Published in Towards AI

Manmohan Singh Follow

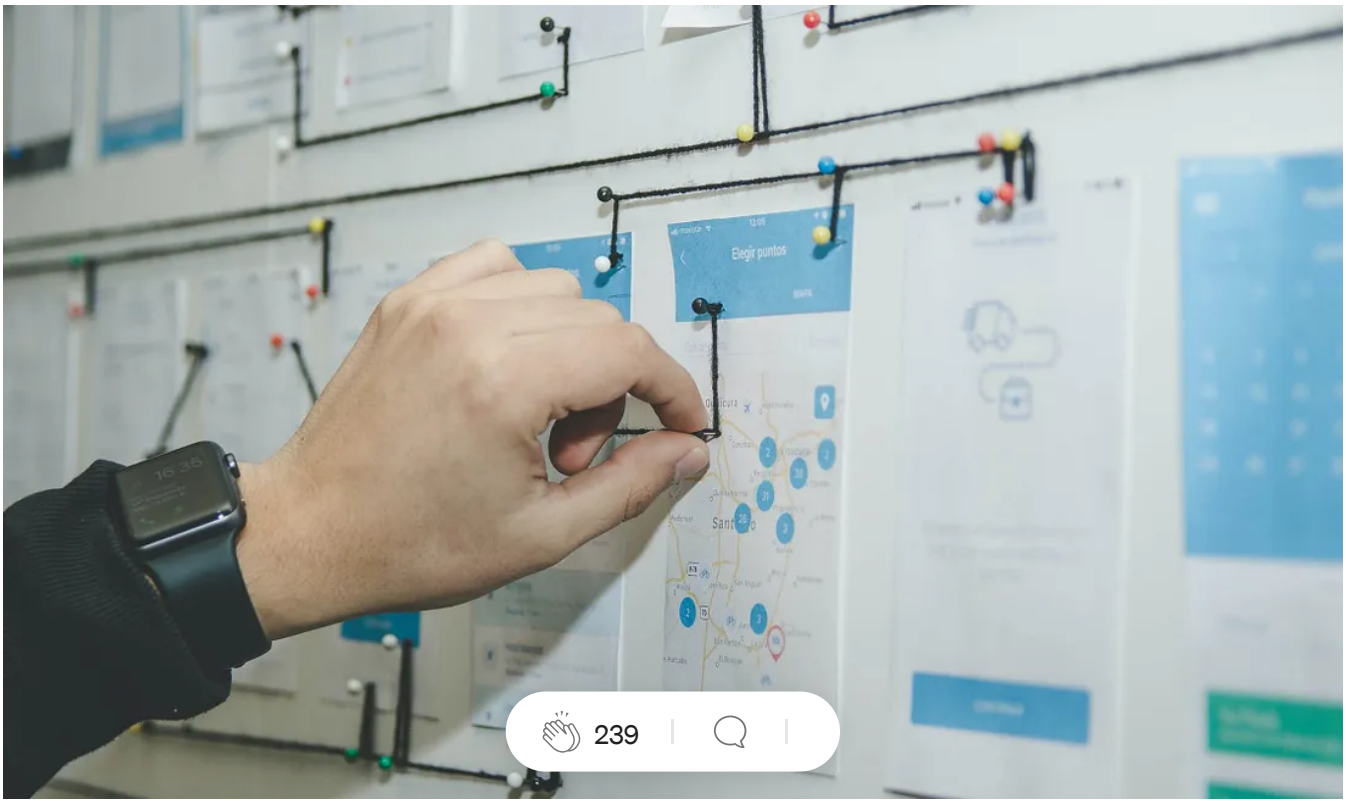
Aug 22, 2020 · 6 min read · ✦ · Listen

PROGRAMMING

Data Ingestion from 5 Major Data Sources using Python

Learn, why data stored in different sources, and how you retrieve them using python?

Stuck behind the paywall? Read this article with my friend link here.



Source

Did you know that in 2020 around 147 GB of data is generated per day? And, we have already stored around 40 trillion GB of data until now. All these stored data are not even the same. Data types like text or numbers have different formats. That explains why we have different types of data sources.

