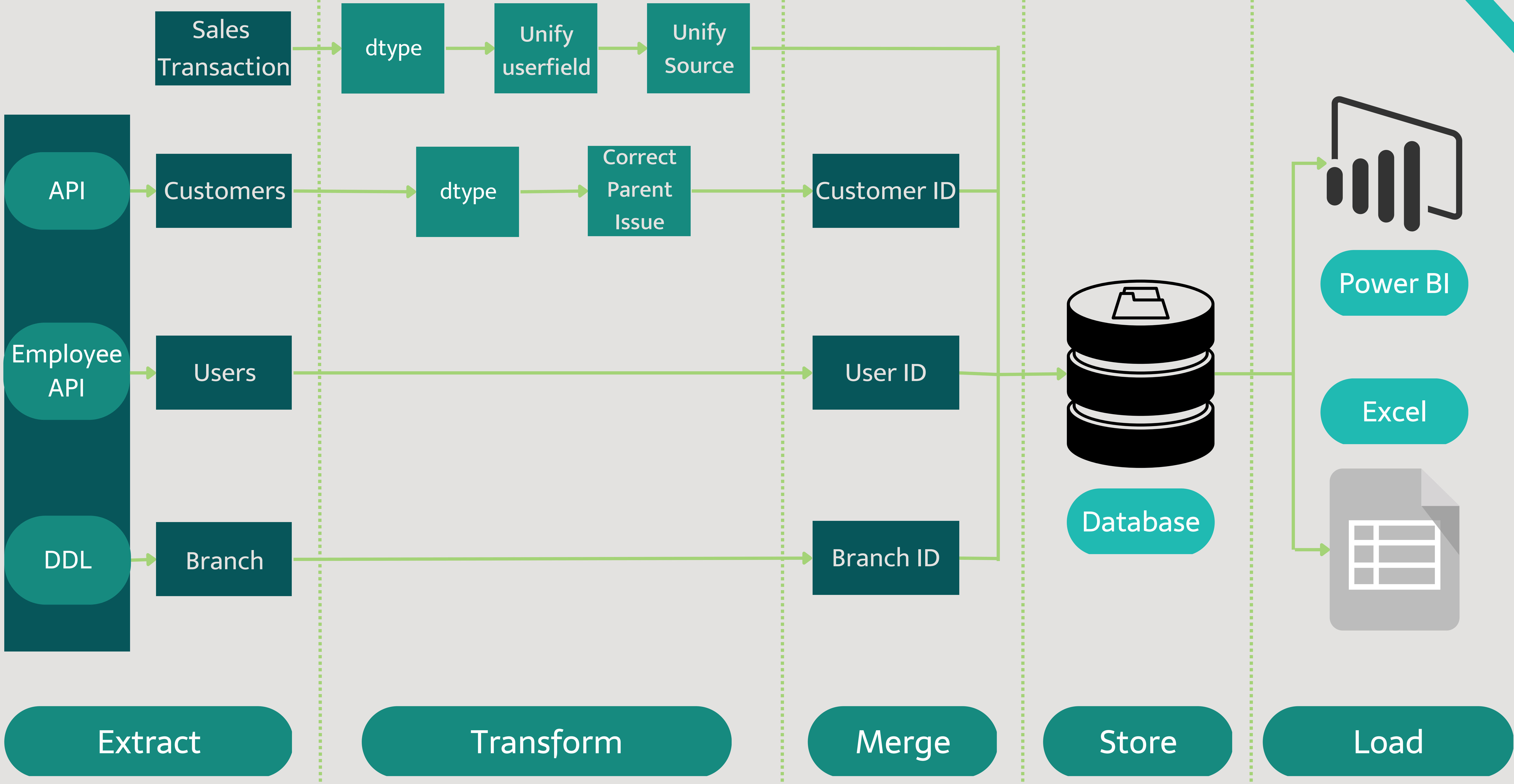
ROSE ALCOLEA

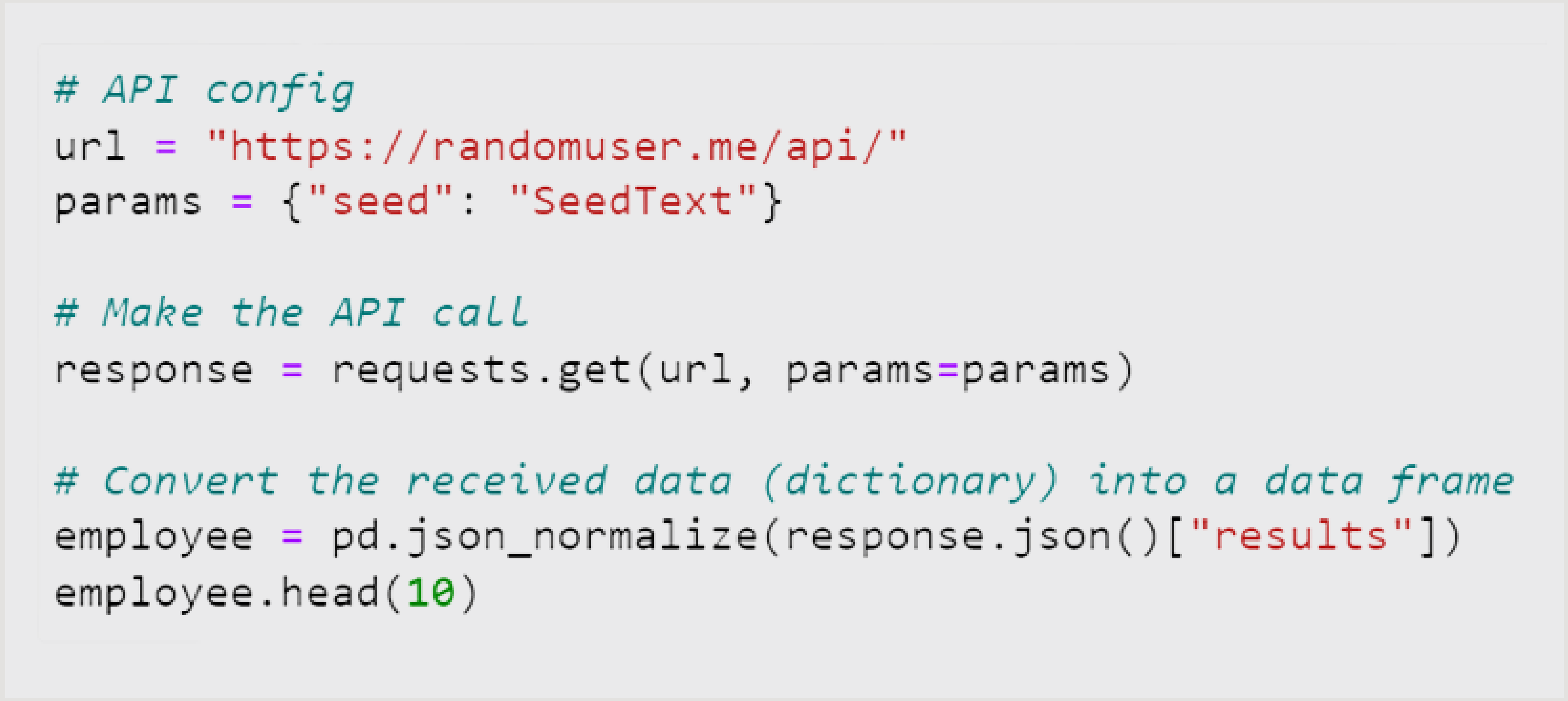
CASE STUDY

Table Of Content


Sales Transaction
dtype
Unify userfield
Unify Source
API
Customers
dtype
Correct Parent Issue
Customer ID
Employee API
Users
User ID
DDL
Branch
Branch ID
Database
Power BI
Excel
Extract
Transform
Merge
Store
Load

