6. Feature Engineering and Transformation

6.1 SQL Schema

The following PostgreSQL schemas are used to store engineered features for the recommendation system. These tables act as a structured feature warehouse layer.

User Features Table

```
CREATE TABLE user_features (
    user_id VARCHAR PRIMARY KEY,
    total_interactions INT,
    avg_interaction_score FLOAT
);
```

Description:

- Stores aggregated user-level behavioral features
- Captures overall user activity and engagement strength

Item Features Table

```
CREATE TABLE item_features (
    product_id INT PRIMARY KEY,
    total_interactions INT,
    avg_interaction_score FLOAT,
    avg_rating FLOAT
);
```

Description:

- Stores item-level popularity and quality signals

- Supports popularity-based and content-aware recommendations

---

User–Item Interaction Features Table

```
CREATE TABLE user_item_features (
    user_id VARCHAR,
    product_id INT,
    interaction_count INT,
    total_interaction_score FLOAT,
    PRIMARY KEY (user_id, product_id)
);
```

Description:

- Captures historical interaction strength between users and items
- Forms the basis for collaborative filtering and matrix factorization models

---

6.2 Transformation Scripts

Feature transformation is implemented using a standalone Python script:

Script Location:

`src/transformation/feature_engineering.py`

Responsibilities of the Script:

- Load prepared interaction data from `data/processed/prepared_interactions.csv`
- Aggregate user-level, item-level, and user–item features
- Connect to PostgreSQL using configuration-driven credentials

- Persist engineered features into relational tables

The script is designed to be **idempotent**, allowing safe re-execution during experimentation or pipeline reruns.

6.3 Summary of Feature Logic

User-Level Features

- **Total Interactions:** Number of interactions performed by a user
- **Average Interaction Score:** Mean interaction strength across all user actions

Purpose:

- Represents user engagement intensity
- Useful for user profiling and cold-start handling

Item-Level Features

- **Total Interactions:** Number of interactions received by an item
- **Average Interaction Score:** Mean engagement strength per item
- **Average Rating:** Mean product rating from metadata

Purpose:

- Captures item popularity and perceived quality
- Supports popularity-based and hybrid recommendation models

User–Item Interaction Features

- **Interaction Count:** Number of times a user interacted with an item
- **Total Interaction Score:** Cumulative interaction strength

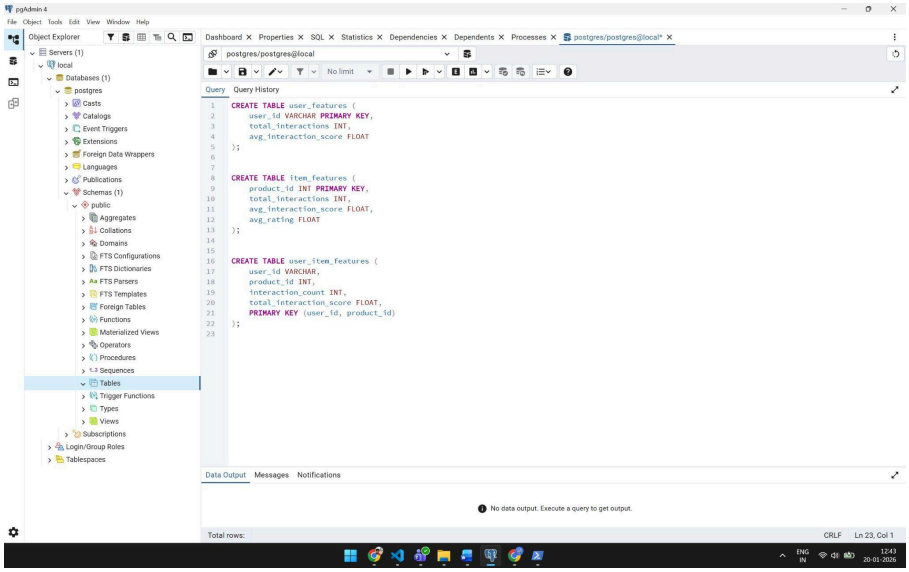
Purpose:

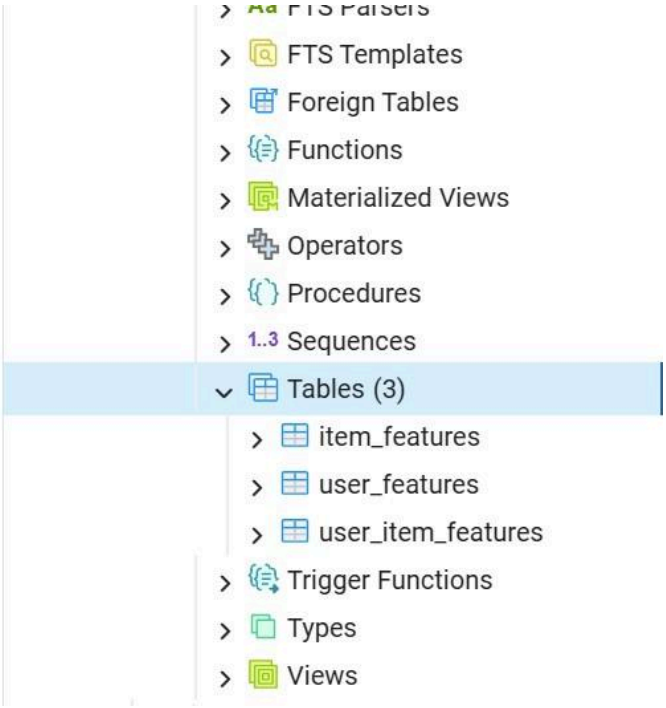
- Encodes historical preference strength
- Directly usable in collaborative filtering algorithms

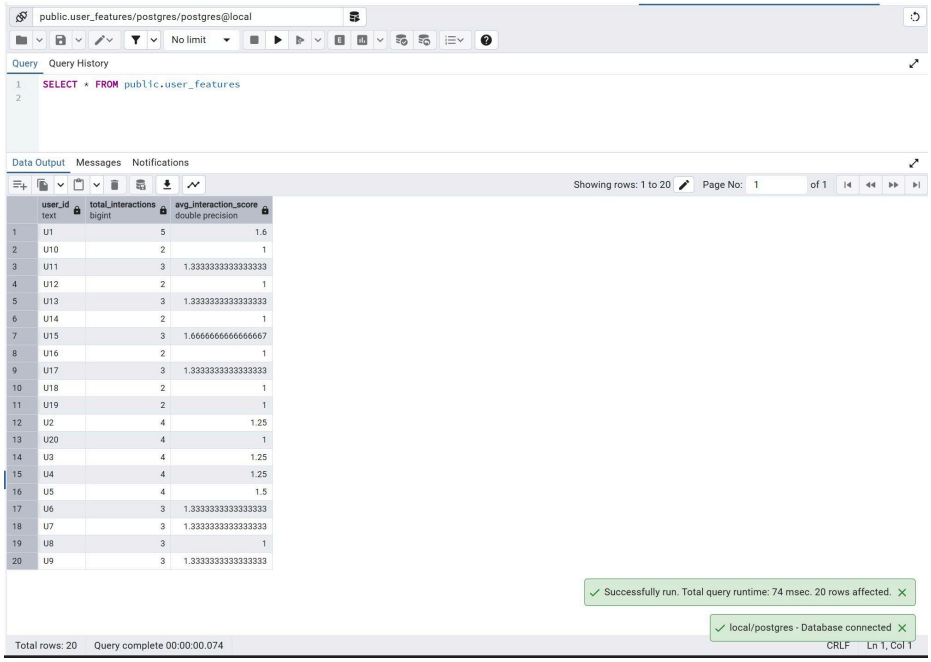
6.4 Outcome

The engineered features are stored in a structured relational format and are ready for:

