New Approach of Storing and Retrieving Large Data Volumes

Nedeljko Šikanjić[1], Zoran Ž. Avramović[2]

[1]*PhD student at Pan-European University Apeiron, Banja Luka, Bosnia and Herzegovina*

[2]*professor at Pan-European University Apeiron, Banja Luka, Bosnia and Herzegovina*

A General Survay

DOI: **10.7251/JIT1902089S** UDC: **004.6:658.4]:519.8**

Abstract: In today's world of advanced informational technologies, society is facing a huge amount of data that is just getting impossible to store, process and analyze. In these big data volumes, some of the important information is being lost, that could help us improve the quality of personal and business life. This paper focus is on finding the best possible way of approaching this issue to find a feasible solution in increasing the efficiency and quality of data.

Keywords: Data Warehouse, Data Lake, Lambda architecture.

Introduction

When it comes to every day of people's lives, including the social and business perspective of it, it generates various types of data every second. The internet, different tracking and transaction logs, various documents, emails, numerous business applications such as ERP or CRM, IoT systems and devices, they all produce a high volume of data. In this data, is hidden right information for the right process, which might be used depending on the need of the system, organization or person. In this paper analysis is made on the existing solutions, their benefits, and their faults or disadvantages, to find a better approach or solution for coping with a large volume of data.

Data Warehouse

When people think about structured data, the first thing that comes in mind is data warehouses. This approach with data warehouses is since the 1990s emerged as a need to have a solution for storing a large volume of data. William H. Inmon has created the term data warehouse and contributed to creating and developing data warehouse architecture [9].

The data warehouse is well known for its structured data and schema. The schema represents the way of how data will be grouped and organized, including a well-structured hierarchy. In this way, we have the benefit and disadvantage of knowing before time what data and in what format the data will be stored. These data warehouses are optimized for reading in terms of query performance, and therefore it is a performance-based big advantage in using these data warehouse systems. When we compare with transactional database models (OLTP), in data warehouses it is being used as an analytical model approach (OLAP) where the reading of data is a key factor.

This data warehouse for storing large volumes of data approach was good enough but just for a time being as information society developed, so did data also. We are faced with different types of data coming in like data from IoT (internet of things), social media data that was well unstructured, so this has

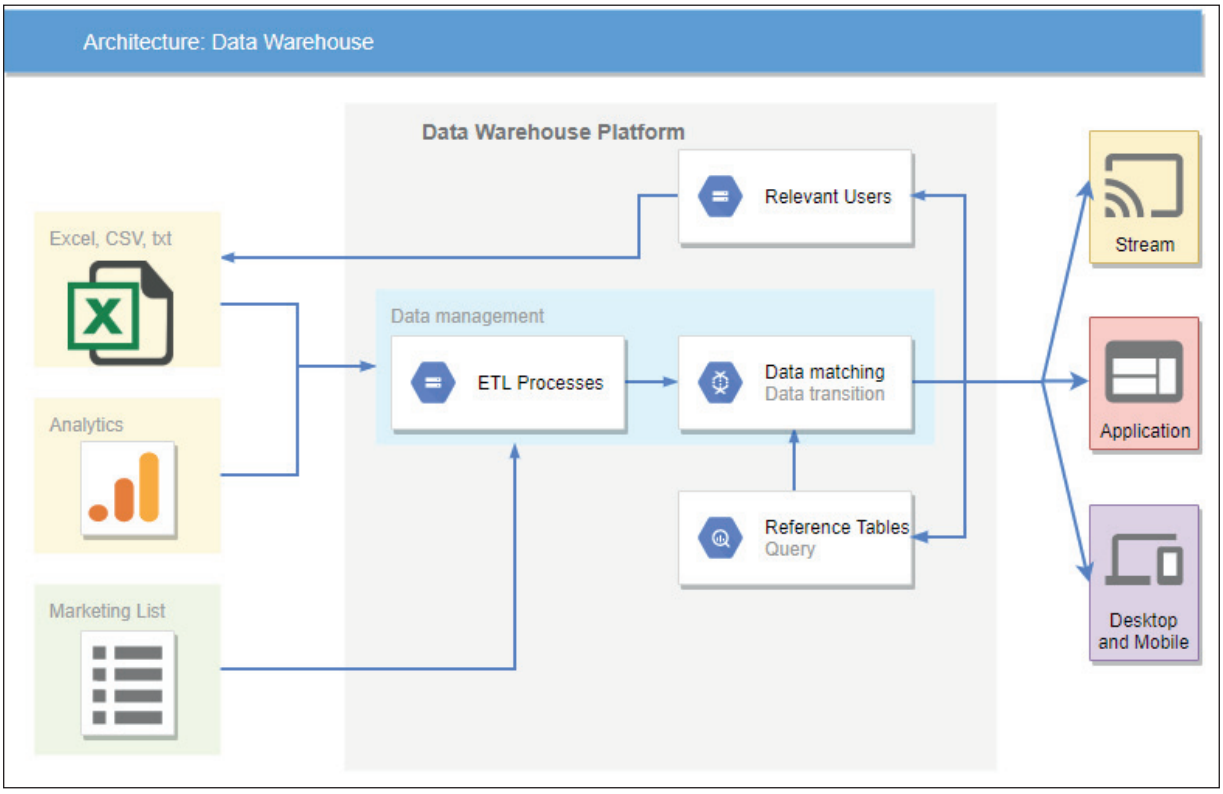

Figure 1. *Data warehouse architecture*

imposed huge pressure on existing data warehouse in order to cope with this kind of new data.