DDL

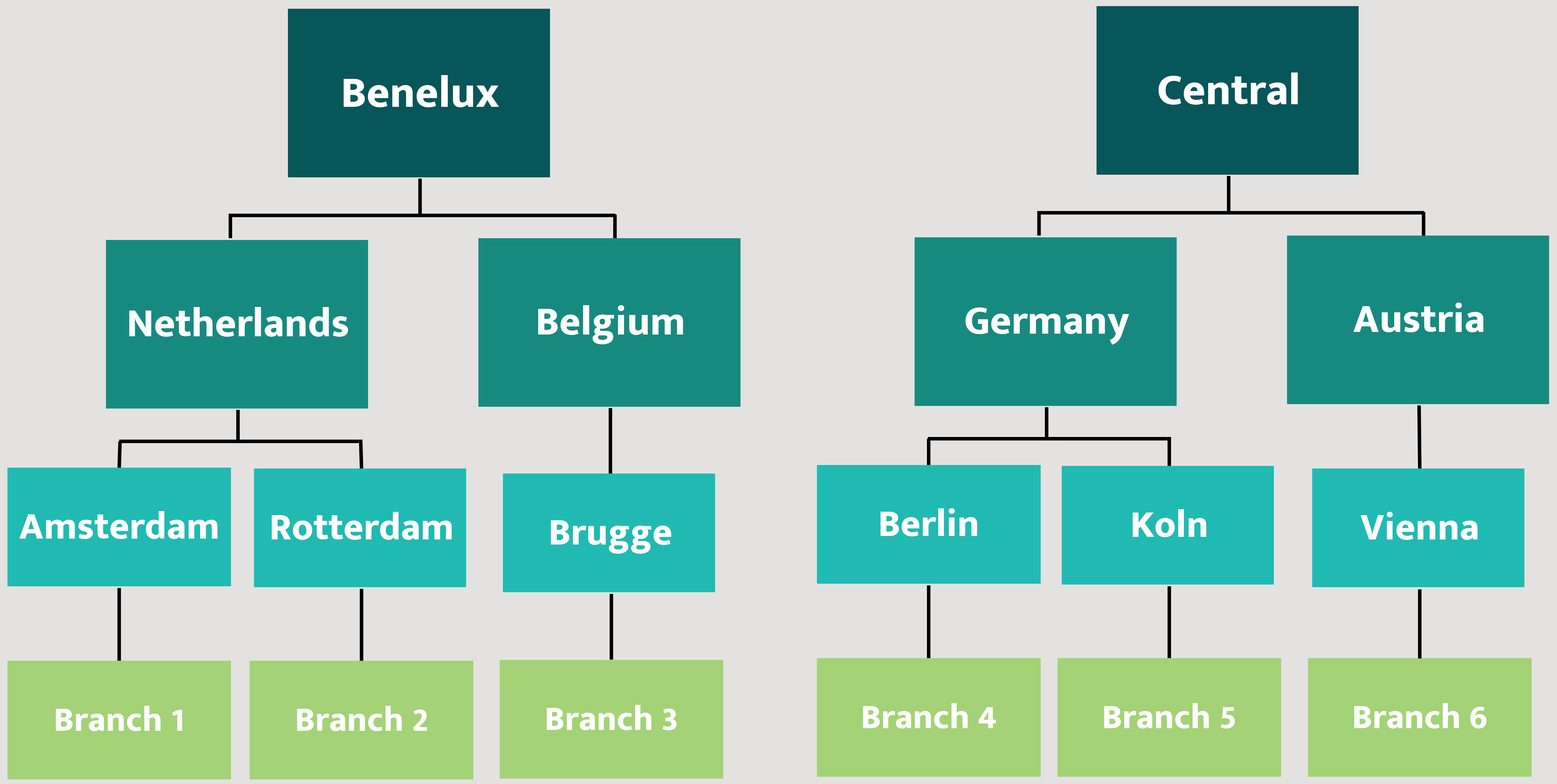
I created a DDL (Data Definition Language) statement to define the structure and properties of a branch hierarchy table. This table is used to store information about the relationship between branches.