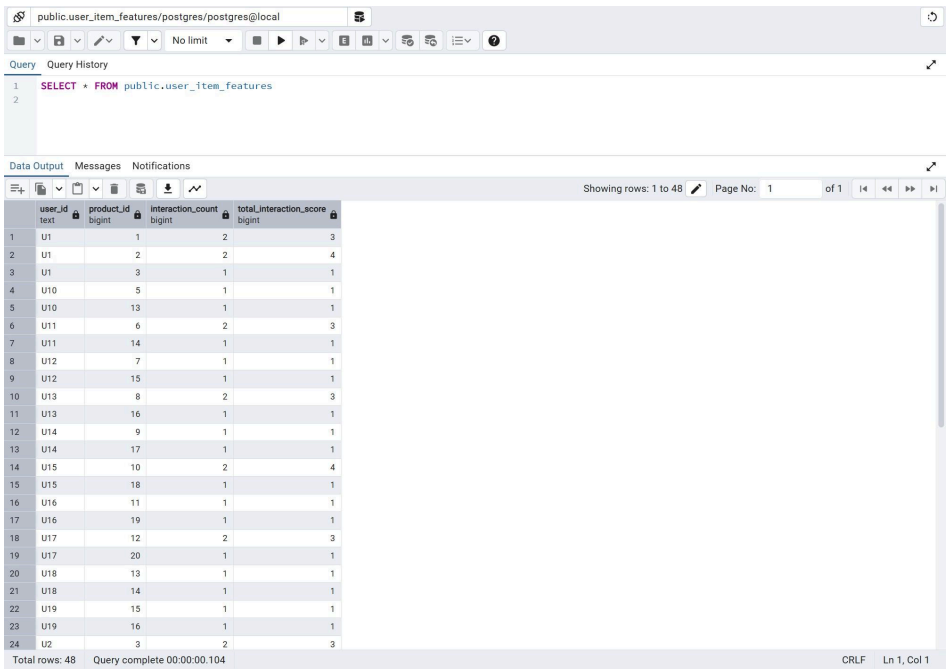
- Model training
- Inference-time feature retrieval
- Feature store abstraction in subsequent pipeline stages


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Servers (1)
local
Databases (1)
postgres
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables
Trigger Functions
Types
Views
Subscriptions
Login/Group Roles
Tablespaces
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
postgres/postgres@local*
postgres/postgres@local
No limit
Query
Query History
CREATE TABLE user_features (
user_id VARCHAR PRIMARY KEY,
total_interactions INT,
avg_interaction_score FLOAT
);
CREATE TABLE item_features (
product_id INT PRIMARY KEY,
total_interactions INT,
avg_interaction_score FLOAT,
avg_rating FLOAT
);
CREATE TABLE user_item_features (
user_id VARCHAR,
product_id INT,
interaction_count INT,
total_interaction_score FLOAT,
PRIMARY KEY (user_id, product_id)
);
Data Output
Messages
Notifications
No data output. Execute a query to get output.
Total rows:
CRLF
Ln 23, Col 1
ENG
IN
12:43
20-01-2026