When you are working with data, you should know how to ingest the data from different sources. In this article, we are going to ingest data from various sources with the help of python libraries.

We will go through the below Data sources.

1. RDBMS Database

2. XML file format

3. CSV file format

4. Apache Parquet file format

5. Microsoft Excel

Do we have one python library which fetches data from all the sources?

Nope, because every data source has its own protocol for data transfer. We have multiple python library which does this job. Consider this article as a one-stop place to know about these python libraries.

In this article, we explain why we save data in different sources and how we retrieve data using python library.

Let's start with our data fetching story.

1. Relational database management system (RDBMS) Database

The data in RDBMS has saved in rows and columns format. Tables present in the database have a fixed schema. We can directly use Structured Query Language

(SQL) in the database to update the table. Examples of RDBMS are oracle, Microsoft SQL Server, etc.

Why we use an RDBMS database?

· Easy to use by users due to tabular format.

· A standard language SQL is available for RDBMS to manipulate data.

· The processing speed increases if we optimize RDBMS properly.

· Maintenance is easy.

· More people can access the database at the same time.

Now we can access different RDBMS databases using python libraries.

```
import pyodbc
server_name = “SQL instance of your database”
username = “username of your database”
password = “password of your database”
database_name = “name of your database”
port = “connection port for your database”

conn=pyodbc.connect(‘DRIVER={PostgreSQL ODBC Driver(UNICODE)};
                    SERVER=’+ server_name +
                    ‘;UID=’ + username +
                    ‘;PWD=’ + password +
                    ‘;DATABASE=’ + database_name +
                    ‘;PORT=’ + port + ‘;’)
cursor = conn.cursor()
cursor.execute(query)
query_data = cursor.fetchall()
```

We will be using this code in MySQL and Postgress database connection. The MySQL database connection does not need a port variable.

The contents of the Driver variable are different for different databases. Driver for MySQL and Postgress databases are SQL Server and PostgreSQL ODBC Drive(UNICODE). Also, check what types of database drivers are available in your computer. Use pyodbc.drivers() function.

Use the below code for the Oracle database.

```
import cx_Oracle

dsn_tns = cx_Oracle.makedsn(server_name,
                            port,
                            service_name=server_name)

conn = cx_Oracle.connect(user=username,
                         password=password,
                         dsn=dsn_tns)
cursor = conn.cursor()
cursor.execute(query)
query_data = cursor.fetchall()
```

Reach out to your database admins to get the values of username, password, server_name, port, service_name, and database_name variables.

2. XML file format

XML is a file extension for the External Markup Language (XML) file. It stores those textual data that is human-readable and machine-readable. XML has designed in such a way that it's format not change across the internet.

Why we use an XML file format?

· XML is a plain-text file format which can be understood by both human and machine.

· XML has a simple and common syntax rule to exchange information between applications.

· We can use a programming language to manipulate the information inside the XML file.

· We can combine multiple XML documents to form one large XML file without adding extra information. You can also divide XML into various parts and use them separately.

· The XML file format is preferable in web applications.

Now, we can access the XML file using the xml library.

```
import pandas as pd
import xml.etree.ElementTree as etree
```

```
xml_tree = etree.parse(“sample.xml”)
xml_root = xml_tree.getroot()
columns = [“A”, “B”]

datatframe = pd.DataFrame(columns = columns)
for node in xml_root:
    name = node.attrib.get(“A”)
    mail = node.find(“B”).text if node is not None else None
    datatframe = datatframe.append(pd.Series([A,B], index=columns),
                                   ignore_index = True)
```

You can use the request library to post the XML file in SOAP API.

3. CSV file format.

A Comma Separated Values (CSV) is a file format that stores plain text and tabular data. The first line of CSV file generally contains the columns and comma separate each column. Second-row and onwards have contents of the columns. It could be a text, number, or date. Tab Separated values file also has .csv file extension. It solves column separation issues related to CSV file format.

Why we use the CSV file format?

· Easy to create and manipulate data.

· Easy to read and understand data.

· We can organize a large amount of data.

· We can easily import and export CSV files.

Now we can access CSV files using pandas and the CSV library.

