As I work as a data engineer, my day to day life involves playing with data. So, here I'll be explaining Data catalog generation which I have been working on. We were given the following problem statement.

Problem statement:

A Catalog of all data sources that displays relationships and direct links between tables and their attributes. A responsive web API that reflects a user's search/request.

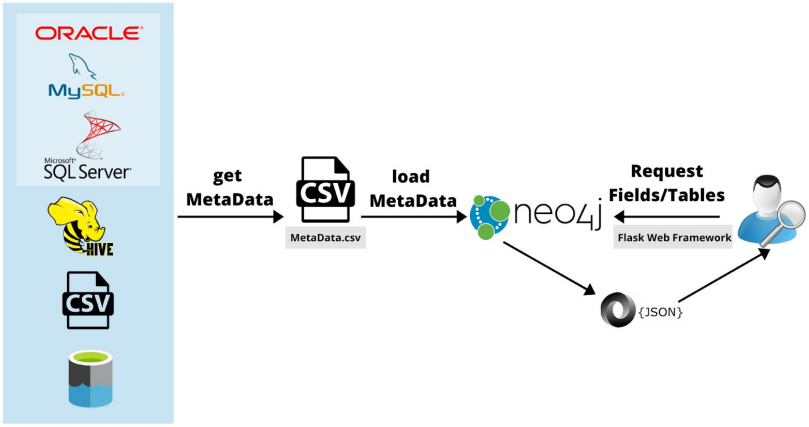
That was very ambiguous initially. We had several questions about **data input**.

- What will be our data sources
- how many data sources we have
- is our data is RDBMS or non-RDBMS
- in what format we are going to get our data
- Where are we going to get the data
- Are we accessing their data server or they are going to drop at our company's data lake

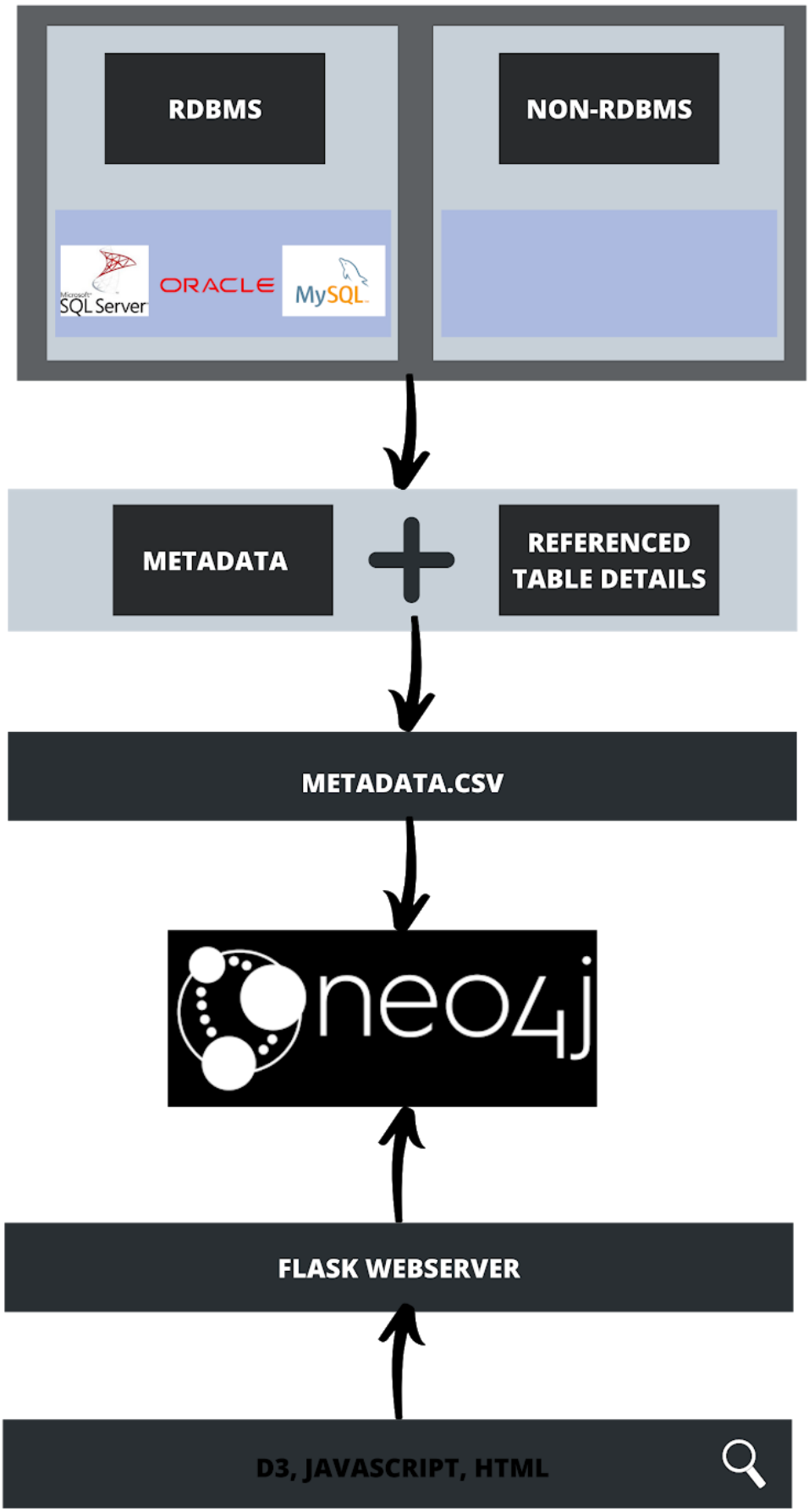
We also had various questions about the output as well.