The benefit of the data warehouse is that has proven itself on the market for a long time, that human resources are well-skilled, that is has matured over time in the term of stability and reliability. Performance is another key aspect of the data warehouse as it is based on good structure and great query engines that are fully optimized for reading and are supporting various incremental changes of the data. Another great characteristic is usability as users may not be familiar with how to get information from source data, but with the analytical approach of the data warehouse, users can by transforming, filtering or slicing the data to find the information they need. In this way, users are getting a single source of data, instead of matching various sources of data, trying to find the information they are looking for. With the coming of cloud services, data warehouse systems are very well adapted to new technology in this way, where we have the flexibility of having on-premise or in the cloud the data while keeping this architecture.

Speaking of some of the downsides of data warehouses, we must mention storage cost as this kind of data model or data architecture is requiring lots of storage resources. As we already have explained the benefit of reading time, that does come with a certain price in terms of time. Time is required by preparing the processes and components that are needed in order to pre-structure the source data that is coming into the data warehouse. As we know what structure of data we are looking for, we might lose some data that could be useful in the long run, as we remove this data in ETL (extract, transform, load) processes. Another disadvantage of the data warehouse as it is not designed for the large volume of various data or better known as big data that includes the internet of things and social media for example.

DATA LAKE

The other competitor in storing large data volumes is data lake. The first term of data lake was introduced by James Dixon, where he has compared data lake as a large whole of water stored in a natural state [2]. This concept was created as it was noticed that only part of data has been visible and processed, as data has attributes that are predefined and after data is aggregated, subset levels of data are not seen

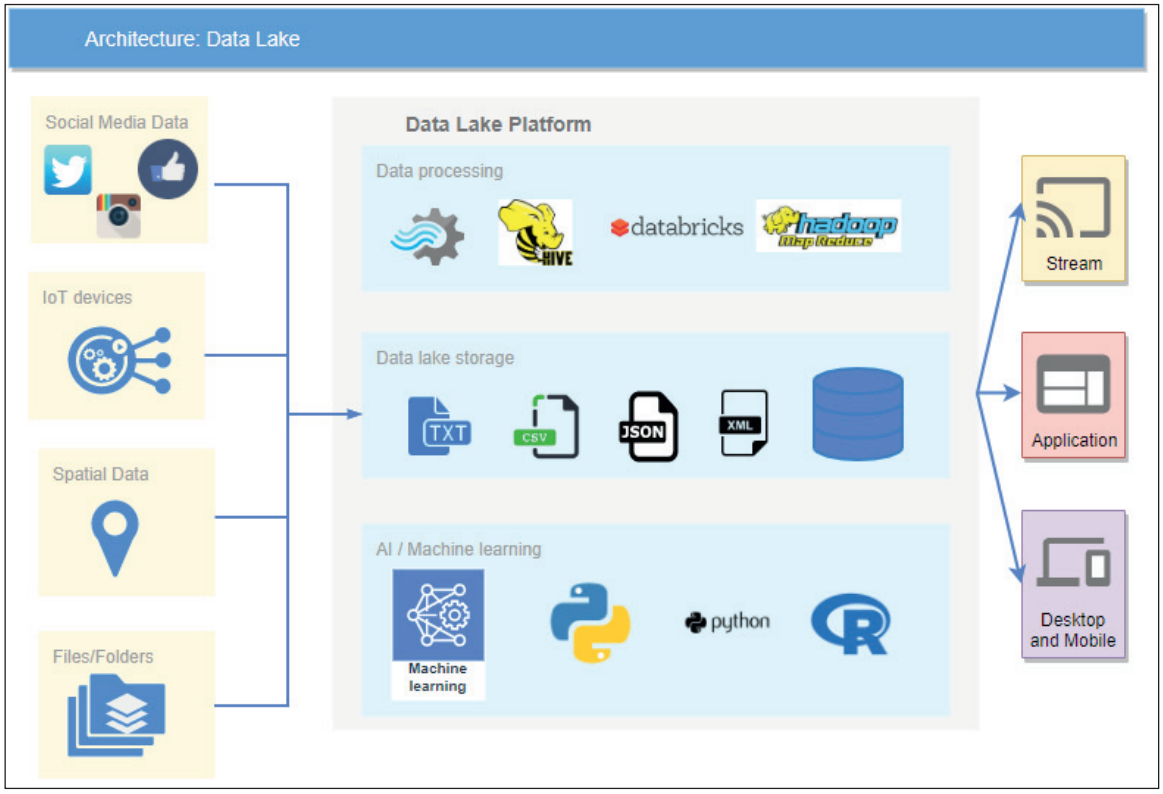

Figure 2. *Data lake architecture*

more. In order to keep all the data, we have or generate, with data lake we will keep them in their original form, and then we will introduce the processing of data we require in that specific moment.

