Python

08 August 2024 21:25

Inheritance
Abstraction
Access Modifiers
File handling
List comprehension

Decorators
Classmethod
Static method
Frozen set

```
Print("This is print statement",value)
Print("Addition of value1: %d and value2: %d = %d  " %(value1,value2,value3))
Print("Addition of value1: {} and value2: {} = {}  " .format(value1,value2,value3))
Print(f"Addition of value1: value1 and value2: value2 = value3  ")
```

Sys - pythons std library and offers functions and variables that interact with directly interpretor and os of python.

- Sys.argv - list of command line argument that is passed to the script
- Sys.exit - exit the program with a specified exit status
- Sys.path - consists of list of string which specify the search path for the modules
- Sys.stdin - file like object that represents standard input stream and is used to read input from user or from input path, used when u need to handle input from command line
- Sys.stdin.readLine - read a single line

MongoDB

19 August 2024 10:46

To store data in better format especially for unstructured data.

To show all databases database
- show databases

To create new database or to use existing database
- use database

To create new collection
- db.createCollection("collection_name")

To insert entries into the Collection
- db.collection_name.insertOne({ "product_id":1,"name": "Laptop"})
- db.collection_name.insertMany([{ "product_id":1,"name": "Laptop"},{"product_id":1,"name": "Laptop"}])

To show all documents (all entries)
- db.collection_name.find()

To show a particular document
- db.collection_name.find({"product":1})

To show document \ records having price greater than 1000
- db.collection_name.find({price: {$gt: 1000}})
- Similarly we can use : lt , gte, lte instead of gt

To update one document
- db.collection_name.updateOne({"product_id":1},{$set:{"discount": .10}})

To update multiplre records
- db.collection_name.updateMany({"category":"electronics"},{$set:{"discount": .15}}) - set is used to create new data field
- All entries having category as electronics will be updated

To delete a document
- db.collection_name.deleteOne({"product_id":1})
- db.collection_name.deleteMany({"category":"electronics"})
- db.collection_name.deleteMany({"price": {$lte: 50000}})

To count no of documents in the collection
- db.collection_name.countDocuments()

Drop collection using collection name
- db.collection_name.drop()

Drop database
- db.dropDatabase()

- db.getCollectionInfo()
- db.getCollectionNames()

To display certain columns - use projection

- Db.collection_name.find(query, projection)

Scala

21 August 2024 16:37

- Object Oriented and Functional Programming Language
- Scala - scalable language

Features:

1. Object oriented
2. Functional
3. Extensible
4. Statistically typed
5. Runs on JVM
6. Processes Concurrently

Types of collection

Array

- Mutable
- Homogenous
- Fixed length
- Access individual values of array using indexing

List

- Immutable
- Homogenous
- Variable length

Functional Programming

- Programming paradigm where computation is treated as evaluation of **mathematical function**
- It avoids changing in state or mutable data
- Functions (**First class functions**) - functions can be passed as an argument to function, return as value from other functions and can be assigned to variables
- Scalable (Resources)

Data Immutability

val (Immutability comes from) , var(mutability comes from)

- **Pure Functional Language** - functions should be pure

  Same input --> Function --> Same output

- **Advantages**

- Bug free
- efficient parallel processing
- efficiency
- nested processing
- lazy evaluation - lazy variable , make use of resources efficiently

Big Data

26 August 2024 09:52

Big data is the data that has :-

- Volume
- Velocity
- Variety
- Veracity
- Value

4 Vs of Big Data

- Volume
- Velocity
  - Pace of the data
- Variety
  - Structured
  - Semi Structured
  - Quasi Structured
  - Unstructured data
- Veracity
  - Inherent discrepancies in data
    - Inaccurate results
  - Truthfulness of data

Hadoop

1. Banking
   - Transaction, credit cards, credit score
2. Healthcare
   - Early medical diagnosis, personalised diagnosis
3. Energy
   - EV ,Hybrid
4. Technology
   - Advancements
5. Consumer
   - Forecasting product demands, managing logistics (Instamart, Bigbasket)
6. Manufacturing

Challenges of traditional decision making

1. Long time to arrive at a decision
2. Human intervention required - Failures may occur.
3. Lack systematic linkage
4. Provides limited scope of data analytics ie, provides only top view
5. Obstructs company's ability to make fully informed decisions

Solution - Big Data Analytics

1. Decision making based on data analytics
2. Analysing data from various sources
3. Streamlined decision making from top to bottom
4. Analysing unstructured data
5. Faster decision making, competitive advantage

Big Data reveals:
Patterns - clusters
Trends - line plot (Forecasting) -> Time Series Forecasting - LSTM, ARIMA
Associations - Data mining - Apriori

Types of data

- **Structured**
  - Data having defined data model
  - Tabular format data
    e.g. Database
- **Semi - Structured**

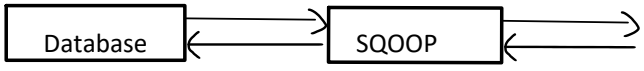
    - Textual data with apparent pattern
    - Json, XML (Extended Markup Language), CSV, TSV(tab seperated), LSV (Delimeter)
- **Quasi Structured**
    - Textual data with erratic formats that can be formatted with effort and software tools
    - E.g. Clickstream data
- **Unstructured data**
    - No inherent structure
    - E.g. Text documents, pdf, images and videos

Netflix Case Study

Big Data Analytics Pipeline

- Data Sources
- Data Ingestion Layer
    - Sqoop
    - Kafka
    - Flume
- Data Collection layer
- Data Storage Layer
    - Hadoop
    - AWS
- Data Processing Layer
    - Batch processing
    - Real time processing
    - Hybrid processing
- Data Query Layer
    - Spark
- Analytics engine
    - Spark - not used for storage
- Data Visualisation
    - Data bricks

Sqoop : Import and export data from source (Structured source) to destination (HDFS - Hadoop distributed file system)



Flume : Flume scales used for migrating data from URL (Live Streaming Source) and collect it to store in HDFS (Storage layer). This is unidirectional

Flume
Source----------> HDFS
At regular intervals

Kafka

Batch Processing

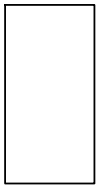
Import Huge Volume ----------> Apply Analysis
SQL SERVER ----------> SQOOP ----------> HDFS |
(Source) (No intermediate storage) (Destination) Batch processing

Real time processing

t0 t1 t2 ….. tn ----------> Real time processing layer ----------> t0 t1 t2 ….

Traditional vs Distributed Machine

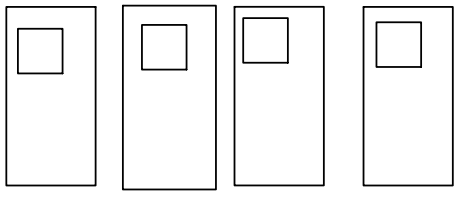
Traditional



- CPU
- 1 TB
- Processing speed of 1 channel - 100 mb/s
- 4 channel I/O
- Time - 43.6 min (approx)

Distributed

1 TB Data stored



- Processing speed = 100 mb/s
- No of channels = 4
- 43 . 6/ 4 = 1.5 10.9 mins (4 machines)
- For 100 machines = 26 sec

Vertical scaling - Increasing capacity of a system
by adding more resources (Storage,RAM,CPU)
or upgrading

Horizontal Scaling - Adding new machine

Challenges of distributed system

- System failure
- Limited bandwidth
- High programming complexity

Soln - Hadoop

Hadoop - framework that allows distributed processing of large datasets across clusters of commodity computers using simple programming models

3 main component of hadoop

Storage HDFS
Processing - MapReduce
Resource Yarn

Characteristics of Hadoop

Scalable - can follow both vertical and horizontal scaling

Flexible - can store huge data and decide to use it later
Reliable - highly available
Economical - can use ordinary computers for data processing

