



UNIVERSITY
OF WOLLONGONG
IN DUBAI

Assignment Cover Sheet

Subject Code: ISIT312

Subject Name: Big Data Management

Submission Type: Group report

Assignment Title: Data Lake Implementation

Student Name: Mahek Sajid, Mohammed Maarij Uddin Khan, Rida Fatimah Asif, Rohan Sudhir, Samir Fazil, Kianoush Rahrvan, Reyon Noronha, Ibrahim Siddiqui, Nageen Irfan Minhas

Student Number: 7068293, 6222560, 7057167, 6866219, 7068943, 5840454, 7427359, 7352396, 6997053

Student Phone : +971 58 282 8467, +971 50 696 6480, +971 58 996 4185, +971 50 877 2853, +971 50 238 4958, +971 50 141 5119, +971 56 972 1277, +971 50 303 9378, +971 54 303 0358

Student Email: mshs511@uowmail.edu.au, mmuk953@uowmail.edu.au, Rfa373@uowmail.edu.au, rs638@uowmail.edu.au, sf880@uowmail.edu.au, Kr704@uowmail.edu.au, rkn914@uowmail.edu.au, las677@uowmail.edu.au, nim591@uowmail.edu.au

Lecturer Name: Dr Patrick Mukala

Due Date: 22 / 6 / 2023

Date Submitted: 22 / 6 / 2023

PLAGIARISM:

The penalty for deliberate plagiarism is FAILURE in the subject. Plagiarism is cheating by using the written ideas or submitted work of someone else. UOWD has a strong policy against plagiarism. The University of Wollongong in Dubai also endorses a policy of non-discriminatory language practice and presentation.

DECLARATION:

I/We certify that this is entirely my/our own work, except where I/we have given fully-documented references to the work of others, and that the material contained in this document has not previously been submitted for assessment in any formal course of study. I/we understand the definition and consequences of plagiarism.

PLEASE NOTE: STUDENTS MUST RETAIN A COPY OF ANY WORK SUBMITTED

Signature of Student: Mahek, Maarij, Rida, Rohan, Samir, Kianoush, Reyon, Ibrahim, Nageen

Optional Marks:

Comments:

Lecturer Assignment Receipt (To be filled in by student and retained by Lecturer upon return of assignment)

Subject:

Student Name:

Due Date:

Signature of Student:

Assignment Title:

Student Number:

Date Submitted:

Student Assignment Receipt (To be filled in and retained by Student upon submission of assignment)

Subject:

Student Name:

Due Date:

Signature of Lecturer

Assignment Title:

Student Number:

Date Submitted:

Table of Contents

Executive summary	3
The Problem	3
Technical requirements	3
Main constructs (aspects/parts)	4
Implementation	7
Limitations/Difficulties and Constraints (if any)	11
Conclusion	11
Team Contribution Worksheet	11

Executive summary

This report details our project with FortyGuard, a company that provides outdoor weather data analysis services. They required a solution to address a batch processing issue with their vast weather data. Our team responded by implementing a comprehensive solution leveraging Microsoft Azure's suite of tools, including Azure Data Factory, Azure Data Lake Storage Gen2, Azure SQL server, and Azure Databricks.

Despite some initial technical difficulties, our team successfully built a pipeline and data lake to efficiently manage both structured and unstructured data. We automated batch processing, which the FortyGuard representative found novel and intriguing. The implemented solution offered scalability, security, and efficiency in managing large datasets and could be extended to more complex operations in the future.

Our practical engagement has enriched our understanding of big data management and its impact on AI solutions. It also highlighted the potential for future exploration of real-time data processing and machine learning logic. Overall, our project underscores the vast potential of cloud computing for big data management and offers a blueprint for handling similar challenges in the future.

The Problem

The company we approached was FortyGuard. Founded in 2020, they provide expertise in outdoor weather data analysis. They offer data science and predictive analysis solutions using their 'FortyEngine' which is their in-house AI cloud based platform. The problem given to us was to address a batch processing issue with their data. They have data sources that provide monthly summaries of weather data and need a pipeline to handle this data stream

Technical requirements

We used Microsoft Azure to implement our proposed solution. It hosts a variety of services that we linked together.