When it comes to the data lake, the big advantage is that we can store all kinds of data in it. For example, we can store structured data, semi-structured of unstructured data. Here we don't have a predefined schema, so it means that we can put or save data in its raw format, without losing any time on preparing the data for storing. Because of this, we don't have a data model in the term of transactional or analytical only, but it is more organized in the way of storing data based on different types of data that we are trying to save in the data lake.

If we would like to make it sounds simplified, we could say that it would be enough to store our data into date lake and at end of the process, we could have some sort of reporting or analysis services, that would be responsible for a representing the data at our end users or clients.

However, there is, of course, more refinements that would need to be implemented, such as security and data governance, to have a sustainable and reliable solution for managing our data.

Some of the disadvantages of data lakes that we need to mention are human resources with skills that are needed to process this data within the data lake environment, then there is an issue with knowing how this data flow will fit in within the organization, that is implementing this approach or has already a data warehouse processes established. Also, as a benefit of saving the data results in low cost in storage, the downside is that will increase the performance cost of implementing the complex queries on the data from the data lake.

Proposed Solution

In most cases, it is not a question if it is an only data warehouse or a data lake approach, but it is determined on the need for the project or organizational infrastructure.

The best approach to have a full potential of both data implementation solution is an integration of both systems in a hybrid solution. With this approach, we can leverage the full capabilities of storing large sets of data while preserving the functionality of data processing, data quality and securing the data.

Here we will analyze the approach of ETL (extract-transform-load) and ELT (extract-load transform).

ETL is a process where is most common when we know what structure we have forehand, so in this way, we can prepare data for the questions we already know to provide the answer.

For the ELT process, it works perfectly in the environment where we want to take advantage of data, which we would like to find answers to the questions that might come up in the future.

For some approaches, we can use a combination of machine learning and artificial network algorithms [4] to automate processes.

LAMBDA APPROACH

Lambda architecture [3] is data processing architecture created on the need of speeding up the processing of data regarding large data volumes or big data implementation. When using the data processing algorithm, we will put data coming in batches. This means that this data will be grouped so we can then try to set some operations based on these batches of data, to get information from this data. Once we get data in batches, we will try to query the data. However, we have some batches of data finished and ready for processing but data that are still coming in are in the middle of preparing batches. This means that we are missing this data in real-time. This is the moment when we implement streaming. Streaming data means that we will take the same data that are coming into batches and make it available for the querying. After the corresponding batch is finished with the processing of data, we will clear the data list that is in the streaming process, to prevent duplication of the data.

Table 1. *Simplified Lambda process flow*

Input data	Batch	Output data
	Speed	

To implement the best practice approach, we will try using lambda processing data flow.

In our sample test case, we will have sample data from the AirVisual meteorological web site. Here we will use data that we will stream into our data lake and then we will process it to see the behavior of the proposed system. For the tools used in this test case, we will use a Microsoft Azure cloud platform, as this is one of the fastest growing online cloud platforms that supports various big data systems and big data platforms.

So, once we get data inside of our system, we will store this data into the data lake store. Data lake store is based on HDFS (Hadoop Distributed File System). Data will be distributed on more nodes, which means that we will have more copies of our data and access to it will be faster because this approach supports the parallel reading of data.

For the batch process, where we process our data, we will use a new approach with U-SQL [8] procedural language. This language is a new way of supporting unstructured data. It is a mixed technology approach of supporting C# programming language and a standard SQL language. Based on our needs we can transform the data in structured data, or we can output it also as unstructured data. This approach gives great flexibility in serving data to the end-users. The component that we will use is called data lake analytics.