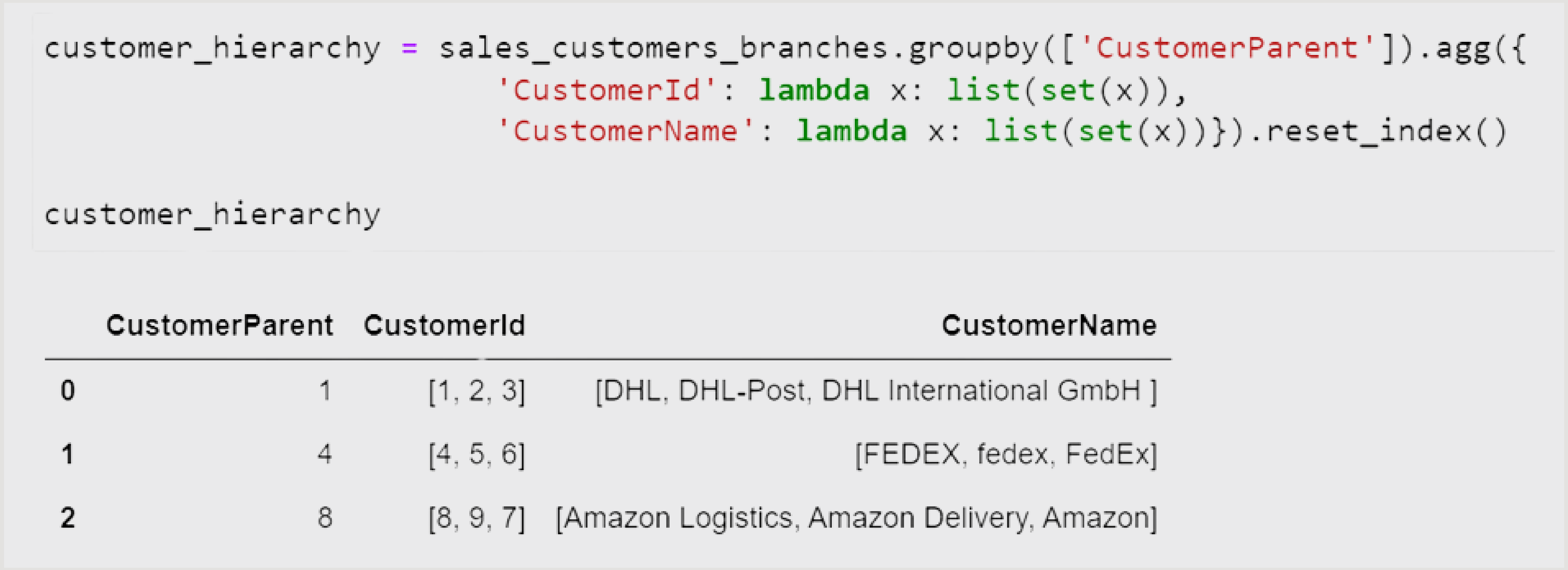


```
# API config
url = "https://randomuser.me/api/"
params = {"seed": "SeedText"}

# Make the API call
response = requests.get(url, params=params)

# Convert the received data (dictionary) into a data frame
employee = pd.json_normalize(response.json()["results"])
employee.head(10)
```


Benelux
Central
Netherlands
Belgium
Germany
Austria
Amsterdam
Rotterdam
Brugge
Berlin
Koln
Vienna
Branch 1
Branch 2
Branch 3
Branch 4
Branch 5
Branch 6



```
customer_hierarchy = sales_customers_branches.groupby(['CustomerParent']).agg({
                        'CustomerId': lambda x: list(set(x)),
                        'CustomerName': lambda x: list(set(x))}).reset_index()

customer_hierarchy
```