- What is customer's need
- Where are they going to use the data catalog utility created by us
- In what format we are doing to deliver data catalog output.

This was our analysis phase. We had lots of meetings with our client. We cleared all the related questions we had. We were running lots and lots of technological permutations and combinations. We wanted to give the client the best utility. We also organized the KT- sessions (knowledge transfer) to get to know about the existing technology the client is using so that we can create a utility which is compatible with their existing technology. At the end following is what we understood from that phase.

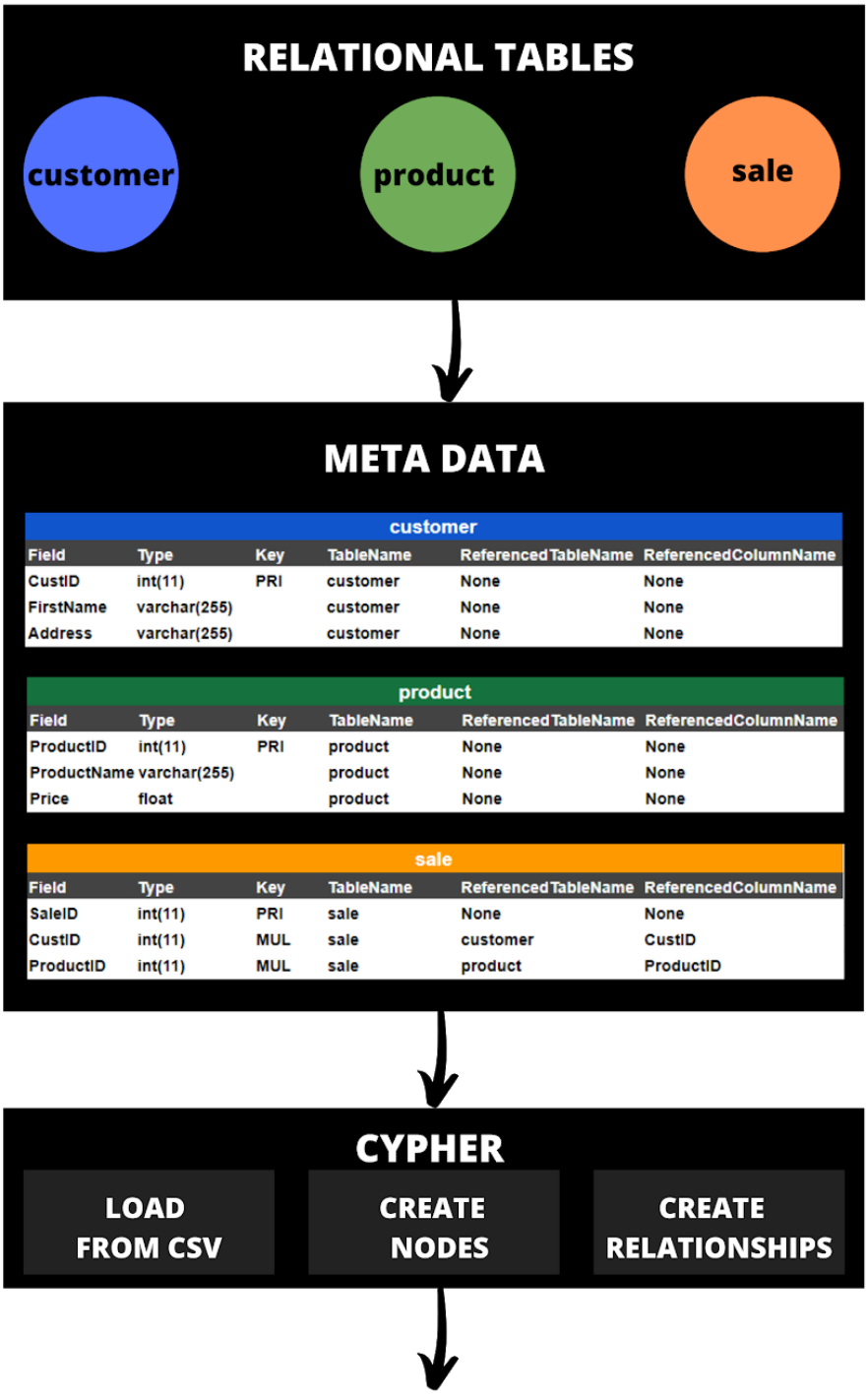


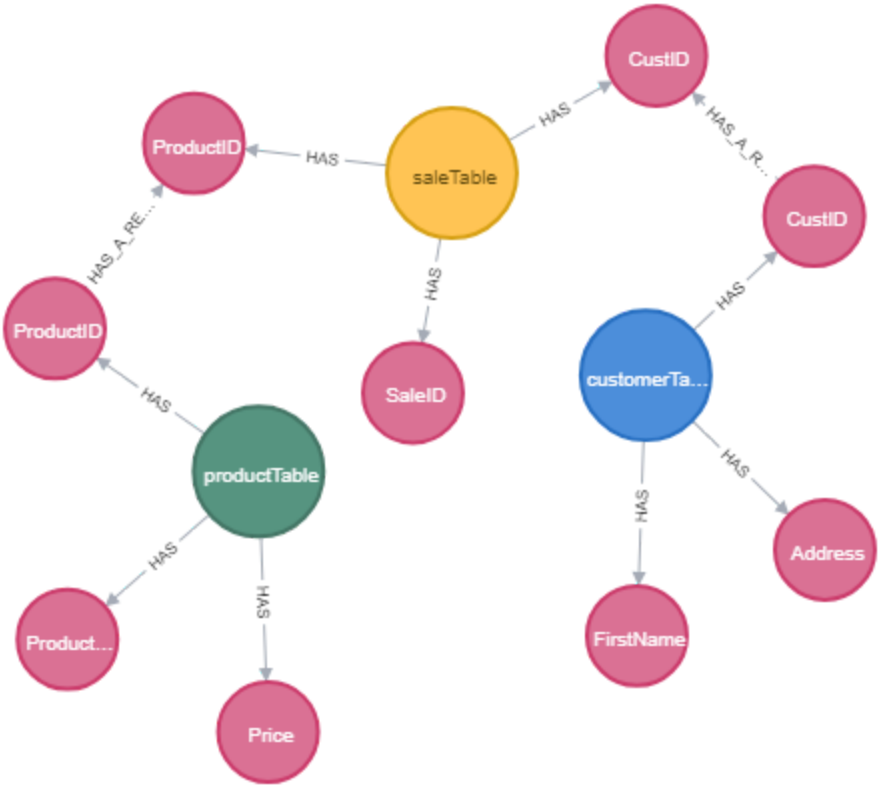
In our designing phase we came up with the following design for the data catalog which we were supposed to build. We also divided the tasks which were very very loose coupled. And the end goal was to create a utility which would be open for extension and close for modification. We decided to implement the following.


RDBMS
NON-RDBMS
SQL Server
ORACLE
MySQL
METADATA
REFERENCED TABLE DETAILS
