YOUR SCIENCE

CREATING OPPORTUNITY

Applied Analytics
AI & ML for BI & Automation

2. Data Collection in an Industrial Setting

Your Science
Mathematical Consulting
Prof. Norbert Poncin

2025

Your Science Center: 31, Boulevard Prince Henri, L-1724 Luxembourg
Phone: +352 621 674 917, **Email**: info@yourscience.eu, **Website**: yourscience.eu
Business Permit: 10154927, **LBR**: A44409, **VAT**: LU35024328

YOUR SCIENCE

CREATING OPPORTUNITY

Applied Analytics
AI & ML for BI & Automation

2. Data Collection
in an Industrial Setting

Your Science
Mathematical Consulting
Prof. Norbert Poncin
2025

Your Science Center: 31, Boulevard Prince Henri, L-1724 Luxembourg
Phone: +352 621 674 917, **Email**: info@yourscience.eu, **Website**: yourscience.eu
Business Permit: 10154927, **LBR**: A44409, **VAT**: LU35024328

Contents

Orders Table:

Order ID	Customer ID	Order Date	Delivery Date	Total Amount
1001	1	2024-04-18	2024-04-22	49,99
1002	2	2024-04-15	2024-04-19	23,75
1003	3	2024-04-12	2024-04-16	115,98

In this bookstore's database, the relationship between the two tables is represented by the Foreign Key 'Customer ID' in the Orders Table referencing the Primary Key 'Customer ID' in the Customers Table.

2.2 SQL Databases and SQL Database Management Systems

Relational databases, i.e., structured databases with relationships between tables, are often referred to, for simplicity, as *structured* databases. They are also commonly known as SQL databases. Why this name? Special software is required to *query* these SQL databases, and this software, similar to Python, employs a specific programming *language*. The terms *structured*, *query*, and *language* combine to form the name of this programming **language**: Structured Query Language (SQL). This name further extends to the **software** used to manage these databases, referred to as the SQL database management system (SQL DBMS), and to the **files** (the databases) themselves, which are called SQL databases, as mentioned above.

So, just as you have the Python installation (**software**), the Python syntax and rules used to write code (**coding language**), and Python files (PY **files**), you also have the SQL DBMSs (**software**), the SQL query syntax (**coding language**), and SQL DBs (e.g., SQL **files**).

2.3 SQL Editors and Programmatic Access

Before visualizing the previous components schematically, we add one final actor to the scene. Just as Python **software** uses Python **code**, with code written and executed in Python editors or **interfaces** like JupyterLab

or Google Colab, SQL **software** (SQL DBMS) uses SQL **code**, with code typically written and executed in SQL editors, which provide user-friendly coding **interfaces** (bridges) between the user and the software. Alternatively, SQL code can be embedded directly in Python code and executed within a Python editor.