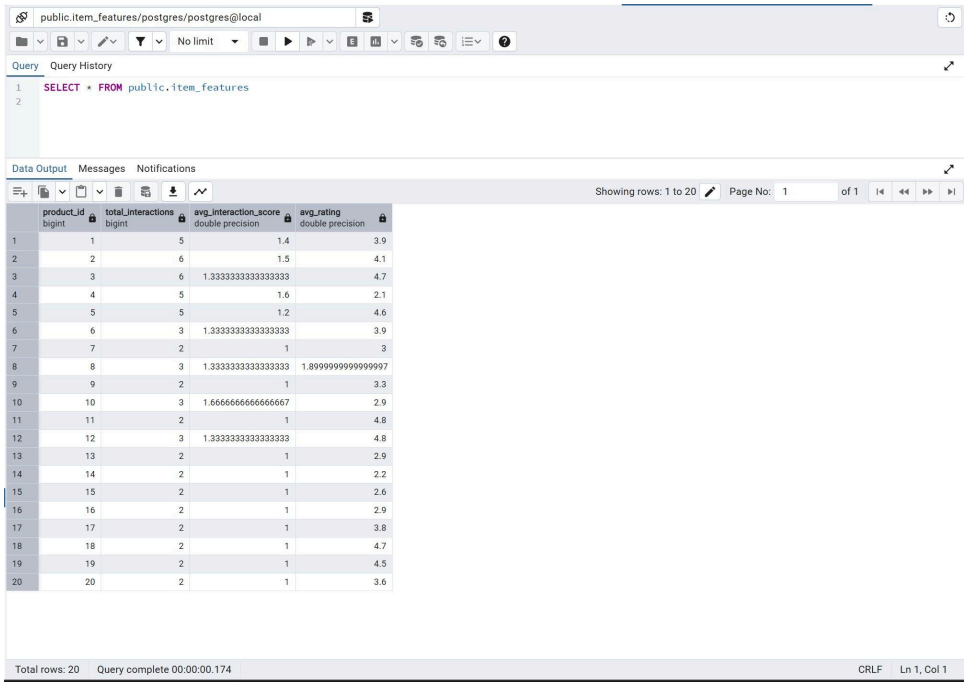

FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (3)
item_features
user_features
user_item_features
Trigger Functions
Types
Views



	user_id text	total_interactions bigint	avg_interaction_score double precision
1	U1	5	1.6
2	U10	2	1
3	U11	3	1.3333333333333333
4	U12	2	1
5	U13	3	1.3333333333333333
6	U14	2	1
7	U15	3	1.6666666666666667
8	U16	2	1
9	U17	3	1.3333333333333333
10	U18	2	1
11	U19	2	1
12	U2	4	1.25
13	U20	4	1
14	U3	4	1.25
15	U4	4	1.25
16	U5	4	1.5
17	U6	3	1.3333333333333333
18	U7	3	1.3333333333333333
19	U8	3	1
20	U9	3	1.3333333333333333



	user_id text	product_id bigint	interaction_count bigint	total_interaction_score bigint
1	U1	1	2	3
2	U1	2	2	4
3	U1	3	1	1
4	U10	5	1	1
5	U10	13	1	1
6	U11	6	2	3
7	U11	14	1	1
8	U12	7	1	1
9	U12	15	1	1
10	U13	8	2	3
11	U13	16	1	1
12	U14	9	1	1
13	U14	17	1	1
14	U15	10	2	4
15	U15	18	1	1
16	U16	11	1	1
17	U16	19	1	1
18	U17	12	2	3
19	U17	20	1	1
20	U18	13	1	1
21	U18	14	1	1
22	U19	15	1	1
23	U19	16	1	1
24	U2	3	2	3



	product_id bigint	total_interactions bigint	avg_interaction_score double precision	avg_rating double precision
1	1	5	1.4	3.9
2	2	6	1.5	4.1
3	3	6	1.3333333333333333	4.7
4	4	5	1.6	2.1
5	5	5	1.2	4.6
6	6	3	1.3333333333333333	3.9
7	7	2	1	3
8	8	3	1.3333333333333333	1.8999999999999997
9	9	2	1	3.3
10	10	3	1.6666666666666667	2.9
11	11	2	1	4.8
12	12	3	1.3333333333333333	4.8
13	13	2	1	2.9
14	14	2	1	2.2
15	15	2	1	2.6
16	16	2	1	2.9
17	17	2	1	3.8
18	18	2	1	4.7
19	19	2	1	4.5
20	20	2	1	3.6

This transformation layer bridges raw interaction data and recommendation models in a scalable and reproducible manner.