Traditional database system - data sent to program
Hadoop - program sent to data

	Traditional - RDBMS	Hadoop
Data types	structured	Multi and Unstructured
Processing	Limited, No data processing	Processing coupled with data
Schema	Required on write	Required on read
Governance	Standards and structured	Loosely structured
Speed	Reads are fast	Writes are fast
Cost	Software License	Support Only
Resources	Known entity	Growing, Complexities, Wide
	Read many and write many	Read many write once
Best fit use	OLTP Complex ACID Transactions Operational Data Store	Data discovery processing unstructured data massive storage/processing- batch processing

Hadoop configuration

Modes of Hadoop Configuration

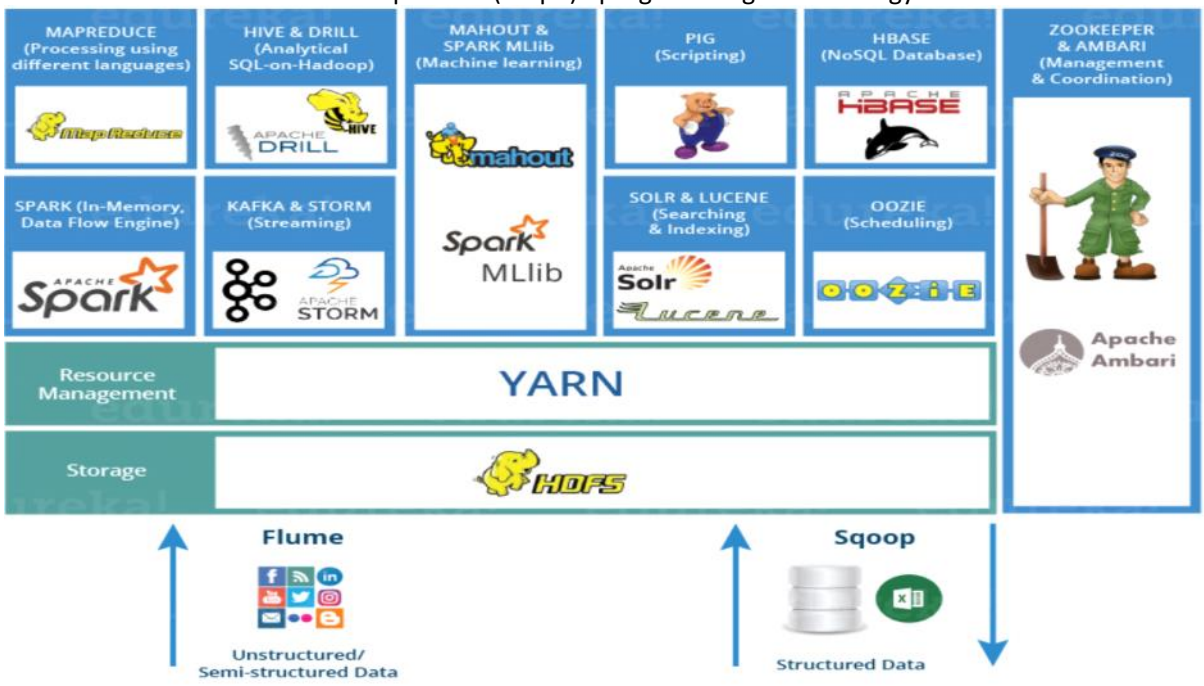
- Standalone Mode
  - All hadoop services run on a single JVM on a single machine
- Pseudo - distributed mode
  - Each hadoop runs on its own JVM but on a single machine
- Fully distributed mode
  - Runs on individual JVM, but these reside in separate commodity machine in single cluster

Hadoop

26 August 2024 15:31

Hadoop

3 components -----------> - Hadoop distributed file system (HDFS) - for distributed storage
- Yet Another Resource Negotiator (YARN) - for efficient resource management while using processing of data
- MapReduce(MapR) - programming methodology



HDFS Architecture

- **Hadoop Distributed File system (HDFS)** is one of the key component of Hadoop ecosystem designed to store and manage large volumes of data across a distributed network of machines.
- Its architecture is designed to handle high throughput (output) access of data and provide fault tolerance and high scalability

HDFS Architecture Components:

- **NameNode:** It act as master server that manages meta data of the file system. Its main role is to keep file system information (path to the data blocks)
    - It keeps all path to the databases (Folder location)
    - Keep tracks of replicas (path) file permission etc.
- **DataNode**: Act as a worker node and it keeps actual block of data
- **Secondary NameNode:**

1 active name node - master

As user u will only interact with NameNode

HDFS has default block size of 128 mb. So if file size is 512 mb. File will be seperated into 4 blocks

Client only interacts with namenode

- All data blocks will be replicated into 3 blocks
- NameNode tries to access nearest DataNode while performing read and write operations

Replicas will be never placed in same rack
Same copy will not be placed in same node

Conditions

- NameNode tries to access nearest DataNode while performing read and write operations
- Name distributes such that all resources will be equally utilized by the name node
- Same data block will be never replicated in the same data node

- ***If any datanode goes down*** *the blocks get replicated (to maintain replication factor) and are put in another data nodes which are available.*
- ***If the previous node comes back*** *- the other one will be dropped/deleted*

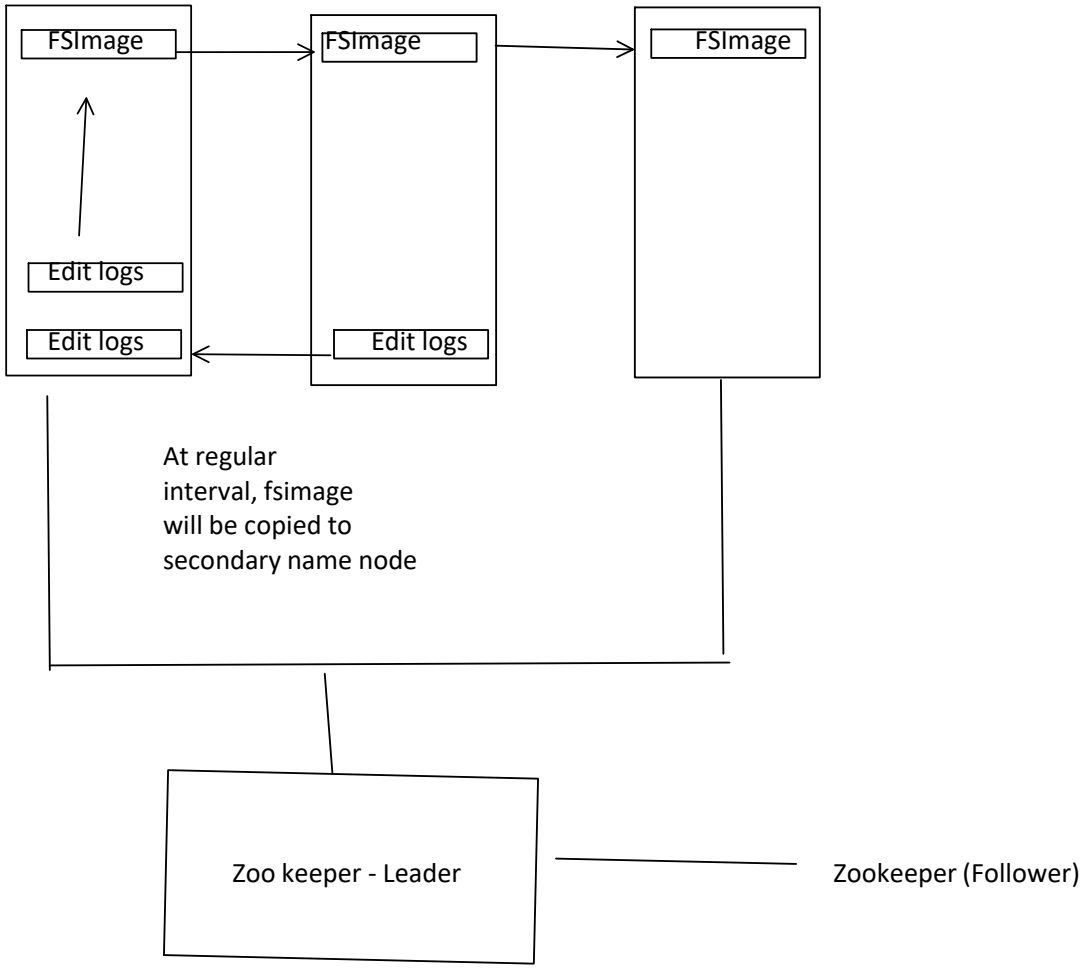
What if NameNode goes down

Concept of Multiple NameNodes:

a. Active NameNode
    i. Edit logs- at regular interval fsimage will be copied to secondary NameNode as a backup
    ii. FsImage
b. Secondary nameNode
    i. FsImage - copied to standby namenode
c. Standby nameNode
    i. FsImage

