Web Data Management Development Project

Group 4

Salim Salmi Jari Blom Bojana Urumovska

TU Delft

Content

- Approach
- Migration
- API
- Limitations
- Conclusion

PostgresSQL → Neo4J

Process

- Use SQL to write postgreSQL tables to .CSV files
- Import those into Neo4J
- Create nodes on idmovie, idactor & idgenre
- Create relationships using the tables acted_in & movies_genres

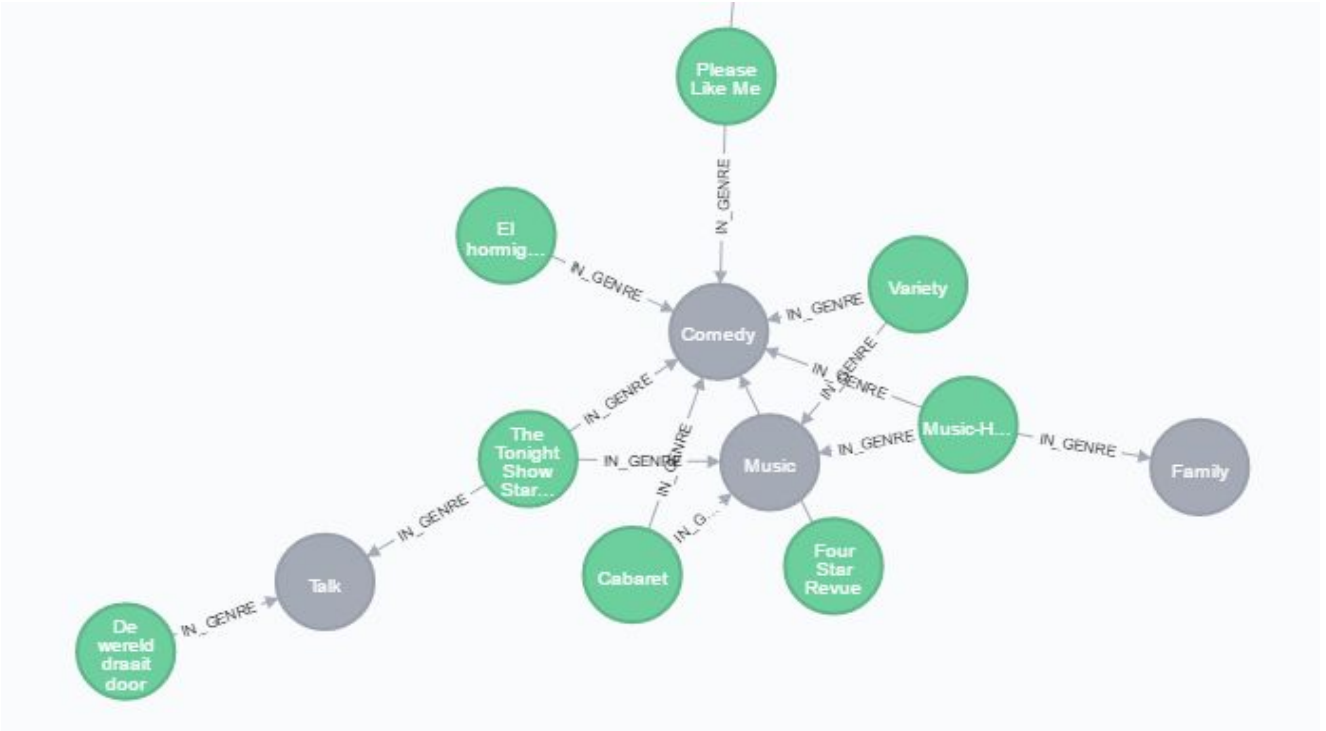
PostgresSQL → Neo4J

Pitfalls

- Installation required of psql
- Little documentation
- Definition of file location OS specific

PostgresSQL → Neo4J

Data Model



TU Delft

PostgresSQL → Neo4J

Data Model

- Nodes for tables “Movies”, “Actors” and “Genres”.
- Replaced intermediate tables “ActedIn” and “MoviesGenres” by graph connections.

PostresSQL → MongoDB

Process

- Mongify to map database (1 to 1)
- Adjust mongo database
- Update relationships

PostresSQL → MongoDB

Process

- Mongify to map database (1 to 1)
- Adjust mongo database
- Update relationships

PostresSQL → MongoDB

Pitfalls

- No easy way to map intermediate tables
- Long time to migrate
- Version conflict with mongify

PostresSQL → MongoDB

Data Model

- Collections for the tables “Genres”, “Movies”, “Actors”
- Removed intermediate tables: “ActedIn” and “MoviesGenres”
- Replace them by array of ObjectIDs per relationship
- Did not migrate other tables that were not used by the API

API: Django

- Django rest framework.
- Different engine for each database so Django knows how to connect.
- Defined models in python.
- Used django to query the databases.

Limitations

- **Neo4J:** Fast ‘join-like’ operations but there is no sharding and much language support for it.
- **Mongodb:** No fixed schema needed and easy to scale but querying and ‘join-like’ operations are harder and harder to manage relationships.
- **Postgres:** Structured and has a lot of support but less freedom.

Conclusion

- For this use case mongodb is less suitable due to the nature of the queries the api needs to make. Cannot make use of the strenghts of the Schema-less nature of noSQL.
- Neo4J and Postgresql are both equal alternatives. Postgres is easier to maintain while the api queries benefit more from Neo4J

DEMO