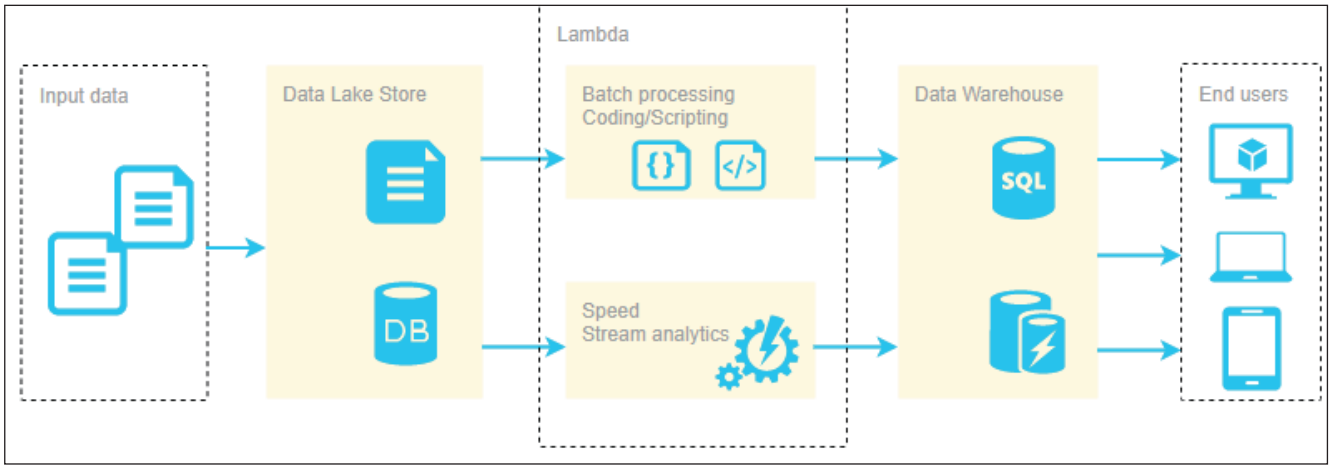

Figure 3. *Diagram of Lambda architecture with data lake and data warehouse*

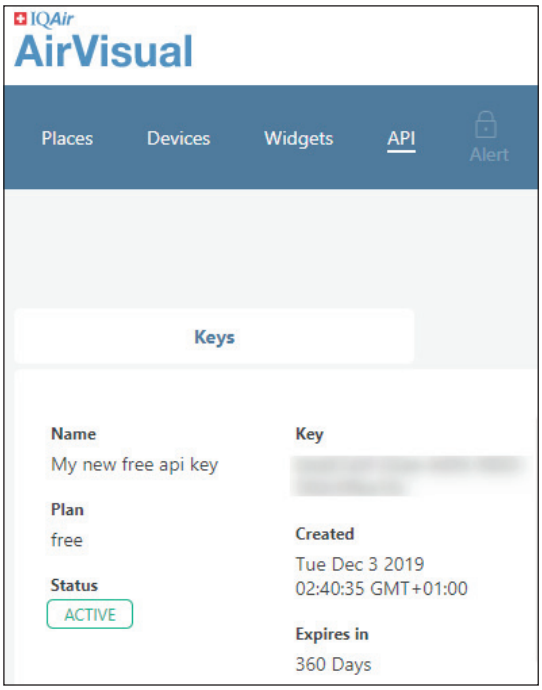

Figure 4. *AirVisual credential page*

For the speed layer, we will use the streaming analytics component, as this is excellent in terms of supporting standard SQL language, supports vertical partitioning and can write to more than one output at the same time.

Before we send data to the users, we will use the data warehouse component, as this is where our end data will be processed and stored. This represents a combination of two different systems we try to combine, to have the best result from both data systems [5].

To implement this solution, we will first set up an event hub, which will act as an IoT input point, which will receive information from the AirVisual website.

First, we test the API of AirVisual, so we are sure to set the right call from the event hub using an API testing tool.