HDFS HA ARCHITECTURE

secondary active standby



Zookeeper - management

Checkpointing :

- The primary function of secondary namenode is to perform checkpointing
- It involves periodically merging namenodes edit logs (changes in the file system) with current file system (FSImage)
- NameNode keeps edit logs of all the modification made to the file system. Over time this log can grow large hence secondary name node helps to convert this edit logs into file system (FSImage) by creating checkpointing
- Over time if edit logs can grow large, namenode will take more time to convert it into fsimage by restarting it, during this phase cluster will be down.

- Secondary namenode helps in reducing recovery time of the cluster.
- Due to checkpointing the down time of cluster minimises

- FSImage - file system Namespace - Path to the file system in clusters DataNode. (Replicas, replica path, File path, file system etc.)
- In older version of Hadoop Distributed file system (HDFS) , cluster used to be down frequently , logs and FSImage used to take lot of space in memory
- Due to shortage of Ram cluster used to go down . This got resolve din Hadoop 2 after introducing secondary namenode concept

ZooKeeper

It acts (follower- backup , Leader - connected to clusters Namenode) as coordinator , that will monitor health of active namenode and it will makes it decision of automatic fail over where it will replace existing namenode with standby namenode.

Heartbeat

Frequent update about current status of the cluster node (each data node) (datanode ---------> Namenode)

a) How does hadoop achieve fault tolerance?
    a. Replicas of data blocks across muliple nodes
    b. Secondary namenode(Multiple namenode)

b) If a file size is 129 mb, how many data blocks will it create in HDFS?
    a. 2 (128 mb + 1 mb)

c) How does hadoop handles data consistency?
    a. Write once and read many. Once a file is written in HDFS it cannot be modified (It can be only deleted and replaced).It ensures that all data nodes read the same version of the file.

d) How does Yarn differ from original mapreduce frame work in version 1?
    a. In original mapreduce framework (version 1) only job tracker was responsible for both recource management and job scheduling and this led to : scalability issues and regular job tracker failures
    b. Due to overburden of tasks
    c. Now, better scalability and flexibility

e) Suppose mapreduce job has 10 map tasks and 5 reduce tasks . If one of the node running map fails , what impact will it have on overall job execution and how does hadoop handle it?
    a. If a node running a map task fails , Hadoop will reschedule failed map task on another available node. The system ensure job continues by rerunning only failed map task . The fault tolerance mechanism ensure that overall job execution is resilient to individual node failure

f) In a hadoop cluster where multiple applications are running simultaneously what would be impact of increasing replication factor of HDFS on overall system performance.
    a. Positive impact:
        i. Improve data reliability
        ii. Fault tolerance by storing more copies
    b. Negative Impact:
        i. Storage Utilisation
        ii. Write performance (more files will perform write operation)
        iii. Network Traffic (High)

g) How would u optimise performance of mapreduce jobs that can be running longer than expected?
    a. Resource allocation
    b. Data Locality - minimise data transfer across the network by ensuring that tasks run on nodes where data is located
    c. Balancing number of maps and reduce tasks that align with clusters capacity

Hadoop Commands

27 August 2024 09:41

sudo	super user do - Administrator	
mv	move	source_directory destination_directory
cd	change directory	cd newpath
ls	list	Ls
ls-l		
sudo tar xvzf	x - extension v- verbocity z - filtering out .z zip file f - filename	sudo tar xvzf filename
sudo mv filename newname	rename	sudo mv filename newname
Java ws	Java web start	Use to run java application on web server
ssh	Secure shell	For communication
cd ~		
ssh localhost		
sudo apt-get install	install	sudo apt-get install openssh-server
sudo apt-get remove	uninstall	
reboot	restart	
Sudo apt-get update		
Sudo shutdown -h now		
777	drwx-rwx-rwx -rwx-rwx-rwx	d means directory -Means file
7 = 4 (read only) + 2 (write only) +1 (Execute)		
7 (Users permission), 5 (Groups permission), 5 (others permission)		
Mkdir -p Chmod -r	-p to create directories to make directories and sub directories -r recursive path	

Prerequisite for hadoop

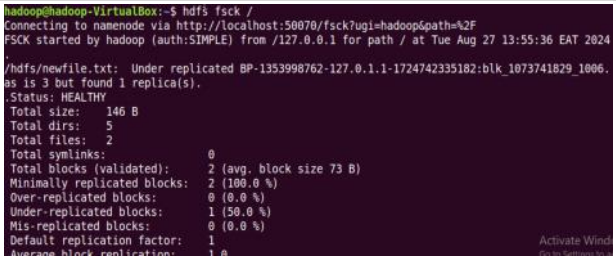
- Java environment
- SSH (secure shell)
- We need to disable ipv6, we only use ipv4

Hadoop namenode -format	to empty namenode
Start-dfs.sh && start-yarn.sh	
jps	
hdfs dfs or hadoop fs	
Hdfs dfs -put newfile.txt /newdir	

Hdfs dfs -cat /newdir/newfile.txt	
Hdfs dfs -touchz /newdir/tempfile.txt	To create new empty file
`hdfs dfs -appendToFile newfile.txt /newdir1/tempfile.txt` `hdfs dfs -cat /newdir1/tempfile.txt`	
`hdfs dfs -appendToFile newfile1.txt newfile.txt /newdir1/tempfile.txt`	Append the contents from newfile1.txt and newfile.txt to the new file Local to hdfs (content copy)
`sudo chmod 777 newfile2.txt` `hdfs dfs -cat /newdir1/tempfile.txt > newfile2.txt`	-cat used to copy contents from hdfs to local (content copy)

copyFromLocal or put
`hdfs dfs -put newfile2.txt /hdfs`
 	
`hdfs dfs -copyFromLocal newfile2.txt /hdfs`
	Copy file from local to hdfs

`hdfs dfs -cp /hdfs/newfile2.txt /newdir`	cp is used to copy file from one directory of hadoop to another directory
`hdfs dfs -copyToLocal /newdir1/tempfile.txt` No destination - means copied to home copyToLocal or get	Copy file from hadoop to local
`hdfs dfs -ls /`	Shows directories
`hdfs dfs -ls -R /`	-R recursive path
`hdfs dfs -rm /hdfs/newfile2.txt`	Delete a file
`hdfs dfs -rm -r /newdir`	Delete a non empty directory
`hdfs dfs -rmdir /hdfs`	To delete empty directory
`hdfs dfs -expunge`	Empty the trash
`hdfs dfs -chmod 766 /hdfs/newfile.txt`	Change access permission of a file
`hdfs dfs -chown hadoop:hadoop /hdfs`	Change ownership
`hdfs dfs -chown -R hadoop:hadoop /hdfs`	Change ownership of file and folder recursively (-R)
`hdfs dfs -stat /hdfs/newfile.txt`	Displays status
`hdfs dfs -stat %r /hdfs/newfile.txt`	Displays status of the replicas
`hdfs dfs -stat %b /hdfs/newfile.txt`	Byte size %b
`hadoop@hadoop-VirtualBox:~$ hdfs dfs -setrep 3 /hdfs/newfile.txt` `Replication 3 set: /hdfs/newfile.txt`	Setup new replicas
`hdfs dfs -du /`	Amount of size occupied by each directory in bytes
`hdfs dfs -du -h /`	-h human readable format
`hdfs dfs -df -h /`	File size whole cluster