Data factory

Azure Data Factory is a data integration solution offered by Azure. It allows users to construct, schedule, and manage data workflows for moving, transforming, and processing data across on-premises and cloud-based data sources.

Azure Data Lake Storage Gen2

A data lake framework that offers directory management and container specifications that can host unstructured data inputs. We used a storage account to create blobs for our unstructured data.

Azure SQL server

Azure SQL Server is a fully managed relational database service provided by Microsoft on the Azure cloud platform, offering high availability, scalability, and security. It allows organisations to store, manage, and access their structured data using the familiar SQL language and tools.

Azure Databricks

Azure Databricks is an analytics platform built on Apache Spark that is quick, scalable, and collaborative. It is offered as a fully managed service on the Azure cloud infrastructure. It connects with a number of Azure services and offers a single analytics platform for data scientists, engineers, and analysts to work together on advanced analytics, data processing, and data exploration tasks.

Main constructs (aspects/parts)

Aspect 1: Motivation.

The problem requires us to create a pipeline equipped to deal with large weather datasets. A data lake would be an appropriate solution to contain this data. From our conversations with the company representative, we understood the following about their data:

1) How much data they collect

FortyGuard collects an average of 500 thousand data points over a period of 25-30 days, from various sources, such as Google API, satellites, sensors and official bodies of government and weather stations. It is collected in a variety of file formats that contain structured data as well as unstructured data. Structured data includes json files and its size is around 1 KB and the csv files are 2MB.

2) How they collect it

FortyGuard makes use of sensors, and online and open projected sources making use of the APIs, to gather their data. Weather stations collect various types of data to monitor and understand atmospheric conditions, which is also gathered as one of the data source points in FortyGuard

3) Where they store it

Data is stored locally on hard drives, AWS storage and Google Cloud which is accessible via APIs.

4) How they process it

Using batch processing as well as stream processing. Stream processing is done at regular intervals of 5 minutes when weather records are updated at the sources to trigger the listener mechanism set up in FortyGuard. Batch processing takes place monthly. Also, data was processed using satellite access. Data was also collected from the USA in the form of csv files. The company also used historical data.

5) What is used for integration

A cloud based AI powered engine is used in combination with AWS services for data management. FortyEngine gathers data from global sources, transforms and uses the data with dynamic AI analytics and then provides solutions based on different use cases.

6) Potential issues

When it comes to the APIs, the current API used by the company returns rows of data but instead, the company wants the API to return data analytics, graphs with specific tags etc. Other issues that the company have faced include: the sources of data they are using as well as batch processing.

Moreover, we also got information from their company in regards to the 5 Vs of big data. These include:

1. Veracity

In regards to the accuracy of the data, the company computes the data between up to 1m² points in between local sensors per minute. This data is made sure to be compliant and secure.

2. Velocity

Forty Guard is built atop huge amounts of data which is continuously increasing with every day weather readings and forecasts. These readings are produced for a large number of locations, leading to the company gathering as many data points as possible in order to collect information

3. Variety

The company uses unstructured data which is converted to csv files. They also make use of tiles that are an image of a location such as a weather station. The data varies due to the diversities of different locations.

4. Volume

As mentioned above, the company stores data from numerous different locations, this data is stored in the form of 1 KB json files as well as 2 MB CSV files.

