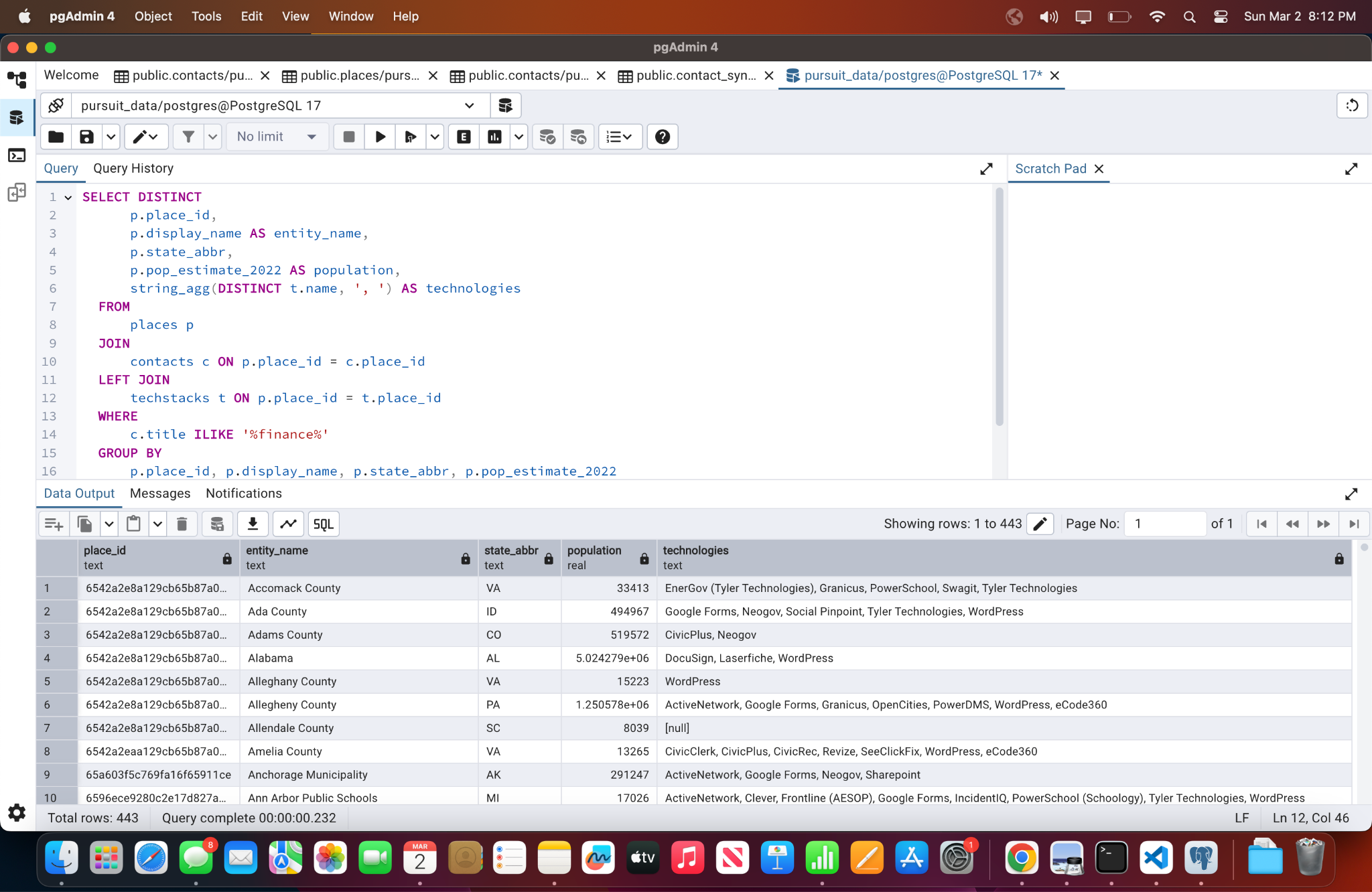
c.title ILIKE '%finance%'

GROUP BY

p.place\_id, p.display\_name, p.state\_abbr, p.pop\_estimate\_2022

ORDER BY

p.display\_name;



**2. Show me all contacts where an email contains bob and entity has technology “Accela” and population > 10k**

SELECT

c.contact\_id,

c.first\_name,

c.last\_name,

c.emails,

c.phone,

c.title,

p.display\_name AS entity\_name,

p.pop\_estimate\_2022 AS population

FROM

contacts c

JOIN

places p ON c.place\_id = c.place\_id

JOIN

techstacks t ON p.place\_id = t.place\_id

WHERE

c.emails ILIKE '%bob%'

AND t.name = 'Accela'

AND p.pop\_estimate\_2022 > 10000;