`hdfs dfs -count /`	Count displays no of directories and file
	fsck - Returns status of cluster
`hdfs fsck / -files`	File status of each file
`hdfs fsck / -files -blocks -locations`	Files and block locations
`hdfs fsck / -files -blocks -locations -racks`	
`hdfs balancer -threshold 1`	HDFS Balancer utility
Stop-yarn.sh && stop-dfs.sh	Stopping the cluster service
Hadoop fs-Ddfs.blocksize = block size in bytes-put /home/hadoop/test/test.text/hdfs	To change block size of a specific file in a cluster

HDFS Balancer Utility

mapreduce

28 August 2024 13:04

```
  64  hdfs dfs -mkdir /harrypotter
  65  sudo mv /home/hadoop/Downloads/HarryPotter_Sorcerers_Stone.txt  /home/hadoop/
  66  hdfs dfs -put /home/hadoop/HarryPotter_Sorcerers_Stone.txt /harrypotter
  67  hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.7.6.jar -file /home/hadoop/mapr
educe/mapper.py -mapper mapper.py -file   /home/hadoop/mapreduce/reducer.py -reducer reducer.py -input /wordc
ount -output /wordcount/output/
  68  hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.7.6.jar -file /home/hadoop/mapr
educe/mapper.py -mapper mapper.py -file   /home/hadoop/mapreduce/reducer.py -reducer reducer.py -input /harry
potter -output /harrypotter/output/
  69  hdfs dfs -cat /harrypotter/output/*
```

Yarn

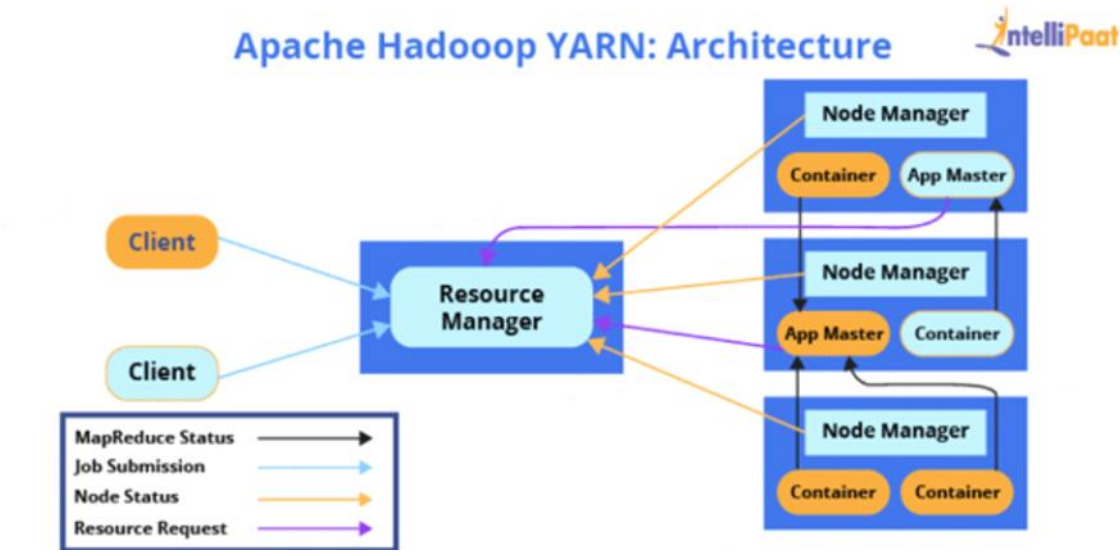
27 August 2024 16:38

YARN (Yet Another Resource Negotiator)

Introduced in hadoop 2.0

Job tracker in hadoop 1

Architecture



Resource manager (Application manager + job scheduler) in master mode
Node manager in slave mode

Job schedular

- When user submits job, it goes to him and it schedules job using FIFO, FAIR, Capacity schedulers
- Allocates resources for job

Application manager

- It will accept job from job schedular
- Request node manager to allocate containers (resources ram, disk, network, datablocks)
- It will monitor job execution flow
- If required more resources it will request to distribute resources as required
- If job fails , it will request to restart

Node manager 1

- Allocate app master to monitor containers

- App master will monitor resources and will negotiate about resources for running job
- App master will running one for each job
- App master will kill itself once job is finished

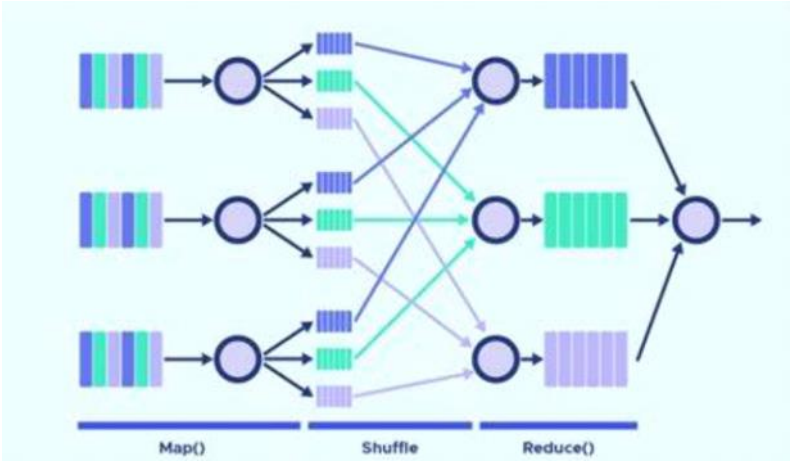
Node manager 2

- Node manager will send the status of slave node using signal (heartbeat) , signal sent at regular interval

Map Reduce parallelism

To speed up the execution of a specific task by splitting task

Mapper can be independent but not reducer



Input Split and record reader

Input ----------> Input split ----------> record Reader ----------> map ----------> intermediate O/p in disk (Block 1)

Input ----------> Input split ----------> record Reader ----------> map ----------> intermediate O/p in disk (Block 2)

Input ----------> Input split ----------> record Reader ----------> map ----------> intermediate O/p in disk (Block 3)

Input split - split the data as key and values
Record Reader - RR picks the data (key,value)
Intermediate output is stored in the disk (HDFS)

Partitioning (after intermediate stage) is pre reducer - It will ensure same key goes to same reduce
Partitioner determines how the output of mapper is distributed across reducers. It decides which key

value pair will go to which reducer - helps in **load balancing**

Combiner - optional local reducer , which performs partial aggregation of mapper output before it sent to the reducer