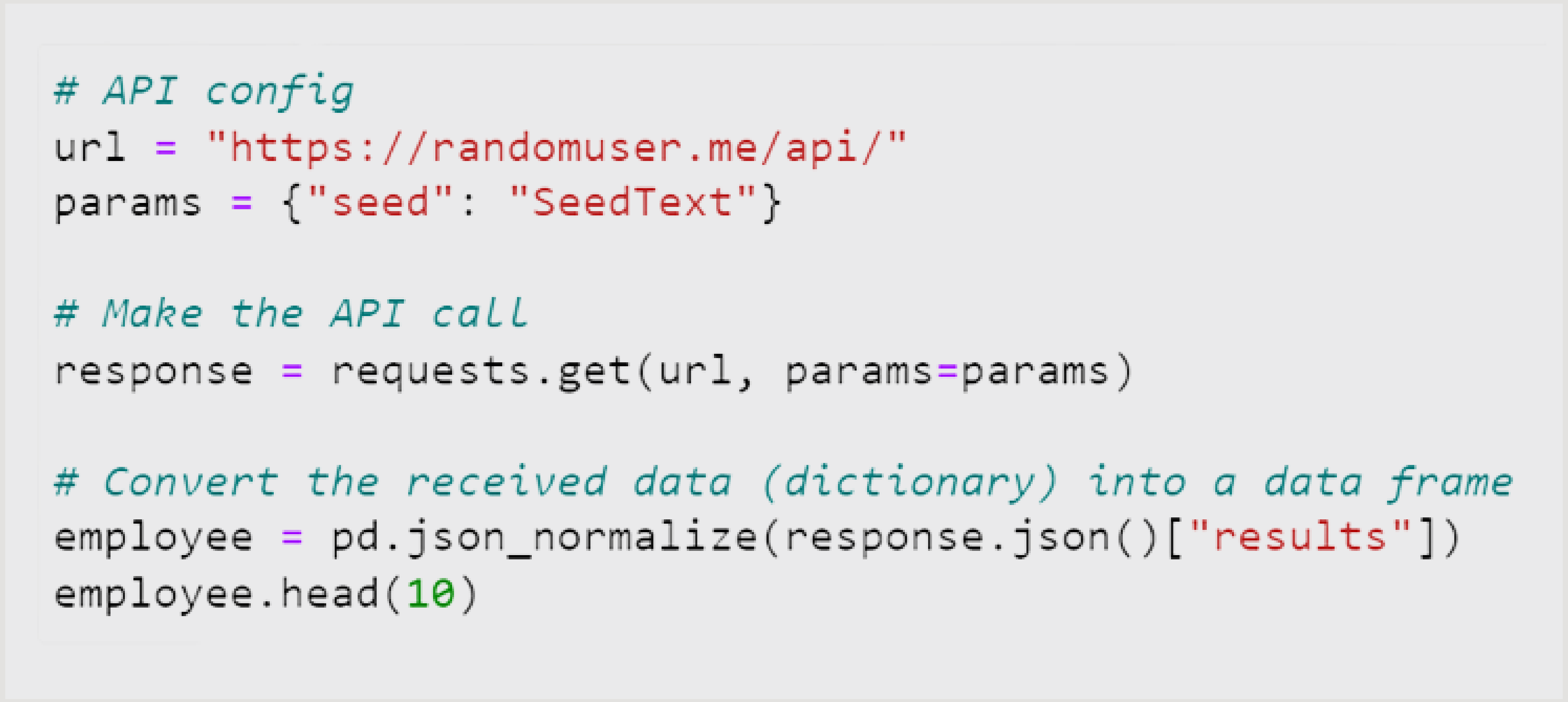
	CustomerParent	CustomerId	CustomerName
0	1	[1, 2, 3]	[DHL, DHL-Post, DHL International GmbH]
1	4	[4, 5, 6]	[FEDEX, fedex, FedEx]
2	8	[8, 9, 7]	[Amazon Logistics, Amazon Delivery, Amazon]

Customer Hierarchy

I created a Python script that generates a hierarchy table for customer data, grouping the data by customer parent and aggregating customer ID and name columns using a lambda function to display a concise and easy-to-read table of customer information.

API

I used Python to make an API call, which involved sending a request to a web server and retrieving the requested data. By using Python to interact with APIs, it becomes possible to automate data retrieval and processing.