-- Query 3: Find all entities synced with a specific customer's CRM

SELECT

p.place\_id,

p.display\_name AS entity\_name,

p.state\_abbr,

p.pop\_estimate\_2022 AS population,

cm.external\_crm\_id,

cm.account\_owner,

string\_agg(DISTINCT t.name, ', ') AS technologies

FROM

places p

JOIN

customer\_mappings cm ON p.place\_id = cm.place\_id

LEFT JOIN

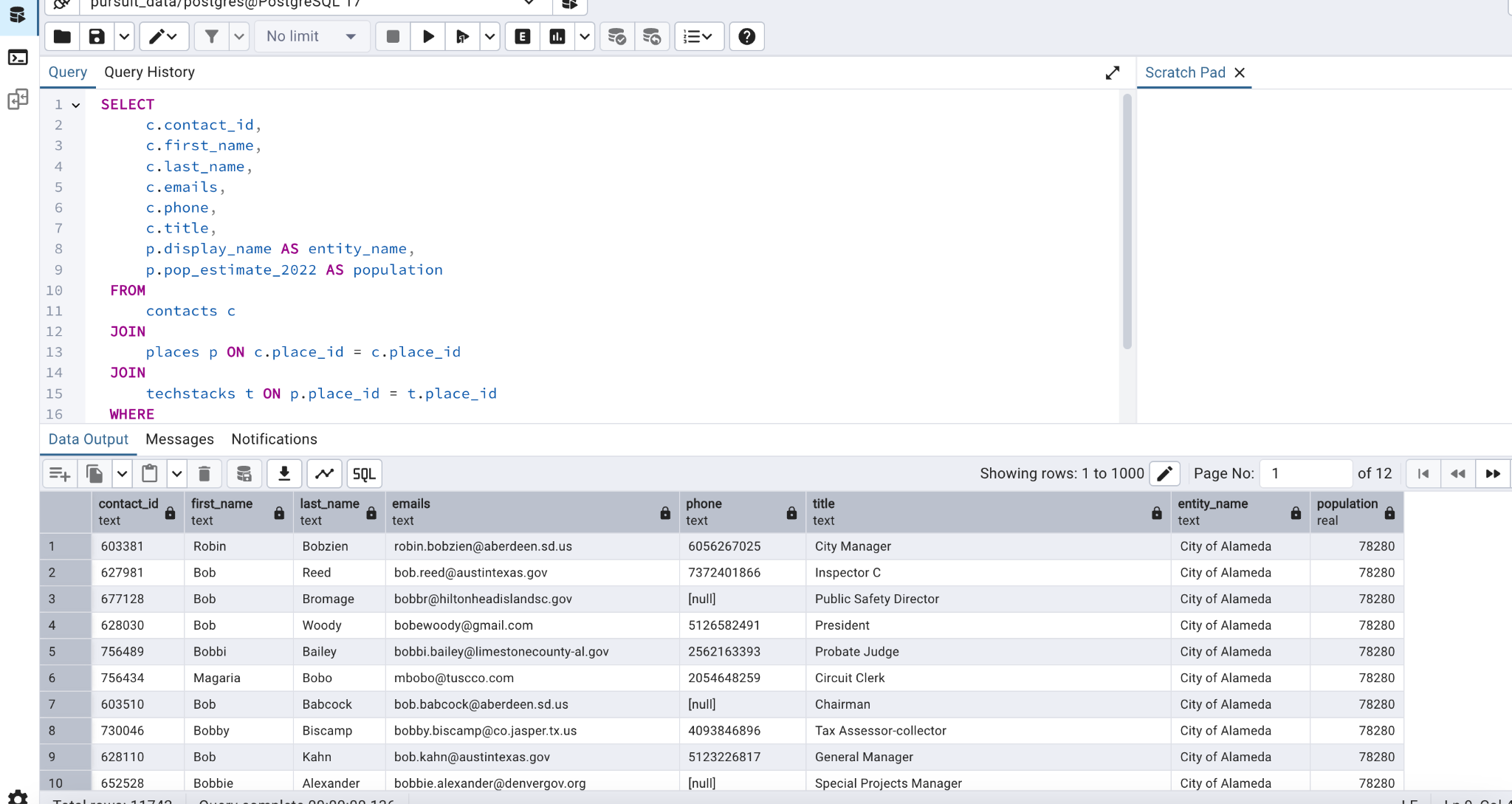
techstacks t ON p.place\_id = t.place\_id

WHERE

cm.customer\_id = 'CustomerA'

GROUP BY

p.place\_id, p.display\_name, p.state\_abbr, p.pop\_estimate\_2022, cm.external\_crm\_id, cm.account\_owner;



**3. Give me all entities that are synced w/ Customer A’s CRM.**

SELECT

p.place\_id,

p.display\_name AS entity\_name,

p.state\_abbr,

p.pop\_estimate\_2022 AS population,

cm.external\_crm\_id,

cm.account\_owner,

string\_agg(DISTINCT t.name, ', ') AS technologies

FROM

places p

JOIN

customer\_mappings cm ON p.place\_id = cm.place\_id

LEFT JOIN

techstacks t ON p.place\_id = t.place\_id

WHERE

cm.customer\_id = 'Customer\_A'

GROUP BY

p.place\_id, p.display\_name, p.state\_abbr, p.pop\_estimate\_2022, cm.external\_crm\_id, cm.account\_owner

ORDER BY

p.display\_name;