- It reduces volume of data which is transferred between mapper and reducer

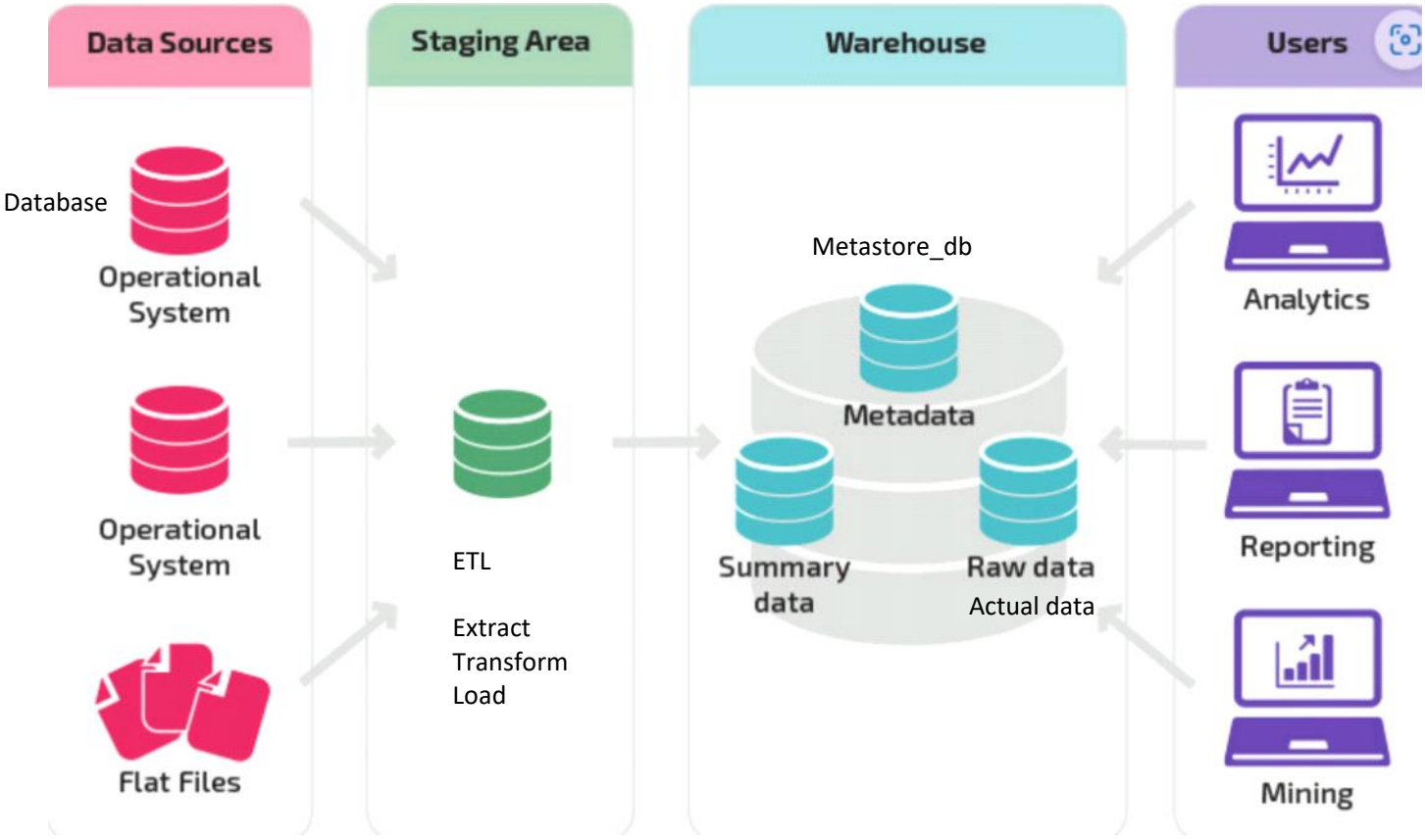
Reducer will do the aggregate operation
Path should be unique in map reduce job
Output will be written back to

ETL

06 September 2024 16:49

Data Warehouse Architecture

Landing zone



Types of transformation

Refer Sir's pdf for studying

Filter
Select
Datatypes
cleaning

Steps in ETL

Complexity

Diversity of source systems
Heterogeneity of source systems

Types of data extraction techniques

- 'as is ' ie, static data extraction
- Data of revision

Data Transformation

- **Major transformation Types**
- Transformation for Dimension

Primary Key

- Uniquely identifies a row

Foreign Key

- Common in both tables

Surrogate Key

- Artificially generated to uniquely identify
- Eg: Random UID generated by the system

Data Loading

- Initial load
    - Populating all data warehouse tables for the 1st time
- Incremental load
- Full Refresh

- During the loads, data warehouse has to be offline
- Find window of time
- Divide up whole load process into small chunks

ETL Tools

Categories

- Data Transformation engines
- Data capture through replication
- Code generators

Ex: SSIS, Informatica, Pyspark

SSIS

07 September 2024 12:05

SQL Server Integration Services

SSIS ETL Tool:

- Tool for data integration
- Allow users to create workflows , dataflows for ETL

Components

- Control flow
  - Manages workflow of tasks
- Data Flow
  - Responsible for ETL
- Event handler
  - Responds to specific events occuring during package execution
- Variables
  - Values which can be dynamically used within the package during runtime
- Package Explorer
  - Provides hierarchical view of SSIS packages

Merge Transformation in SSIS is used to combine 2 or more than 2 sorted inputs into single output. Eg: we have one data source in CSV file and another data source in SQL Server table then using merge transformation we can merge both data sources into a single output data source.

We will use Sort Transformation to sort input source while merge Transformation

Fuzzy Lookup transformation

Fuzzy Lookup transformation is SSIS component that can be used to clean/Standardize or correct data in input data source with a reference dataset.

Fuzzy Lookup Transformation uses fuzzy matching to return one or more close matches in the reference table.

SSIS Questions

12 September 2024 10:54

Full cache , partial cache, no cache difference

Spark

29 August 2024 10:02

Types of big data:

- Unstructured
  - Images and videos
- Structured
  - RDBMS,Excel, Spreadsheet
- Semi Structured
  - Json,xml,csv
  - ORC, Parquet, AURO (Big Data File formats) - these are faster as well as have high compression compared to other file formats

- Quasi structured
  - Web
  - clickstream

Spark

- Real time processing or stream processing

Uber Old Infrastructure

- **Data Sources:** Kafka(Real time), Key value DB, RDBMS
- **Storage**: Vertica (fast, reliable, column oriented)
- **Processing**: ETL (Extract Transform and Load)
- **Problems**
  - Large no of files put strain on the HDFS namenode

New Infrastructure

- **Storage**: HDFS, Hive
- **Processing**: Spark (can handle both real time and batch processing)

Data Processing

Transformation of unstructured data into meaningful and readable info.

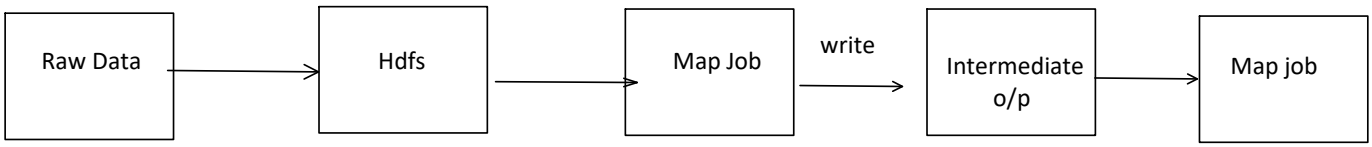
Types

- **Batch processing**
  - Large volume of data is processed in a single batch
  - Collected over a certain period & processed in every batch window
  - Bulk processing

    Challenges
    - Cleaning(null), duplicates, outliers, multiple formats, incorrect data, indexing, inconsistency

- **Real time processing**
  - Data will be send at regular intervals to spark
  - Data is processed as soon as it is generated

Map reduce



Hadoop ecosystem

- Hive (Ad hoc Query + warehouse)
- Pig(Scripting lang)
- Tez(faster version of mapreduce)
- Mahout
- Apache storm (real time computation)

Limitations of mapreduce

Unsuitable for real time
Unsuitable for trivial operations - joins, filters
Unsuitable for large data on network
Unsuitable with OLTP
Unsuitable for processing graph (property graph)
Unsuitable for iterative execution - k means

Spark over MapReduce

- Batch processing
- Structured data analysis
- Machine learning analysis
- Interactive sql analysis
    - Hadoop - no interactive services
    - Spark - shell (scala) , pyspark
- Real time streaming data analysis