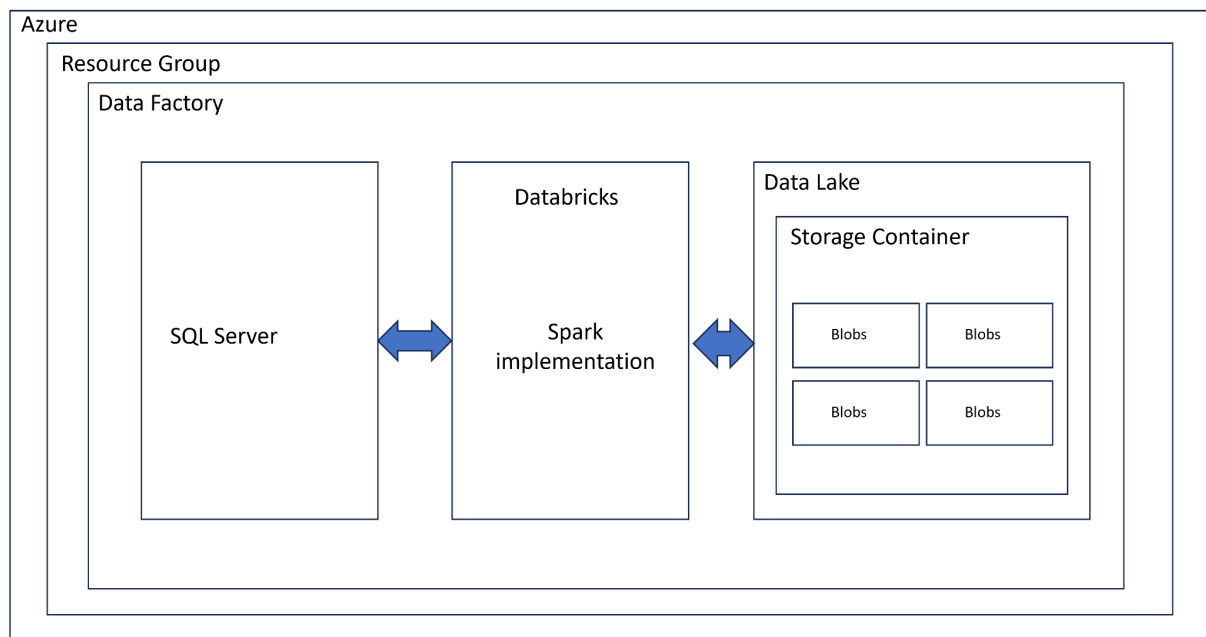
5. Value

The data stored is used in a number of functionalities including for research purposes, predicting cooler routes for users, training and modelling, and producing

constantly changing 2D and 3D dynamic maps.

Aspect 2: Solution Logic

We implemented a solution using Azure architecture. The following diagram provides an overview of our solution design.



This multilayer solution uses multiple services provided by Azure. The data factory is the main area of focus. We have implemented a sql server and a data lake to intake structured and unstructured data. They are then linked to Azure Databricks which allows us to program specific operations on the data using python and spark.

Using spark, we can code in python to implement the necessary data operations. In this scenario, it may be predictive modelling, aggregations or transformations of the data for dashboarding. For example purposes, we ran a simple operation of calculating aggregate data for multiple weather stations over a single day.

Azure Data Factory has a trigger function (executes the pipeline according to a schedule) that can ensure that this operation runs once a day, hence implementing batch processing. Furthermore, the trigger can be specified to run at smaller intervals in the case of stream processing data.

Aspect 3: Solution Evaluation and Lessons learned.

Our solution is a primitive proof of solution which can be expanded to a much more complex process. Future iterations can implement predictive analysis using multiple types of data sources (real time) and machine learning logic. We learnt that big data management is highly relevant to real time AI solutions. We understood the scope of

data management that is required to run an industry standard consulting agency.

The theory we learnt in class was implemented via Databricks where we used spark to code scala and python implementations of data access and manipulation. We had the opportunity to also learn about services that can be used for real world implementations (the Azure framework).

The company representative admired our effort to understand and solve this problem. Our approach of using Data Factory to automate batch processing was interesting and novel to them. They have encouraged us to next consider how to process stream data.