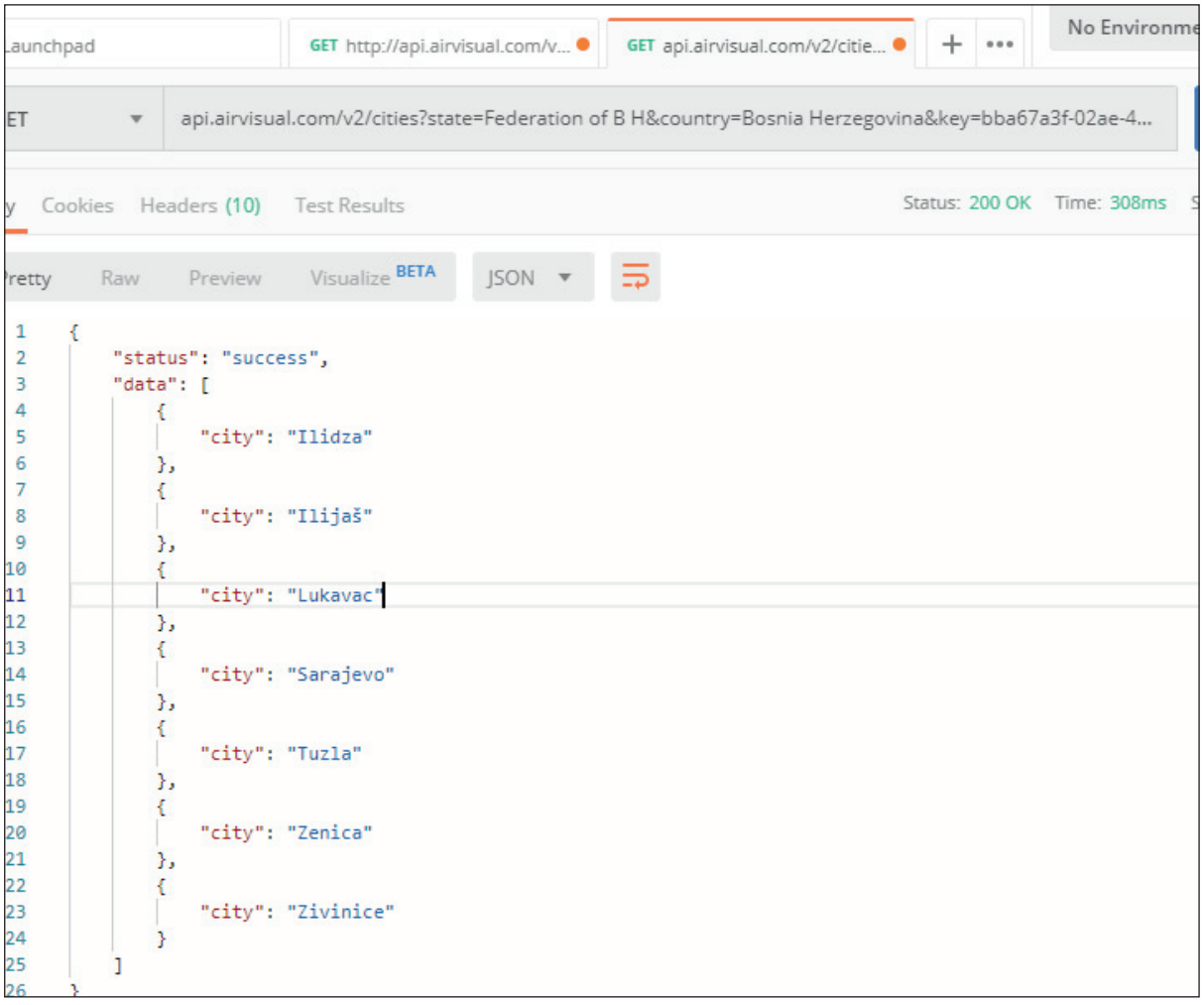

Figure 5. *Data structure of response from Web API*

After having an input data source, we will set up a streaming job, where we set a source for our streaming job.

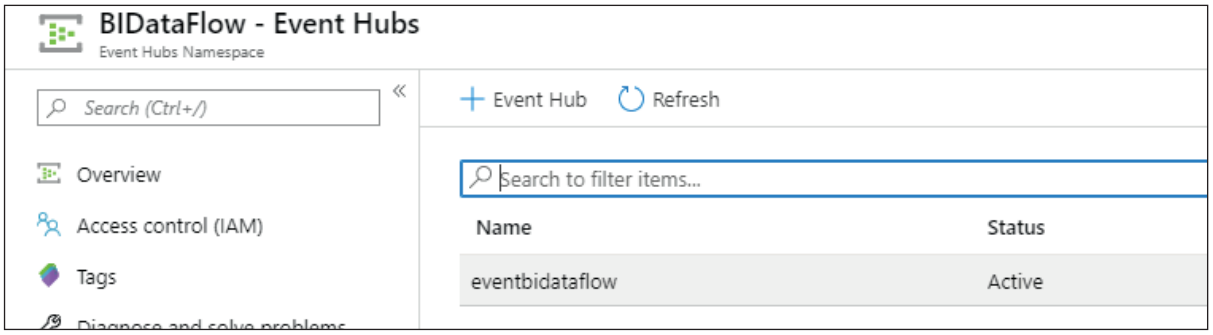

Figure 6. *Event Hub for data input*

For each of our data sources, we will set the output of this component to a data lake store.