Hadoop - Java
Spark - Scala

Apache Storm

Challenge: no framework level support

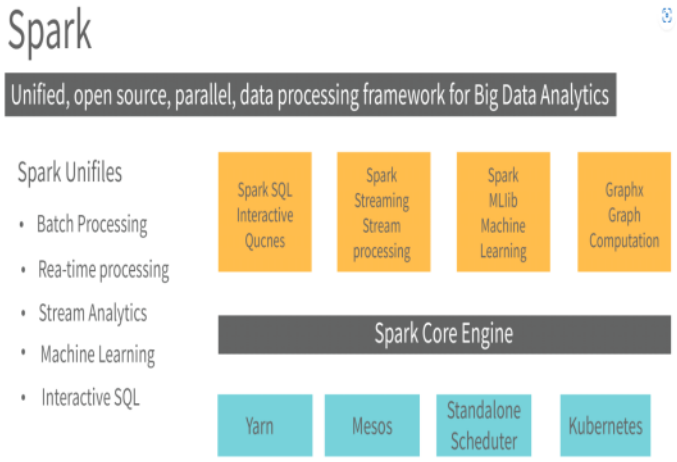
Apache Spark (in memory)

- Open source cluster computing framework for real time data processing.
- It contains:
    - Python
    - Java
    - Scala
    - R - statistical analysis
- **Features**
    - real time operations and to process larger data on network
    - Open source cluster computing framework

- 100 times faster for a few applications and 10 times faster for in disk compared to mapreduce
- Suitable for machine learning algorithms

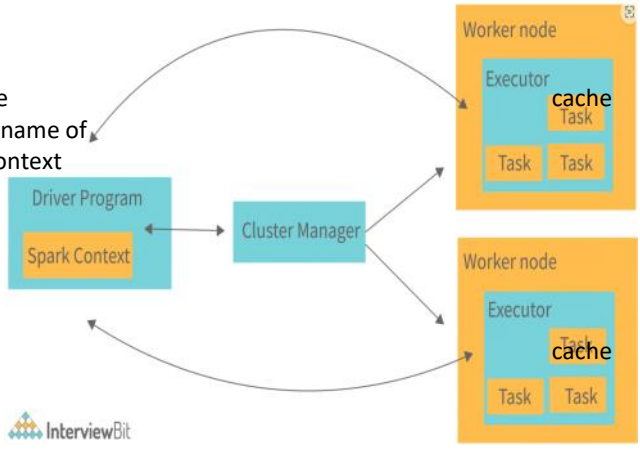
Components of spark

1. Spark SQL
2. Spark Streaming
3. MLLib
4. GraphX
5. SparkR
6. Spark core (every other one is built on top of this)



Spark Architecture

SC is the default name of spark context



Spark 2

30 August 2024 14:21

- when u start pyspark it starts two sessions :
  - spark context
    - Compulsory
    - Entry point
  - spark session

SPARK CONTEXT

- Crucial component that represents connection to the spark cluster
- Entry point for using spark api, methods
- Responsible for managing spark application life cycle
- Resource management - allocation of resources (cpu,memory)required by executors
- This manages configuration for spark application
- Resilient distributed dataset(RDD) - core data structure of spark

Cluster manager

- It schedules and dispatches tasks to the worker node. It ensures that tasks are distributed among the worker nodes
- Scaling - up or down depending on the workload or demand
- Fault tolerance - handles node failures and job failures for maintaining the reliability of cluster
- If job fails it can reschedule the tasks
- Resource management - Monitors and manages clusters resource efficiently

RDD

- Immutable collection of objects which defines the data structure of spark
- Features:-
  - Lazy evaluation
    - Parallelised collections
      - sc.parallelize()
    - Existing RDDs
      - rdd2 = rdd1.transform()
    - External Data
      - sc.textfile()
  - Coarse grained operation
  - In memory computation
  - Fault tolerant
  - Partitioning - no of workers
    - Rdd.repartition() - immense no of partitions
    - Coalesce() - reduce no of partitions
  - Persistent
    - Storage - in disk
  - Immutable
  - Location stickiness

DAG (impt)

Transformations

- Narrow
  - Self sufficient
  - Non aggregate transformation
  - Filter() , map()
- Wide
  - Not self sufficient
  - GroupByKey(),

RDD Persistence Storage Levels

Memory only: only memory, no disk, deserialized

- Uses cache if needed

Memory and disk: deserialized

Memory only ser

Disk only

Memory_only_2 2 - means replication

Memory and disk_2

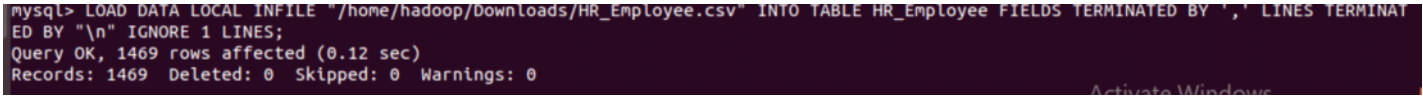
Off Heap - similar to memory only ser

Code ----> spark context -----> job scheduling ---> DAG ---> Stage division ---> Task scheduling(DAG Schedular) ---->Task querying ---------> WORKER 1(Executor)
Assign task to exectors (FIFO)

Worker2 (Executor)

Connecting sql to spark

mysql -u root -p



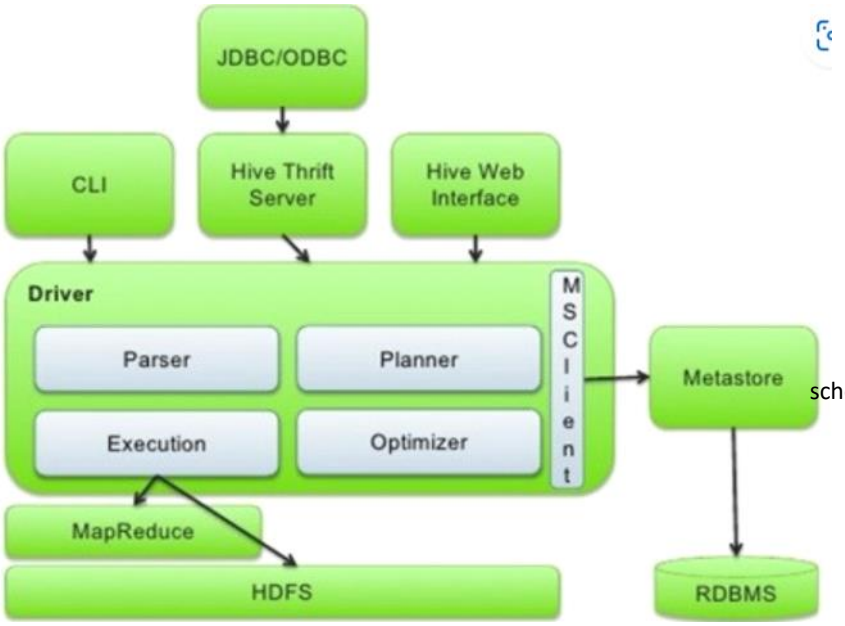
Spark 3

04 September 2024 09:18

Hive

- SQL over hadoop mapreduce
- Part of hadoop ecosystem
- Hive query - runs on top of hadoop services ---> execute each job as map reduce job
- Hive query - map + reduce job
- Sql like open source data warehousing application that extracts data from hadoop and related systems
- Used for batch processing
- Runs query on top of hdfs storage
- Provides high level abstraction layer on top of mapreduce and apache spark
- Useful for structured data (or semi structured) - unstructured can also be used
- Load - used only for inserting data into hive table using load
- Load - cut and paste (in path)

Hive Architecture



HDFS
- File
- Table name
- tables

Metastore_db
Rdbms - durby db or sql

Ways to enter hive
- .hql file format
- Hive cli
- Hue - hive web based interface
- Hive thrift store - Beeline CLI - multiple users can use CLI at same time