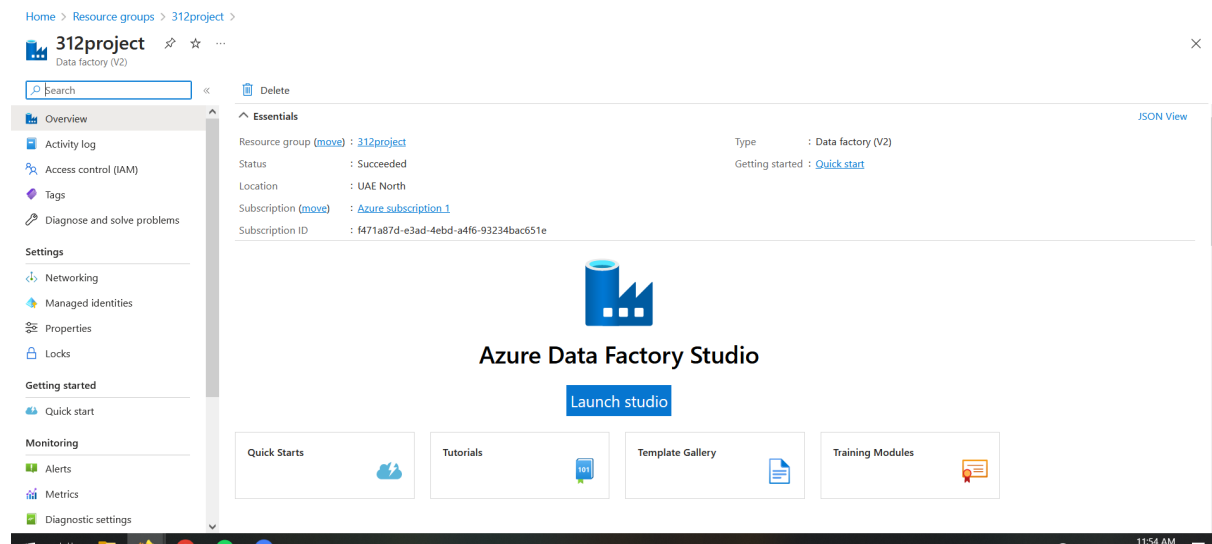
Implementation (Azure Screenshots):

Resources Used:

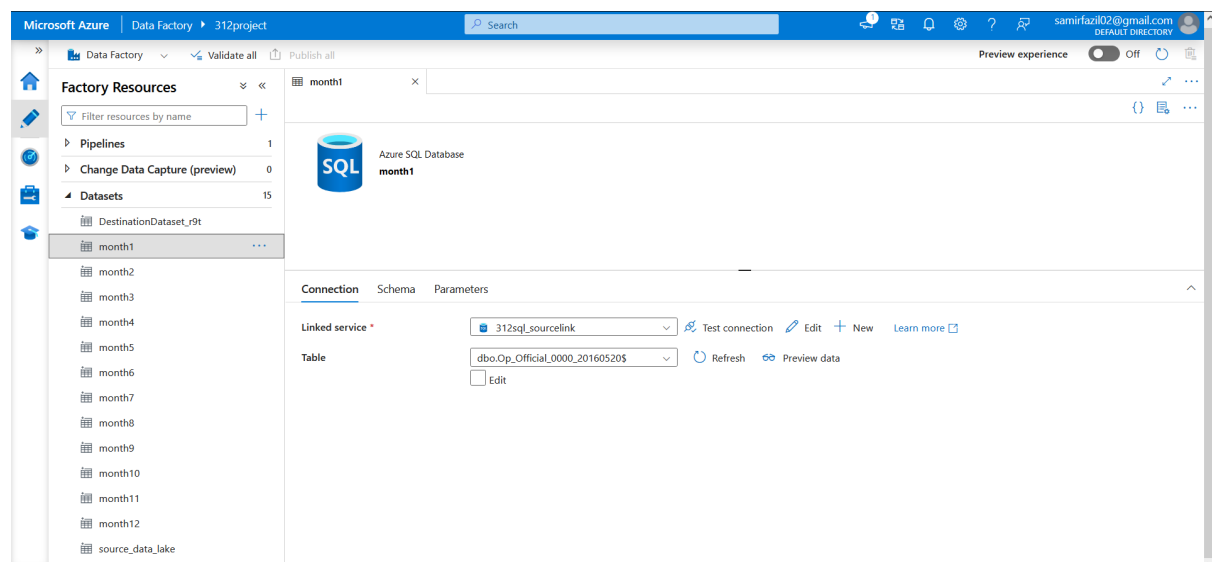
The screenshot shows the Azure portal interface for a resource group named '312project'. The left sidebar contains navigation options like Overview, Activity log, Access control (IAM), Tags, Resource visualizer, Events, Settings, Deployments, Security, Policies, Properties, Locks, Cost Management, Cost analysis, Cost alerts (preview), and Budgets. The main area displays the 'Essentials' tab with a 'Resources' section. A table lists the resources within the group:

Name	Type	Location
312datalake	Storage account	UAE North
312project	Data factory (V2)	UAE North
312project	SQL server	UAE North
arm	API Connection	UAE North
BatchProcessing	Azure Databricks Service	East US
office365	API Connection	UAE North
project312 (312project/project312)	SQL database	UAE North

Data Factory:



Datasets Used:



Pipeline used:

Pipeline runs

Triggered

Debug

Rerun

Cancel options

Refresh

Edit columns

List

Gantt

Filter by run ID or name

Abu Dhabi, Muscat (... : Last 24 hours

Pipeline name : source_pipeline

Status : All

Copy filters

Export to CSV

Runs : Latest runs

Triggered by : All

Add filter

Showing 1 - 1 items

Last refreshed 0 minutes ago

<div><div></div></div> <div>Pipeline name</div> <div>↕</div>	<div><div></div></div> <div>Run start</div> <div>↕</div>	<div><div></div></div> <div>Run end</div> <div>↕</div>	<div><div></div></div> <div>Duration</div>	<div><div></div></div> <div>Triggered by</div>	<div><div></div></div> <div>Status</div> <div>↕</div>	<div><div></div></div> <div>Run</div>	<div><div></div></div> <div>Parameters</div>
<div><div></div></div> <div>source_pipeline</div>	6/22/2023, 2:17:28 AM	6/22/2023, 2:17:51 AM	24s	Manual trigger	<div><div></div></div> <div>Succeeded</div>	Original	<div><div></div></div> <div></div>

Datalake With source data:

Home > 312datalake

312datalake | Containers

Storage account

Search

+ Container Change access level Restore containers Refresh Delete Give feedback

Search containers by prefix

Show deleted containers

Name	Last modified	Public access level	Lease state
<input type="checkbox"/> \$logs	6/22/2023, 2:07:17 AM	Private	Available
<input type="checkbox"/> sourcedata	6/22/2023, 2:08:40 AM	Container	Available

Overview

Activity log

Tags

Diagnose and solve problems

Access Control (IAM)

Data migration

Events

Storage browser

Data storage

Containers

File shares

Queues

Tables

Security + networking

Networking

Access keys

Source Data:

Home > 312datalake | Containers >

sourcedata

Container

Search

Upload Add Directory Refresh Rename Delete Change tier Acquire lease Break lease Give feedback

Search blobs by prefix (case-sensitive)

Show deleted objects

Authentication method: Access key (Switch to Azure AD User Account)

Location: sourcedata

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
<input type="checkbox"/> dboOp_Official_0000_20160520%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.87 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20160604%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.9 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20160731%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.86 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20160818%24source csvs	6/22/2023, 2:17:41 AM	Hot (Inferred)		Block blob	5.89 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20160914%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.92 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20161023%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.89 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20161102%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.89 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20161229%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.96 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20170102%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.93 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20170219%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.89 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20170331%24source csvs	6/22/2023, 2:17:48 AM	Hot (Inferred)		Block blob	5.86 MiB	Available
<input type="checkbox"/> dboOp_Official_0000_20170430%24source csvs	6/22/2023, 2:17:47 AM	Hot (Inferred)		Block blob	5.85 MiB	Available

Overview

Diagnose and solve problems

Access Control (IAM)