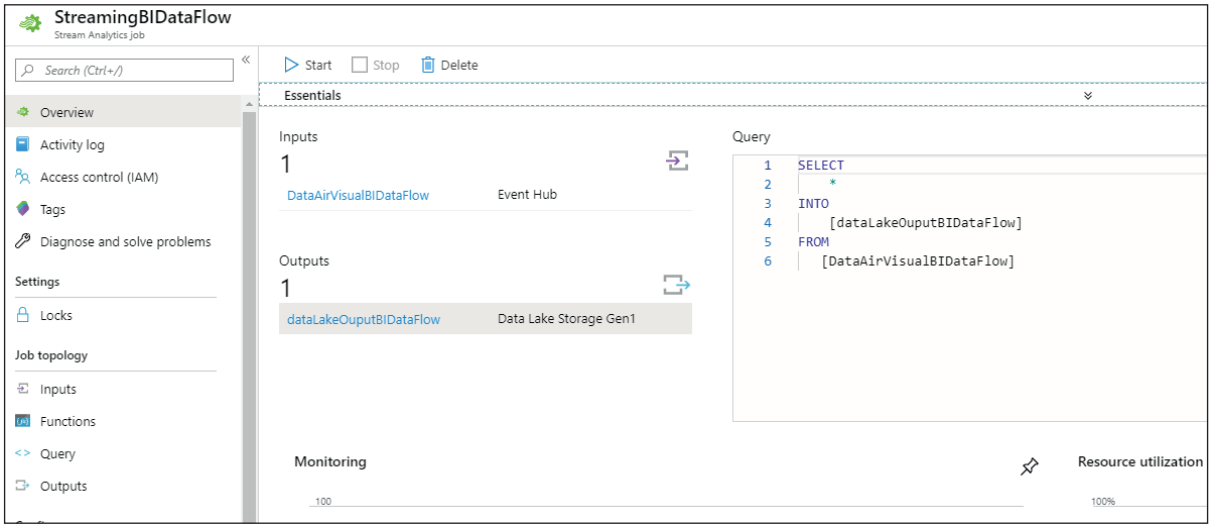

Figure 7. *Streaming analytics for speed layer*

For the batch processing, we will use a data lake analytics, where we will be doing the processing of our data. At the end of each processing of data, we will be having an output flat files, in this case in CSV (Comma-separated values) format.

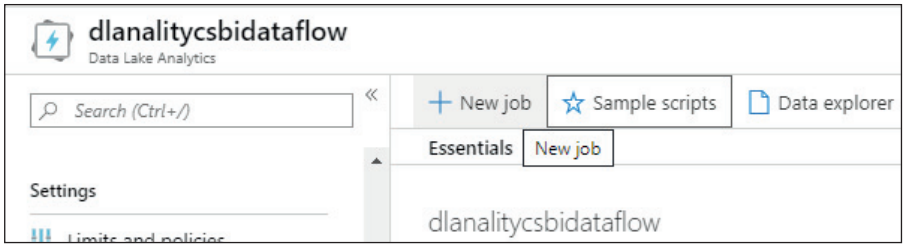

Figure 8. *Data lake analytics for batch processing*

Just go a little bit back to our first step of loading the data, let's examine the data we are interested in. This important to show how we integrate this into the existing lambda architecture we have created. The data is coming in JSON (JavaScript Object Notation) format.

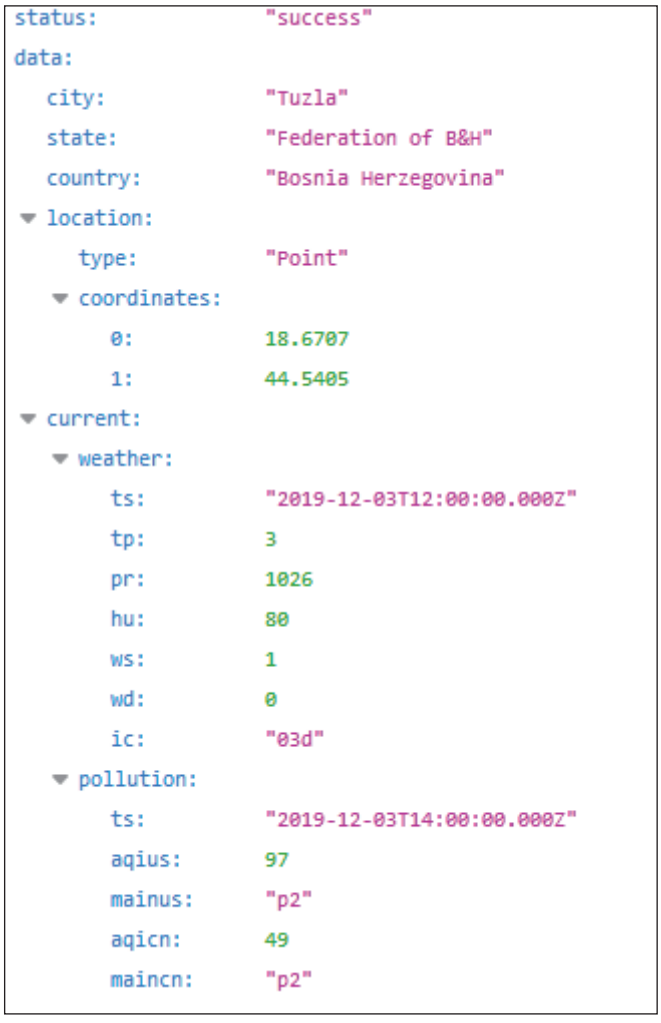

Figure 9. *JSON structure of raw data*

So, as we can see on the JSON result above, we need to have a value for pollution "aqius" - AQI value based on US EPA standard [6]

Based on this data, we have structured our U-SQL query as it follows:

```
DECLARE @InputDirectory string ="/SourceData/{FileName}.csv"; //source files
DECLARE @OuputDirectory string = "/OutData/Polution.csv";     //output file

@RawData=
    EXTRACT City string,
                Aqi int,
                TS DateTime
FROM @InputDirectory
    USING Extractors.Csv(skipFirstRows:1);

@OutputData =
    SELECT City,
           SUM(Aqi) as TotalAQI
    FROM @RawData
    GROUP BY City;
OUTPUT @OutputData TO @OuputDirectory USING Outputters.Csv();
```

Figure 10. *U-SQL for batch processing*

When it comes to the data warehouse, we will be using a tool called PolyBase [1]. With the PolyBase approach, we will set the source of our queries using a table that is dynamically connected with underlying flat files. Then we can use a query that we can use to merge the results from these outputs of batch processing and speed processing, following a lambda architecture design process flow.