Job execution flow in hive

a) **Parser**
   a. Hive query ---> parser ---------> check syntax, error, schema & its datatypes.
   b. If everything is fine it will convert into DAG

b) **Optimizer**
   a. Optimizes the query
   b. We can add catalysts for faster execution

c) **Executor**
   a. Splits job into multiple tasks and executes the task
   b. Fetch the DAG (physical execution plan) and converts into map reduce job

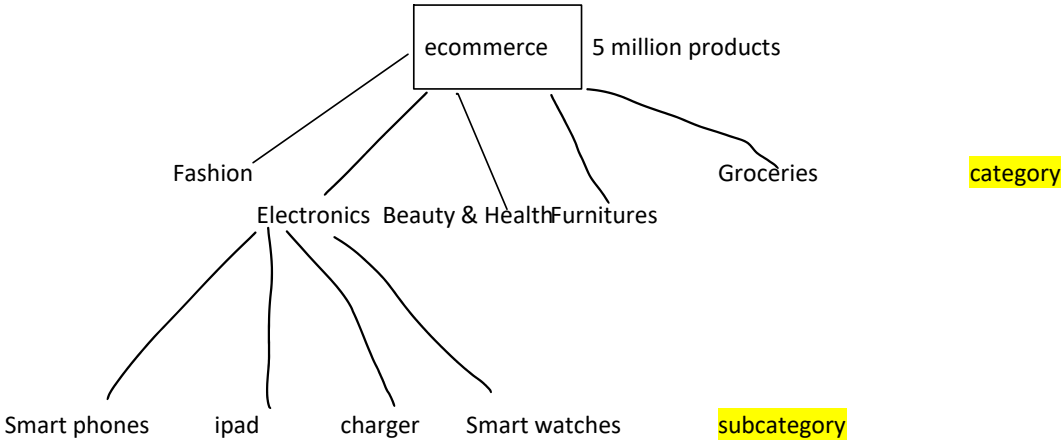
JDBC - sql , java, spark
Odbc - oracle db

Use of jdbc or odbc (open database connectivity) depends on the rdbms
Used to connect 3rd party services to hive also

Yarn - resourse management
Storage - hdfs or hbase(Nosql)

Partitioning and bucketing (another level of indexing in Hive)

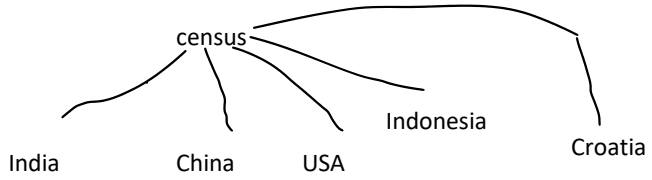
- Splits data
- Partitioning is used to make the query faster
- Search faster but load on namenode
- Namenode have to create that many fs image



Bucketing

- Used for unequal distribution
- In hive bucketing is done by partitioning

- Makes file size same by merging some records together
- If clustered by states into 30 buckets then Every country will create 30 buckets
- For country like india 30 buckets will be filled equally (Makes file size same by merging some records together)
- But for country like croatia 30 records may not be needed so some buckets will be empty



Partitioning + Bucketing (into 30 Buckets) - partitioning and bucketing is different

Partioning is a map job
Bucketing is mapper + reducer

Cluster by - is buckets

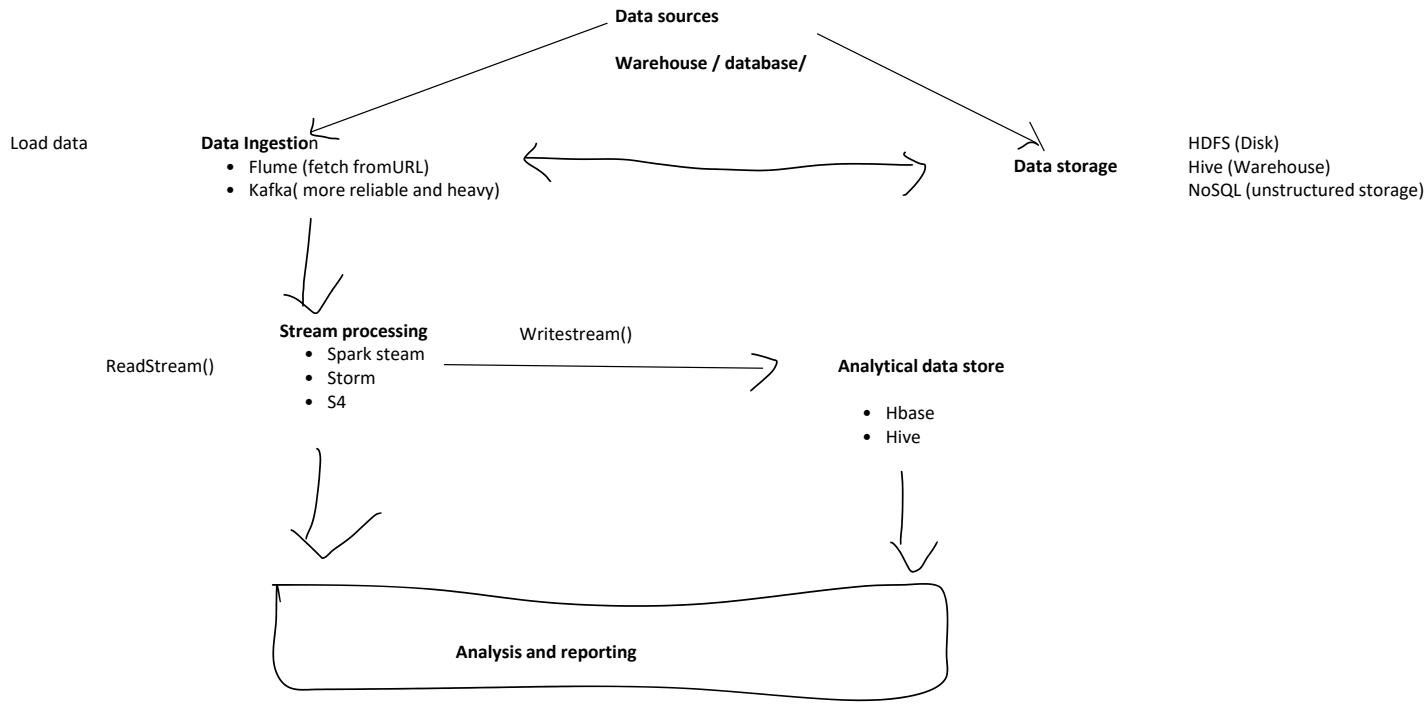
Streaming

04 September 2024 12:52

Streaming

- Continuous flow of data at high speed rate

Life cycle of real time big data processing



Batch layer

Raw data
Compute batch views for consumption
Manages historical data
Recomputes result
Operates on full data
Most accurate results
High latency

Real time layer

- Receives arriving data
- Performs incremental updates to the batch
- Incremental algo implemented at the speed layer
- Reduced computation

Serveing layer - queries are run on this

Intro to Spark Streaming

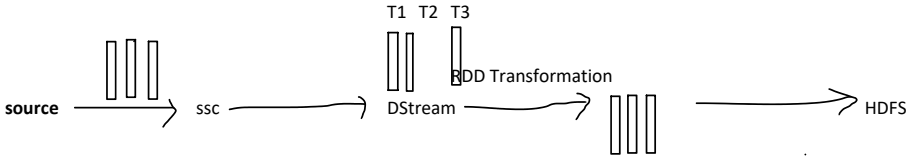
- **Kafka**
  - Act as broker
  - Producer ----> Broker -----> Consumer

- Ssc - spark structured streaming context
  - Entry point to spark streaming

- **Features of Spark streaming**
  - Scaling
  - Speed - low latency
  - Fault tolerance
  - Integration
  - Business Analysis

- **Workfolw of spark streaming**
  - Data is read as Dstream

  Dstream (Basic abstraction in spark streaming)
  - Discretized stream format



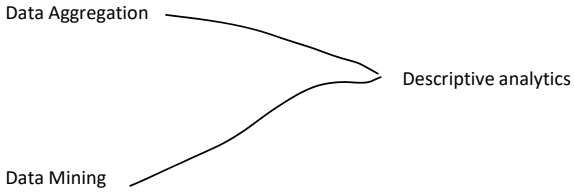
URL
Twitter
HDFS
Flume
Kafka
Web socket

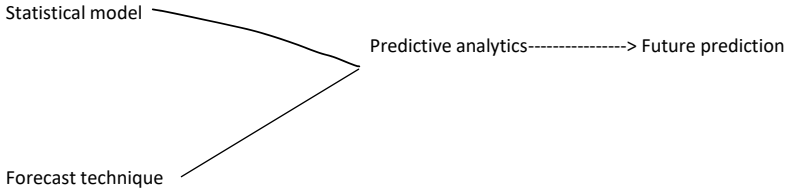
MLib Spark

05 September 2024 10:21

Types of analytics

1. **Descriptive Analytics**
   - Historical data (Insights about the data)
2. **Predictive analytics**
   a. All machine learning models





3. **Prescriptive analytics**
   a. What should be done?