With the help of the pandas library, you can directly import the CSV file into the dataframe.

```
# importing Pandas library
import pandas as pd
csv_dataframe = pd.read_csv(“hr_data.csv”, sep=”,”,)
print(csv_dataframe)

Name Hire Date Salary Sick Days remaining
0 Graham Chapman 03/15/14 50000.0 10
1 John Cleese 06/01/15 65000.0 8
2 Eric Idle 05/12/14 45000.0 10
3 Terry Jones 11/01/13 70000.0 3
```

```
4 Terry Gilliam 08/12/14 48000.0 7
5 Michael Palin 05/23/13 66000.0 8
```

If CSV file has '\t' separator then use sep="\t". In case of space use sep=" ". Visit here for more information about read_csv function.

```
import csv

with open("hr_data.csv") as csv_file:
    csv_reader = csv.reader(csv_file, delimiter=',')
    line_count = 0
    for row in csv_reader:
        if line_count == 0:
            print(f'Column names are {", ".join(row)}')
            line_count += 1
        else:
            print(f'\t{row[0]} first column content {row[1]} second
                  row content {row[2]} third row content.')
            line_count += 1
print(f'Processed {line_count} lines.')
```

4. Apache Parquet file format

Apache Parquet is a column-oriented data storage file format. The data stored in parquet files have compressed efficiently. Shredding and assembly algorithms are used in parquet to store the data. It is enhanced to handle complex data in bulk. Generally, parquet formats are useful in big data technologies.

Why we use the apache parquet file format

· The parquet file created using an efficient compression algorithm that saves a lot of storage space than other file formats.

· Queries that fetch columns data need not scan the whole row, which improves performance.

· Each column has its own encoding techniques.

· Parquet files have optimized for queries that process a large amount of data.

Now we can access parquet files using pandas and pyarrow libraries.

```
import pyarrow.parquet as pq

example_table = pq.read_pandas('example.parquet',
                               columns=['one','two']).to_pandas()

print(example_table)
```

```
one two
a foo bar
b bar baz
c baz foo
```

```
import pandas as pd

pandas_dataframe = pd.read_parquet('example.parquet',
                                   engine='pyarrow')

print(pandas_dataframe)
```

```
one two
a foo bar
b bar baz
c baz foo
```

5. Microsoft Excel

Excel is a spreadsheet developed by Microsoft. It stores data in tabular format. It has a grid of cells, which form rows and columns when combined. It has a lot of inbuilt features such as calculations, graphing tools, pivot tables, etc.

Why we use Microsoft Excel file format?

· You can analyze the data in excel using charts and graphs.

· Excel is good at sorting, filtering, and searching in data.

· You can build a mathematical formula and apply it to the data.

· Excel comes with a password-protected feature.

· You can use excel as a calendar.

· You can also use excel to automate data-related jobs.

Now, we can access Microsoft Excel using openpyxl library.

```
from openpyxl import load_workbook
```

```
workbook = load_workbook(filename=”sample.xlsx”)
workbook_sheets = workbook.sheetnames
sheet = workbook.active

print(sheet[“A1”].value)

“hello”

print(sheet.cell(row=10, column=6).value)

“this is hello world store in row 10 and column 6.”

import pandas as pd

df = pd.read_excel(‘File.xlsx’, sheetname=’Sheet1')

print(“Column headings:”)
print(df.columns)

[‘A’,’B’,’C’]
```

Conclusion

This article helps you to understand why we need different sources to store data and how you retrieve data from these sources. We have used multiple python libraries to ingest data. In this article, I have covered 5 data sources.

Hopefully, this article will help you in data processing activities.

Other Articles by Author

1. First step in EDA : Descriptive Statistic Analysis
2. Automate Sentiment Analysis Process for Reddit Post: TextBlob and VADER
3. Discover the Sentiment of Reddit Subgroup using RoBERTa Model

Python Programming Data Ingestion Rdbms Parquet Big Data

We have moved our newsletter to: ws.towardsai.net/subscribe Take a look.

By signing up, you will create a Medium account if you don't already have one. Review our Privacy Policy for more information about our privacy practices.

Get this newsletter