1. What is the process of loading a data set from a external source?

Ans: If you use no data from STATMON as the source for your IBM Z Performance and Capacity Analytics resource information, you can import data from another external source into IBM Z Performance and Capacity Analytics. The external source serves as the primary repository for network resource information, so you must not change the resource information after you update the NW_RESOURCE table with the information. Any changes you make to the resource information using the network administration dialog will not be present in the external source, and therefore will not reflect the actual state of the network. The data must be in the format presented in "External data file format" for the dialog to be able to use it. Load data from the external source when you are first defining your network to the Network Performance Feature and whenever the network changes. The following figure illustrates the process of loading data from an external source in the IBM Z Performance and Capacity Analytics database.

1. Loading data from an external source. Loading data from an external source If you use external data to create your database, follow these steps:set the defaults for network resources in the network administration dialog. You must specify the name of a data set that you have already allocated in the Output data set name field. The dialog requires that this value be present so you can update the database. Because the dialog does not permit you to enter only the value for that field, you must enter values for all fields, even though the dialog does not use the information.Load the external data into the dialog work table.Verify the resource information.Process the database update statements to load the data in the work table into the IBM Z Performance and Capacity Analytics database.

2. How can we use pandas to read JSON files?

Ans:. Introduction:

Pandas is one of the most commonly used Python libraries for data handling and visualization. The Pandas library provides classes and functionalities that can be used to efficiently read, manipulate and visualize data, stored in a variety of file formats.

In this article, we'll be reading and writing JSON files using Python and Pandas.

JavaScript Object Notation (JSON) is a data format that stores data in a human-readable form. While it can be technically be used for storage, JSON files are primarily used for serialization and information exchange between a client and server.

Although it was derived from JavaScript, it's platform-agnostic and is a widely-spread and used format – most prevalently in REST APIs.

Creating a JSON File:

To create JSON files via Python, data has to be stored in a certain way. There are multiple ways of storing this data using Python. Some of the methods have been discussed in this article.

We'll first create a file using core Python and then read and write to it via Pandas.

Creating JSON Data via a Nested Dictionaries

In Python, to create JSON data, you can use nested dictionaries. Each item inside the outer dictionary corresponds to a column in the JSON file.

The key of each item is the column header and the value is another dictionary consisting of rows in that particular column. Let's create a dictionary that can be used to create a JSON file that stores a record of fictional patients

3. Describe the significane of DASK.?

Ans: Python's role in Data Science Python has grown to become the dominant language both in data analytics and general programming: Graph showing the growth of major programming languages based on Stack Overflow's question views in World Bank high-income countries. A line graph with time from 2012 to 2018 on the x-axis and percent of overall question views each month on the y-axis. Python's question views increase from about 4% to about 11% from 2012 to 2018, reaching the popularity of Java and JavaScript. This is fueled both by computational libraries like Numpy, Pandas, and Scikit-Learn and by a wealth of libraries for visualization, interactive notebooks, collaboration, and so forth. Graph showing the growth of major python packages based on Stack Overflow's question views in World Bank high-income countries. A line graph with time on the x-axis from 2012 to 2018 and percent of overall question views each month on the y-axis. Pandas question views increased to about 0.9% in 2018, exceeding Django and NumPy. However, these packages were not designed to scale beyond a single machine. Dask was developed to scale these packages and the surrounding ecosystem. It works with the existing Python ecosystem to scale it to multi-core machines and distributed clusters. Image credit to Stack Overflow blogposts #1 and #2. Dask has a Familiar API Analysts often use tools like Pandas, Scikit-Learn, Numpy, and the rest of the Python ecosystem to analyze data on their personal computer. They like these tools because they are efficient, intuitive, and widely trusted. However, when they choose to apply their analyses to larger datasets, they find that these tools were not designed to scale beyond a single machine.

Python has grown to become the dominant language both in data analytics and general programming:

Graph showing the growth of major python packages based on Stack Overflow's question views in World Bank high-income countries. A line graph with time on x-axis from 2012 to 2018 and percent of overall question views each month on the y-axis. Pandas question views increased to about 0.9% in 2018, exceeding Django and NumPy.

However, these packages were not designed to scale beyond a single machine. Dask was developed to scale these packages and the surrounding ecosystem. It works with the existing Python ecosystem to scale it to multi-core machines and distributed clusters.