As we are doing implementation on the Azure platform, we are using an Azure Synapse Analytics (formerly known as SQL Data Warehouse). When working with the Cloud applications, then security is one of the key prerequisites. We will be setting an OAuth2 authorization framework [7] as this is the best security option when working with cloud applications and web applications in general.

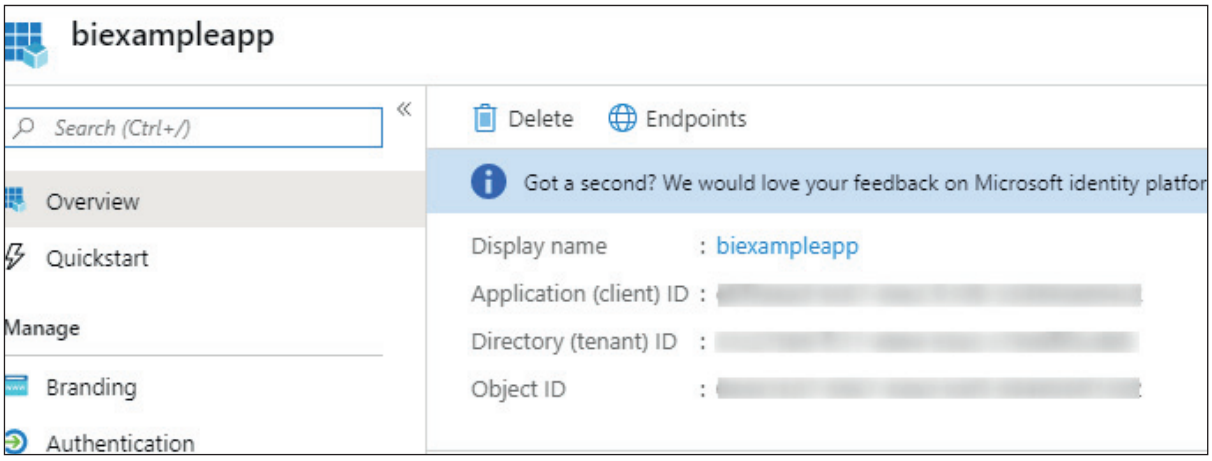

Figure 11. *OAuth credential generation*

Then it is quite straightforward to implement these credentials within our data warehouse.

```
create database scoped credential azureDataLakeCredential
with
identity='                                    2@https://logi
secret='f                                     T';


create external data source DataLake
with (
type=hadoop,
location='adl://datalakebidataflow.azuredatalakestore.net',
credential=azureDataLakeCredential
)
```

Figure 12. *Usage of generated OAuth credentials*

What is important to know, that we can separate the level of access to individual objects into the data lake. We can allow read to some objects as CSV files in our case and we can even allow higher-level permissions, depending on the scenario we would like to implement. This resembles in granular functionality where we can have great control over the security in general.

Here we see the implementation of the PolyBase data lake source file query, where we propagate the location on to the Hadoop system. With this approach, we can use a data warehouse as a central point for the serving layer in our lambda architecture.

Based on this we will get data to the end-users. We can use various tools to analyze this data and excel

```
create external table dbo.SpeedProcess
(
CityName nvarchar(100),
AQI int,
MAINS int,
AQICN int,
MAINUS nvarchar(2),
MAINCN nvarchar(2),
dateTS nvarchar(100)
)
with
(
location='SourceData/AllAirPolutionInputData_20191125_151606.csv',
data_source=DataLake,
file_format=csvfile,
reject_type=Value,
reject_value=0
)

SELECT
cityName,
case when AQI<=50 then 'Good'
when AQI between 51 and 100 then 'Moderate'
when AQI between 101 and 150 then 'Unhealthy for Sensitive Groups'
when AQI between 151 and 200 then 'Unhealthy'
when AQI between 201 and 300 then 'Very Unhealthy'
end as DegreeOfPoluttion,
AQI,
datets as DateCollected
FROM
(
select cityName, AQI, dateTS from dbo.BatchProcess
union
select cityName, max(AQI) as AQI, dateTS from dbo.SpeedProcess
GROUP BY cityName,dateTs
) as t
```

Figure 13. *Server layer output query with data lake and data warehouse*

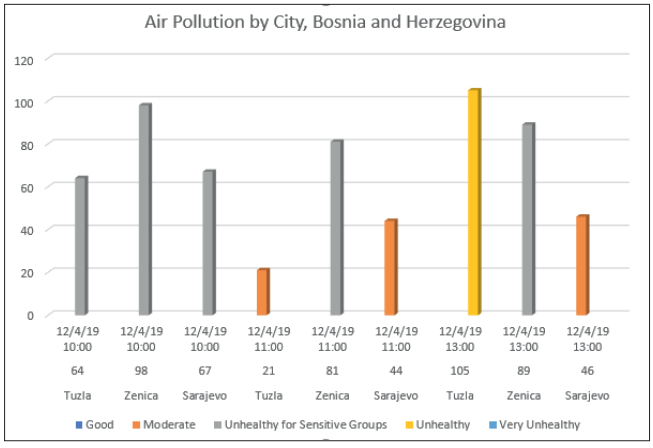

Figure 14. *Reporting analytics for end users*

with data connectors presents a powerful solution as it is easy to use and rich with data analyzing features.