Settings

Shared access tokens

Manage ACL

Access policy

Properties

Metadata

Sql Server with Database:

Home > Resource groups > 312project > 312project >

project312 (312project/project312)

SQL database

Search

Copy Restore Export Set server firewall Delete Connect with... Feedback

Essentials

Resource group (move) : 312project

Status : Online

Location : UAE North

Subscription (move) : Azure subscription 1

Subscription ID : f471a87d-e3ad-4ebd-a4f6-93234bac651e

Tags (edit) : Click here to add tags

Server name : 312project.database.windows.net

Elastic pool : No elastic pool

Connection strings : Show database connection strings

Pricing tier : Basic

Earliest restore point : 2023-06-21 19:18 UTC

Getting started Monitoring Properties Features Notifications (0) Integrations Tutorials

Start working with your database

Connect to your database and start working with data with a few simple steps. Learn more

Configure access

Connect to application

Start developing

Overview

Activity log

Tags

Diagnose and solve problems

Getting started

Query editor (preview)

Settings

Compute + storage

Connection strings

Properties

Locks

Data management

Replicas

Sync to other databases

Integrations

Azure Synapse Link

Azure Databricks for running spark and python:

Microsoft Azure databricks

Search data, notebooks, recents, and more...

CTRL + P

BatchProcessing

🔔

🔒

samirfazi102@gmail.com

Batch Processing Code Example Python

File Edit View Run Help Last edit was 10 hours ago Provide feedback

▶ Run all

■ Terminated

📅 Schedule

Share

Cmd 1

```
1 server_name = "312project.database.windows.net"
2 database_name = "project312"
3 username = "project312"
4 password = "Dfsf_1234"
5
6 jdbc_url = f"jdbc:sqlserver://{312project.database.windows.net:1433;database=project312;user=project312;password=Dfsf_1234"
7 connection_properties = {
8     "driver": "com.microsoft.sqlserver.jdbc.SQLServerDriver",
9     "url": jdbc_url,
10    "dbtable": "dbo.Op_Official_0000_20160520$"
11 }
12
13 df = spark.read.format("jdbc").options(**connection_properties).load()
```

df: pyspark.sql.dataframe.DataFrame = [station_number: double, area_code: string ... 9 more fields]

Command took 2.94 seconds -- by rohan_sudhir@samirfazi102gmail.onmicrosoft.com at 6/22/2023, 1:43:27 AM on Rohan Sudhir's Cluster

Cmd 2

```
1 from pyspark.sql import functions as F
2
3 # Filter dataframe where parameter is 'MaxT'
4 maxT_df = df.filter(df['parameter'] == 'MaxT')
5
6 # Calculate max value for each area code
7 maxT_values = maxT_df.groupBy('area_code').agg(F.max('value'))
8
9 print('Maximum values for MaxT parameter per station:')
```

Batch Processing Code Example Python

File Edit View Run Help Last edit was 10 hours ago Provide feedback

▶ Run all

■ Terminated

📅 Schedule

Share

(2) Spark Jobs

minT_df: pyspark.sql.dataframe.DataFrame = [station_number: double, area_code: string ... 9 more fields]

minT_values: pyspark.sql.dataframe.DataFrame = [area_code: string, min(value): double]

Maximum values for MinT parameter per station:

area_code	min(value)
VIC_PT088	3.9
WA_PT109	7.1
SA_PT006	3.2
SA_PT094	4.3
WA_PT154	16.2
NSW_PT158	7.5
NSW_PT085	6.3
VIC_PT202	6.2
WA_PT168	18.5
WA_PT006	7.4
WA_PT121	5.4
SA_PT009	9.0
NSW_PT035	3.7
NSW_PT125	2.6
VIC_PT093	3.5
NT_PT091	20.3
OLD_PT050	12.7

Command took 3.37 seconds -- by rohan_sudhir@samirfazi102gmail.onmicrosoft.com at 6/22/2023, 1:56:09 AM on Rohan Sudhir's Cluster