METADATA.CSV
neo4j
FLASK WEBSERVER
D3, JAVASCRIPT, HTML

Next phase was the implementation phase and we created the following which we decided to do in our analysis and design phase. We also created a traceability matrix dashboard which had the various input we used. The purpose of that was, if a component fails or the issue arrives, we can track its successors and we can fix it.

Backend flow:



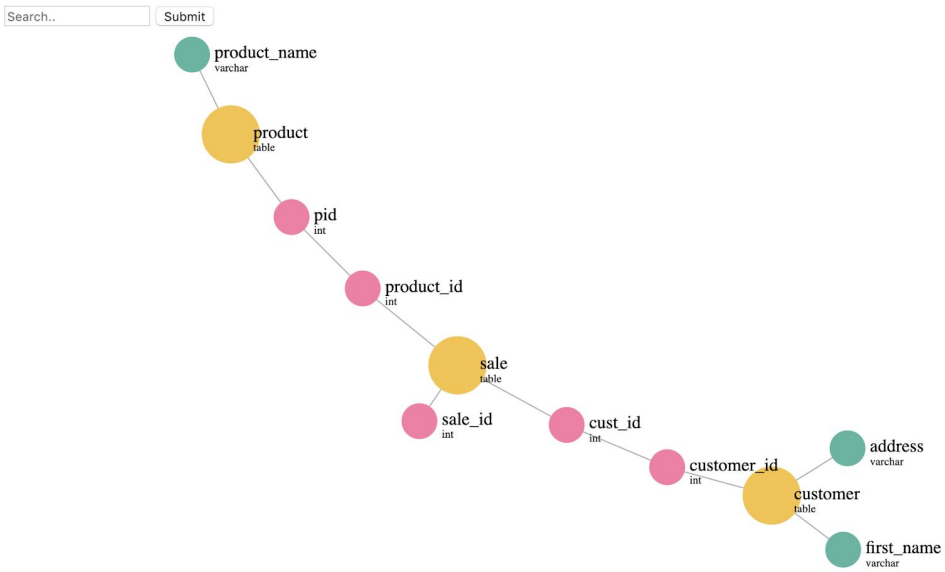


Frontend interface:

Search query format:

Attribute - attribute:<name of attribute>

Table - table:<name of table>



Our product was ready. Then our next phase was the testing phase. We tested on our side first then we passed the initial version to the client to check if the created utility fits in their environment. We took the client's input and fixed a few components the way they wanted.

The last phase, the maintenance phase was super tricky. We have already given the product. Client has started using it but the bugs. If we catch it earlier, it would be cheaper than after passing the product to the client. How to monitor, how to schedule automated processes was a challenge as well.