Dask has a Familiar API

Analysts often use tools like Pandas, Scikit-Learn, Numpy, and the rest of the Python ecosystem to analyze data on their personal computer. They like these tools because they are efficient, intuitive, and widely trusted. However, when they choose to apply their analyses to larger datasets, they find that these tools were not designed to scale beyond a single machine. And so, the analyst rewrites

their computation using a more scalable tool, often in another language altogether. This rewrite process slows down discovery and causes frustration.

4. Describe the functions of DASK.?

Ans:. There is clearly parallelism in this problem (many of the inc, double, and add functions can evaluate independently), but it's not clear how to convert this to a big array or big DataFrame computation.

As written, this code runs sequentially in a single thread. However, we see that a lot of this could be executed in parallel.

The Dask delayed function decorates your functions so that they operate lazily. Rather than executing your function immediately, it will defer execution, placing the function and its arguments into a task graph.

Delayed([obj, name, pure, nout, traverse])

Wraps a function or object to produce a Delayed.

We slightly modify our code by wrapping functions in delayed. This delays the execution of the function and generates a Dask graph instead:

```
Import dask

Output = []

For x in data:

   A = dask.delayed(inc)(x)

   B = dask.delayed(double)(x)

   C = dask.delayed(add)(a, b)

   Output.append©

Total = dask.delayed(sum)(output)
```

We used the dask.delayed function to wrap the function calls that we want to turn into tasks. None of the inc, double, add, or sum calls have happened yet. Instead, the object total is a Delayed result that contains a task graph of the entire computation. Looking at the graph we see clear opportunities for parallel execution. The Dask schedulers will exploit this parallelism, generally improving performance (although not in this example, because these functions are already very small and fast.)

```
Total.visualize()  # see image to the right
```

A task graph with many nodes for "inc" and "double" that combine with "add" nodes. The output of the "add" nodes finally aggregate with a "sum" node.

We can now compute this lazy result to execute the graph in parallel: Total.compute()

45 Decorator

It is also common to see the delayed function used as a decorator. Here is a reproduction of our original.

5. Describe cassandra's features?

Ans:Features of Cassandra Apache Cassandra is an open source, user-available, distributed, NoSQL DBMS which is designed to handle large amounts of data across many servers. It provides zero point of failure. Cassandra offers massive support for clusters spanning multiple datacentres. There are some massive features of Cassandra. Here are some of the features described below: Distributed: Each node in the cluster has has same role. There's no question of failure & the data set is distributed across the cluster but one issue is there that is the master isn't present in each node to support request for service. Supports replication & Multi data center replication: replication factor comes with best configurations in cassandra. Cassandra is designed to have a distributed system, for the deployment of large number of nodes for across multiple data centers and other key features too.scalability: It is designed to r/w throughput, Increase gradually as new machines are added without interrupting other applications.Fault-tolerance: Data is automatically stored & replicated for fault-tolerance. If a node Fails, then it is replaced within no time.MapReduce Support: it supports Hadoop integration with MapReduce support.Apache Hive & Apache Pig is also supported.Query Language: Features of Cassandra

Apache Cassandra is an open source, user-available, distributed, NoSQL DBMS which is designed to handle large amounts of data across many servers. It provides zero point of failure. Cassandra offers massive support for clusters spanning multiple datacentres.

There are some massive features of Cassandra. Here are some of the features described below:

Distributed:

Each node in the cluster has has same role. There's no question of failure & the data set is distributed across the cluster but one issue is there that is the master isn't present in each node to support request for service.

Supports replication & Multi data center replication:

Replication factor comes with best configurations in cassandra. Cassandra is designed to have a distributed system, for the deployment of large number of nodes for across multiple data centers and other key features too.

Scalability:

It is designed to r/w throughput, Increase gradually as new machines are added without interrupting other applications.

Fault-tolerance:

Data is automatically stored & replicated for fault-tolerance. If a node Fails, then it is replaced within no time.

MapReduce Support:

It supports Hadoop integration with MapReduce support.Apache Hive & Apache Pig is also supported.

Query Language:

Cassandra has introduced the CQL (Cassandra Query Language). It's a simple interface for accessing the has introduced the CQL (Cassandra Query Language). Its a simple interface for accessing the Cassandra