As it can be seen from the graph, here is shown data that is coming simultaneously and data that is being processed in batches. This way, we don't miss out on the data while waiting for being processed and served to the client or user service.

Conclusion

This paper has shown how to implement the approach of lambda architecture into the latest technologies while combining the best features from big data and standard database models. Presented re-

search explains how advanced it is possible to go in the term of getting the most out of the data processing while keeping data integrity and minimizing the time of response, from the input data to the serving the data to the end-users. Also, as people as a society in general, are moving into cloud and internet applications in every segment of everyday lives, this paper has demonstrated how to implement big data solutions in terms of data consistency while keeping the focus on the security as one of the important factors as well.

REFERENCES:

[1] Benjamin Weissman, "PolyBase in SQL Server 2019 – The End of ETL?", https://www.red-gate.com/simple-talk/sql/data-platform/polybase-in-sql-server-2019-the-end-of-etl/ (accessed on 23.10.2019)
[2] James Dixon, "Pentaho, Hadoop, and Data Lakes", https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/ (accessed on 28.09.2019)
[3] Nathan Marz, "Big Data: Principles and best practices of scalable realtime data systems", Manning publication Co ISBN-13: 978-1617290343
[4] Nedeljko Šikanjić, Zoran Ž. Avramović, Esad F. Jakupović, "Implementation of the Neural Network Algorithm in Advanced Databases", JITA – Journal of Information Technology and Applications, PanEuropien University APEIRON, Banja Luka, Republika Srpska, Bosna i Hercegovina, JITA 8(2018) 2:54-63, (UDC: 004.738.5:551.588:551.506)
[5] Simon Whiteley, "A Guide to Azure SQL DataWarehouse", https://adatis.co.uk/a-guide-to-azure-sql-datawarehouse/ (accessed on 18.10.2019)
[6] The AirVisual API description, https://api-docs.airvisual.com/?version=latest#detailed-response-example (accessed on 23.10.2019)
[7] The OAuth 2.0 Authorization Framework, https://tools.ietf.org/html/rfc6749 (accessed on 25.10.2019)
[8] U-SQL Language Reference, "A Guide to Azure SQL DataWarehouse", https://docs.microsoft.com/en-us/u-sql/ (accessed on 17.10.2019)
[9] William H. Inmon, "Building the Data Warehouse", Wiley Computer Publishing ISBN: 0-471-08130-2

Submitted: October 16, 2019
Accepted: November 13, 2019

ABOUT THE AUTHORS

Nedeljko Šikanjić holds a Magister degree in Informatics and Computer Science and has worked for more than 15 years as a Software and Database Architect/Engineer. His main fields of studies are in the area of advanced Databases and Software Architectures. He has been a holder of an active Microsoft Certified Trainer Certificate since 2012 and has been teaching courses on various topics in Information Technologies. Doctoral studies of the third degree enrolled in the academic 2017/2018.

Zoran Ž. Avramović was born in Serbia (Yugoslavia) on September 10th, 1953. He graduated from the Faculty of Electrical Engineering, University of Belgrade. At this Faculty he received a Master's degree, and then a PhD in technical sciences. He is:

- Academician of the Russian Academy of Transport (RTA, St. Petersburg, Russia, since 1995),
- Academician of the Russian Academy of Natural Sciences (RANS, Moscow, Russia, since 2001),
- Academician of the Yugoslav Academy of Engineering (YAE, Belgrade, Serbia, since 2004) (today: Engineering Academy of Serbia, EAS)
- Academician of the Academy of Electrotechnical Sciences of the Russian Federation (AES of the Russian Federation, Moscow, Russia, since 2007)

Scientific Secretary of the Electrical Engineering Department of the Engineering Academy of Serbia.

FOR CITATION

Šikanjić N., Avramović Ž. Z., New Approach of Storing and Retrieving Large Data Volumes, *JITA – Journal of Information Technology and Applications Banja Luka*, PanEuropien University APEIRON, Banja Luka, Republika Srpska, Bosna i Hercegovina, JITA 9(2019) 2:89-98, (UDC: 004.6:658.4]:519.8), (DOI: 10.7251/JIT1902089S), Volume 9, Number 2, Banja Luka, december 2019 (49-128), ISSN 2232-9625 (print), ISSN 2233-0194 (online), UDC 004