Cmd 4

1 configs = {"fs.azure.account.key.312datalake.blob.core.windows.net": "/39ZoOLJiJPfj5nvvZKb/oXVhY7uBhovSDAr8DRLuuOHngYieIyM480edR5Y3z3D9H8gnY03VL8K+AST1DH1JQ=="}

Batch Processing Code Example Python

File Edit View Run Help Last edit was 10 hours ago Provide feedback

▶ Run all

■ Terminated

📅 Schedule

Share

12

13 # Set plot title and labels

14 ax.set_title('Maximum Temperature Recorded per Area Code')

15 ax.set_xlabel('Area Code')

16 ax.set_ylabel('Max Temperature')

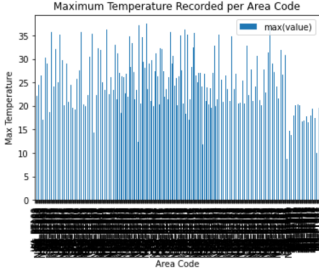
17

18 # Show the plot

19 plt.show()

(2) Spark Jobs

Maximum Temperature Recorded per Area Code



Command took 18.46 seconds -- by rohan_sudhir@samirfazi102gmail.onmicrosoft.com at 6/22/2023, 2:40:32 AM on Rohan Sudhir's Cluster

Limitations/Difficulties and Constraints

While setting up our pipeline and data lake, we faced difficulties with getting the setup done in azure to build the data lake. Multiple issues were brought up throughout the connection process. Some of these were setting up the initial service and then implementing a multi-user environment. However, we were able to overcome these issues through more research and team collaboration.

On discussion with the company representative, we were unable to receive the resources needed to build our solution. To overcome this, we explored for alternatives online for dummy data. To start off, we prepared a list of requirements needed to implement batch processing on the data, this would include the structured data. We then appointed these to include information such as station information, weather conditions and more.

Another issue faced was with the resources collected. We were unable to connect csv files to our azure server, due to which we had to reformat these resources to acceptable formats such as .xlsx to then run them on the database server.

Conclusion

Our project with FortyGuard involved designing a solution to handle batch processing of large-scale weather data, using the Azure architecture. Despite facing setup challenges, we were able to create a solution using Azure Data Factory, Data Lake, SQL server, and Databricks. This experience enhanced our understanding of big data management and its role in AI-driven solutions.

While we successfully implemented batch processing, there is scope for expanding this to handle real-time data and incorporate machine learning logic. The lessons learned from this project have equipped us better for future tasks, bringing the theory learned in class to life. Our efforts have resulted in a solution that is scalable, secure, and efficient, and can be extended to more complex operations in the future.

Team Contribution Worksheet

Student ID	Student Name	Contribution to the Team
7068293	Mahek Sajid	<ul style="list-style-type: none">● Presentation slides
6222560	Mohammed Uddin Khan Maarij	<ul style="list-style-type: none">● Report - Motivation, Limitations /Difficulties and Constraints● Azure databricks● Azure Data factory

		<ul style="list-style-type: none"> ● Presentation slides
7057167	Rida Fatimah Asif	<ul style="list-style-type: none"> ● Report - Main constructs/ Technical Requirements ● Azure Data factory ● Presentation slides
6866219	Rohan Sudhir	<ul style="list-style-type: none"> ● Report - Technical Requirements/ Solution implementation/ Solution Evaluation ● Azure databricks
7068943	Samir Fazil	<ul style="list-style-type: none"> ● Azure Data factory ● Azure data lake ● Azure databricks
5840454	Kianoush Rahravan	<ul style="list-style-type: none"> ● Report - Executive summary ● Report - Conclusion
7427359	Reyon Noronha	<ul style="list-style-type: none"> ● Report - Technical requirements ● Report - Main Constructs
7352396	Ibrahim Siddiqui	<ul style="list-style-type: none"> ● Generating mock data ● Azure SQL server
6997053	Nageen Irfan Minhas	<ul style="list-style-type: none"> ● Report - Limitations/Difficulties and Constraints ● Report - Main constructs