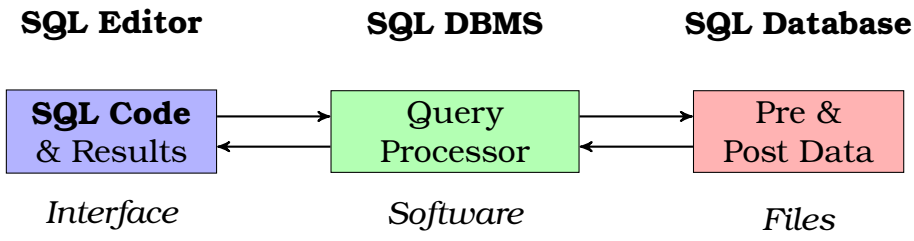

Figure 2: Workflow with SQL Editor

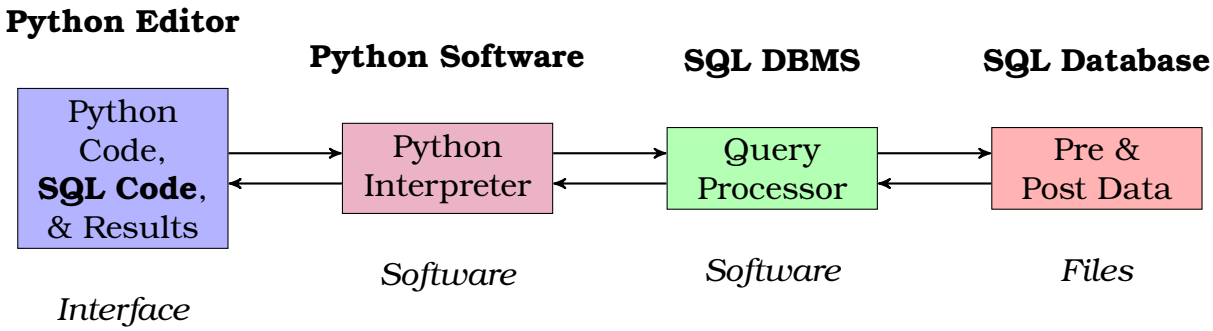

Figure 3: Workflow with Programmatic Access

We will start using SQ**Lite**, a **li**gh**t**w**e**ight **SQL** DBMS that runs locally (serverless, self-contained, ideal for single-user environments, and providing essential functionalities). While SQLite does not include a built-in SQL editor, it is supported by third-party editors such as DB Browser (graphical interface) and SQLite CLI (command-line interface). Additionally, it can be accessed programmatically through libraries like Python's **sqlite3** library (see Figure 3).

2. **Data Request and Retrieval Simulation**: Use `requests.get(url, auth=('username', 'password'))` to send a data request to the Flask API, which simulates retrieving mock customer data and returns it to the local application in JSON format (see below).
3. **Data Formatting**: The local application processes and displays this data in a DataFrame or CSV file, ensuring compatibility with production data (see previous subsection).

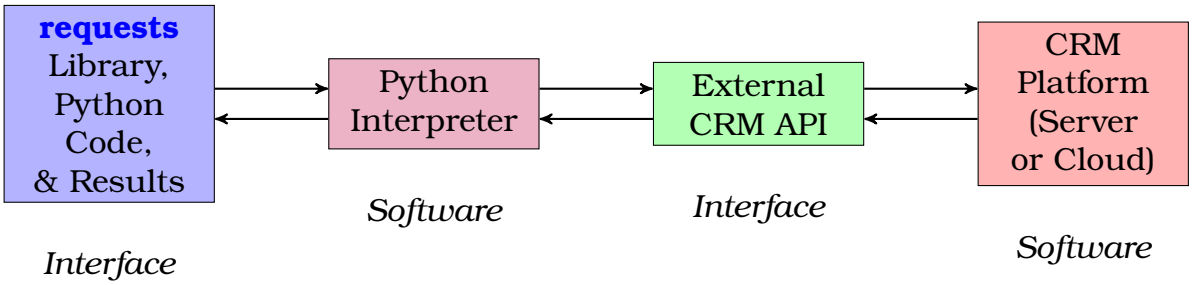

Figure 7: Workflow for Real-World CRM Setting

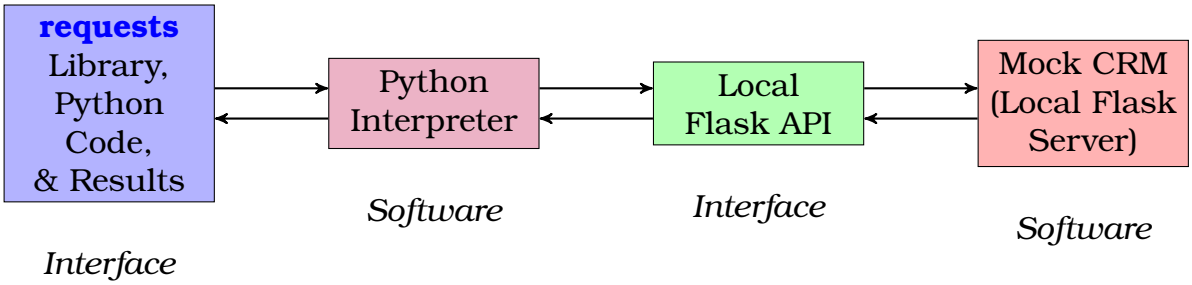

Figure 8: Workflow for Simulated CRM Setting

3.2.2 Flask Web Framework

Flask is:

- **A web framework**: A system that provides pre-built components for building *APIs and their corresponding web applications* (i.e., applications accessed via a web browser such as Google Chrome or Microsoft

```
# Seaborn is more specialized than Matplotlib, focusing on high-level statistical data visualization, while Matplotlib provides more general-purpose plotting tools

# Step 1: Load production data extracted from an SQL database

production_data_path = "C:/Users/norbert.poncin/Downloads/production_data.csv"
production_data = pd.read_csv(production_data_path)

# Step 2: Utilize customer feedback retrieved from the CRM system and
# convert satisfaction ratings into a numerical scale

# Assuming the feedback dataframe `feedback_df' is already available and contains `product_id' and satisfaction' columns

satisfaction_map = {"excellent": 5, "good": 4, "satisfactory": 3, "bad": 2}
feedback_df["satisfaction_numeric"] = feedback_df["satisfaction"].map(satisfaction_map)

# Step 3: Merge production and satisfaction data on 'product_id' and
# select only relevant numeric columns for correlation

merged_data = pd.merge(production_data, feedback_df, on="product_id")
numeric_columns = ["temperature", "pressure", "moisture", "thickness", "basis_weight", "opacity", "satisfaction_numeric"]
correlation_data = merged_data[numeric_columns]

# Step 4: Compute correlation matrix

correlation_matrix = correlation_data.corr()

# Step 5: Plot heatmap

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Matrix Heatmap")
downloads_path = 'C:/Users/norbert.poncin/Downloads'
correlation_matrix_path = os.path.join(downloads_path, "CorrMatrProdSatis.png")
```