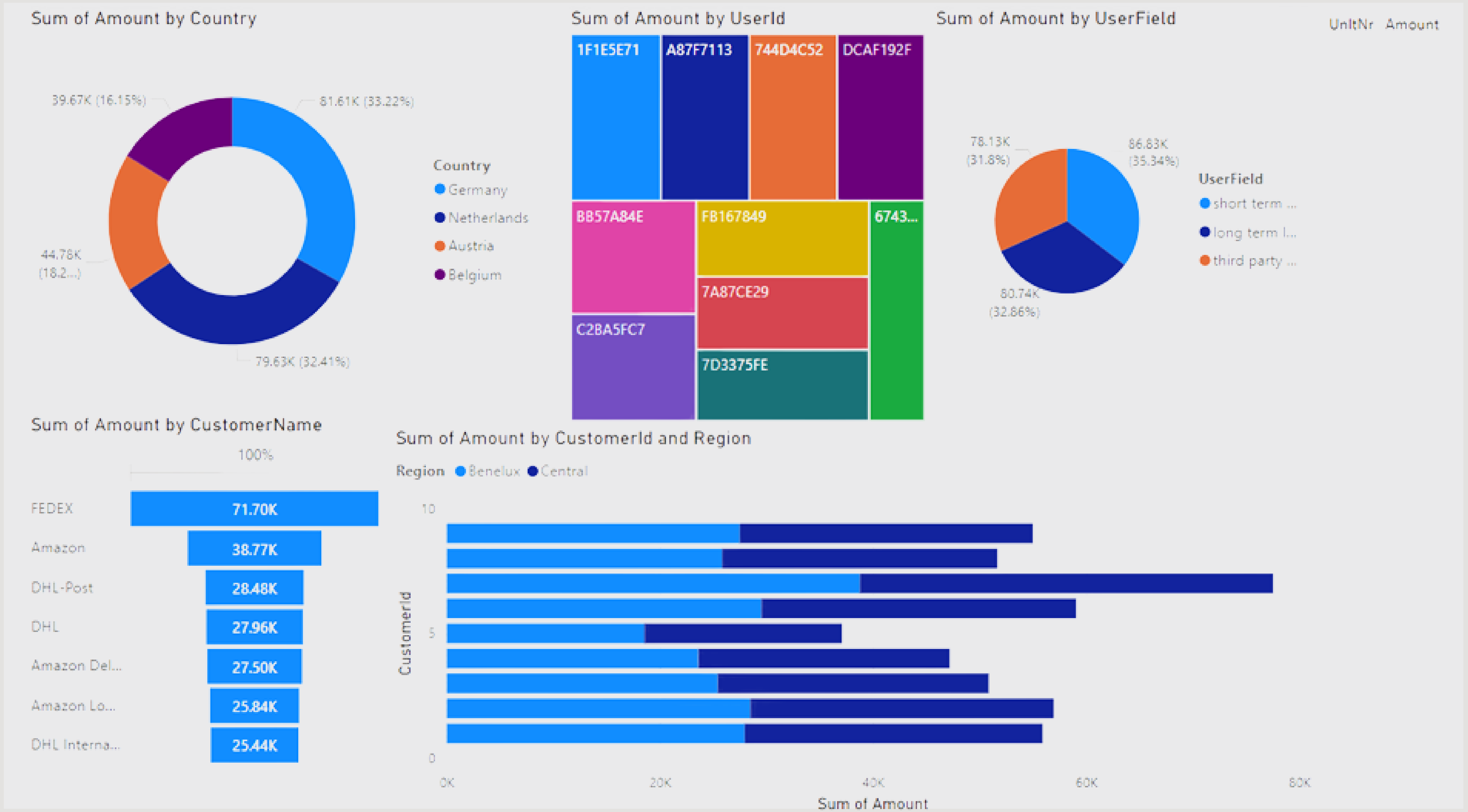
```python
# API config
url = "https://randomuser.me/api/"
params = {"seed": "SeedText"}

# Make the API call
response = requests.get(url, params=params)

# Convert the received data (dictionary) into a data frame
employee = pd.json_normalize(response.json()["results"])
employee.head(10)
```

Power BI

I created a data model/view that combines all provided data sources to display the sales amount per product, location, customer, and user. This allows for a more comprehensive and detailed understanding of sales patterns, and can be used to inform strategic decision-making in a variety of contexts.



Automation

Automate data transformation and data cleaning

Database

Set up a proper database in SQL for easy querying

Standarise Data

Agree on common terms to improve the utility of the data

THANK YOU

This project involved processing and merging data from multiple sources using Python, and creating a dashboard with clear data visualizations using Power BI. The resulting dashboard provided a comprehensive and easily accessible view of the data, allowing for quick and efficient analysis and decision-making.