supper&supper

Graph Database Verification and Optimization

Project Goal

Graph databases offer more intuitive and efficient ways of data querying and analysis than relational databases. For this reason complex SQL databases are migrated into a graph structure. Prior this project a knowledge graph about field trials was created to capture assessment- and metadata about global experiments.[1]

The project focus was to leverage the graph data to train machine learning models. For this reason different scenarios for data export, import and automated processing pipelines were established. In this process the whole graph schema and content was checked and validated against the SQL database to derive optimization approaches. Also different options for enhancing the graph schema with additional data for further analysis were evaluated. Especially the integration of weather data, which was not covered in the SQL database, was a goal for this project.

An efficient way to extract data from the graph and write it back via an Python or R development environment was established. With this set up machine learning models can be trained dynamically and applied on the graph data.

The neo4j graph is fed by an SQL database and should be used for flexible queries to forward data to development environments.

Challenges

A generalized documentation of the complex graph structure is challenging. Also, standard queries and aliases for the entities needed to be defined and documented.

Building up a graph schema offers a lot of complex options and can be done in very different ways. To validate the existing graph schema diverse Machine Learning use cases needed to be defined first. Only then specific graph queries could be created and tested on feasibility. Not all of the future scenarios were defined in the beginning. For this reason the scenarios had to be generic and heterogenic.

Combining and connecting the graph to development environments in a reliable way was necessary for the success of the project.

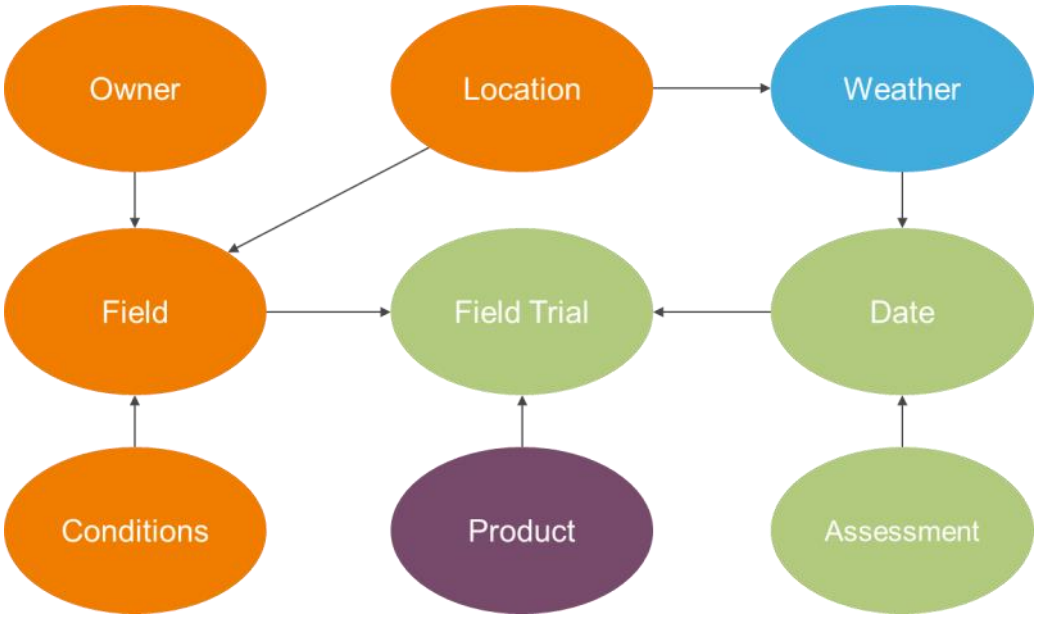
Implementation

Verification

By exporting data from the SQL and the Graph database to the same table-based data format, a full comparison of both databases could be performed. Differences were reverse engineered to their root and recommendations for changes in the graph schema were derived to make sure no information is lost in the future. Also, strategies for the integration of weather data and other data sources were defined. For that several schema design options were tested and evaluated for practical usage.

Querying and interfacing

Guidelines for queries for optimal data extraction, as well as query templates and standard aliases were defined to unify the data extraction. To interface with external development environments, a python BOLT driver was implemented in a machine learning pipeline. This enables dynamic access to the graph database for different analytics and machine learning applications.



Abstracted Graph Schema

Project Outcome

In this project a recommendation catalog was created, including:

- Syntax and datatype changes
- Schema alignments for optimization of frequent queries to increase performance
- Corrections of potential schema deadlocks
- Integration of weather data and further data sources

As an interface to development environments a Python / R pipeline was built up. By integration into analysis tools the graph can be queried on demand so that the results can be leveraged to train machine learning models and apply them directly on more data.

Reference:

[1] Daniel Burkow, Jens Hollunder, Julian Heinrich, Fuad Abdallah, Miguel Rojas-Macias, Cord Wiljes, Philipp Cimiano and Philipp Senger
A Blueprint for Semantically Lifting Field Trial Data: Enabling Exploration using Knowledge Graphs, December 2019.