Suggested Tasks

Write another example for each of the Python code structures we have covered so far, then explore the patterns **Indexing**, **Slicing**, and **Nested Structures** by imagining a suitable example for each:

- **Indexing**: Access elements in sequences (e.g., lists, strings) or mappings (e.g., dictionaries) using indices or keys.
  Structure: `list_name[index]` or `dictionary_name[key]`.
- **Slicing**: Extract portions of a sequence by specifying a range of indices, optionally with a step.
  Structure: `list_name[start:end:step]`.
- **Nested Structures**: Combine multiple operations, such as accessing attributes, methods, or indices, in a single expression.
  Structures: `object_name.method_name()[index]` or `dictionary_name[key].attribute_name`.

Moreover, explore the following Python patterns: `try...except...finally` for **error handling**, `if...elif...else` for **conditional statements**, `for` for **looping**, `with` for **context management**, and `yield` for **creating generators**. Provide a concise explanation and an example code snippet for each pattern.

4.2 Get Survivors of the Titanic Sinking

Perhaps you would like to explore the following Flask server and API code in more detail?

```
import pandas as pd
from flask import Flask, jsonify, request
from functools import wraps

# I. Creation of the Local Flask Server
app = Flask(__name__)  # Initialize Flask app

# II. Loading Titanic Data
# Titanic dataset can be downloaded from Kaggle or another open-
    source repository
```

5 Learning Outcomes

After working through this chapter, the reader should be able to:

- Describe the main layers of the Industrial Data Ecosystem.
- Define SQL, explain the characteristics of SQL code, describe the types of data typically stored in SQL databases, and explain the concept of relational database.
- Differentiate between an SQL DBMS, SQL Editor, SQL Database, and SQL as a language.
- Distinguish between accessing a CRM system and interacting with an SQL database. Refer to Figures 5, 6 and Figures 7, 8 to support these explanations.
- Provide examples of two SQL DBMSs, mention their corresponding Python libraries, and explain the Flask framework, the JSON format, and the concept of an API.
- Explain the notion of correlation, including correlation coefficients and matrices, and their relevance in data analysis.
- Apply essential Python code structures to perform tasks introduced in this chapter.

YOUR SCIENCE

CREATING OPPORTUNITY

About the Author

Norbert Poncin is a Luxembourgish mathematician, who was originally educated as a mathematical analyst and has worked extensively in partial differential equations (PDEs) at the University of Liège. His Master's thesis focused on the propagation of singularities in boundary value problems (BVPs) for dynamic hyperbolic systems. Applying the finite element method (FEM), his subsequent dissertation addressed BVPs for complex elliptic systems of PDEs. For his doctoral thesis, he explored mathematical quantization, while his post-doctoral education at the Polish Academy of Sciences strongly emphasized theoretical physics and its models.

Norbert has served as a Full Professor of Mathematics at the University of Luxembourg for more than 25 years and collaborated with more than 25 foreign professors and post-doctoral scholars. He has organized numerous academic events, notably approximately 10 international research meetings and over 20 research seminars focusing on theories, frameworks, concepts and models in Physics and Engineering. Beyond a substantial publication record in Differential Geometry, Algebraic Topology, and related disciplines, he has contributed roughly 25 papers to the fields of Mathematical Physics and Quantum Theory.

He was the leading instructor for over 20 university courses. Spanning a diverse spectrum of subjects, including mathematical analysis, probability theory, inferential statistics, point and solid dynamics, Lagrangian and Hamiltonian mechanics, mechanics of deformable solids, fluid dynamics, special relativity, quantum physics, geometric methods in mathematical physics, and supersymmetric models, his teaching portfolio underscores his extensive experience in applied aspects of mathematics.

In 2023, Norbert Poncin founded the mathematical consulting agency Your Science, where he currently serves as director. His primary interests include data science and artificial intelligence, along with mathematical modeling and computational science.