MLlib - scalable ml library for spark

Spark ML Tools

- ML Algos
- Featurization
  - - feature vector
- Pipelines
  - Stage 1 + stage 2 +stage 3
- Persistence

String Indexer()

- String ----------> Numerical value

Gender
Male 1
Female 0

Contract
Month to month
One year
Two year

String Indexer() ---------->

results
0
1
2

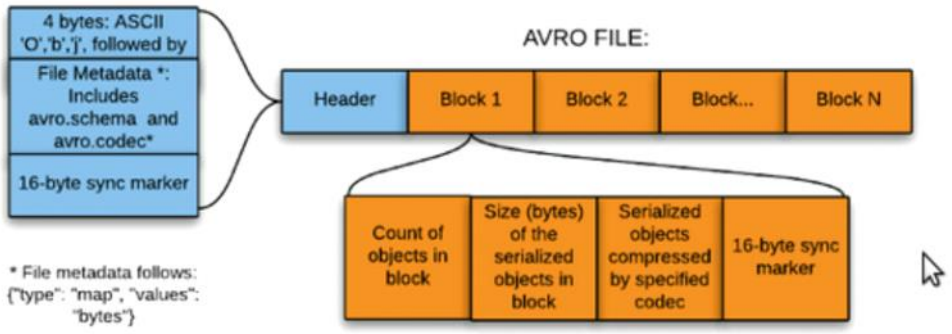
One hot encoder

For each row only one value will be zero

Big data

06 September 2024 09:32

Types of Big data File formats (Read clairvoyant web page)

- Avro
  - Row based storage format for hadoop
  - Stores schema in json format
  - Data itself is stored in a binary format making it compact and efficient in Avro files.
  - 

  - No compression in Avro, but ideal for landing zone for faster reads then transforming it
- Parquet
  - Open source file format for hadoop
  - Stores nested data structures in flat columnar format

Whereas, the same data in a Column-oriented storage format will look like this:

1	2	3	emp1	emp2	emp3	d1	d2	d3

The columnar storage format is more efficient when you need to query a few columns from a table. It will read only the required columns since they are adjacent, thus minimizing IO.

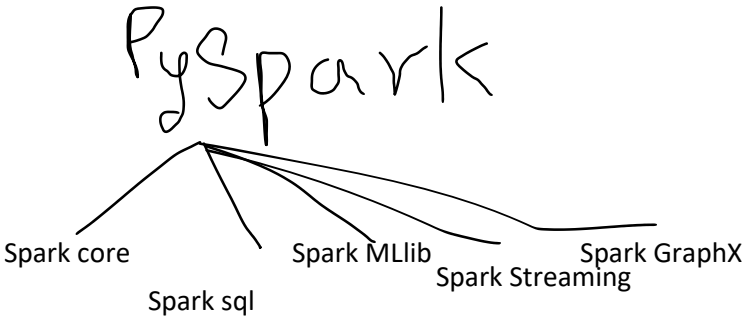
For example, let's say you want only the NAME column. In a row storage format, each record in the dataset has to be loaded, parsed into fields, and extracted the data for Name. The column-oriented format can directly go to the Name column as all the values for that column are stored together. It doesn't need to go through the whole record.

So, the column-oriented format increases the query performance as less seek time is required to go to the required columns, and less IO is required as it needs to read only the columns whose data are required.

- ORC

Spark GraphX

06 September 2024 12:22



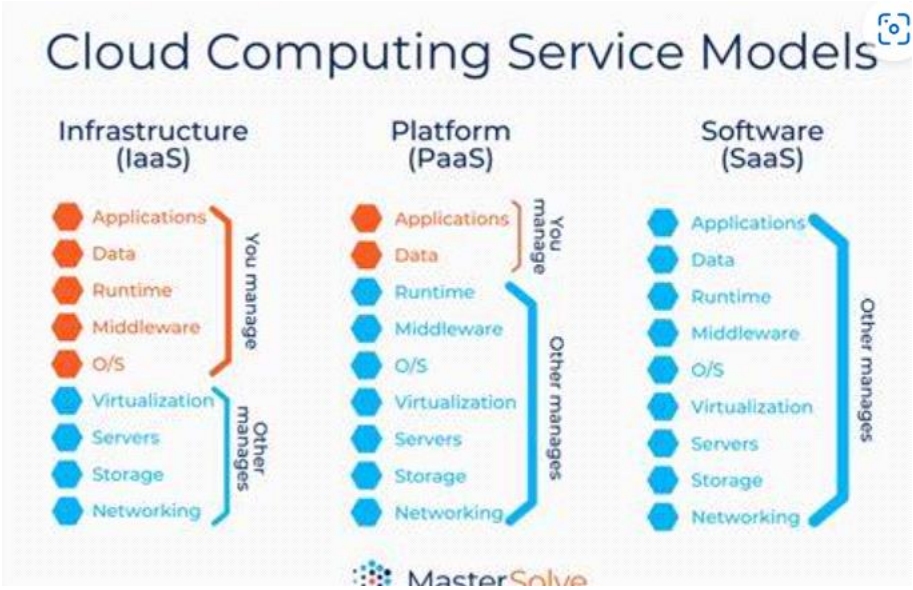
Cloud Computing

12 September 2024 15:19

Virtualization

Cloud Services

Iaas - Infrastructure as a service
Paas -



Azure

17 September 2024 09:27

The Storage Dilemma

Capacity charges
Backup and disaster recovery
Unmanageable complexity
Limited IT agility

Azure Storage
Azure storage offers a massively scalable object store for data objects, a file system for the azure cloud, a messaging store for reliable messaging and a NoSQL Store.

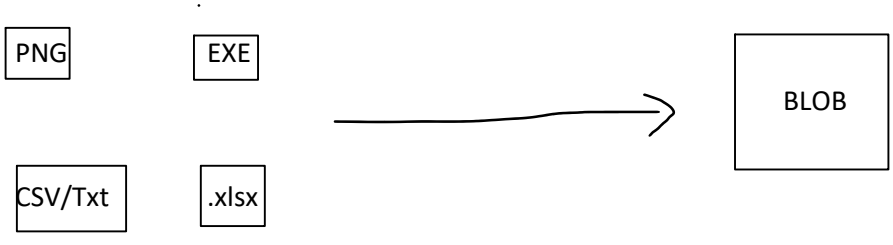
Azure storage characteristics :

- Durable
- Secure
- Scalable
- Fault Tolerance
- High Availability
- Managed
- Accessible

Blob Storage

A massively scalable object store for text as well as binary data. This is ideal for serving images or documents directly to a browser. Blob stores files for distributed access. Streaming video and audio. Blob offers backup and restore disaster recovery and archiving.

Each file will be accessible using same URI



Blob Storage Access Tier

Azure storage provides different options for accessing block of blob data based on usage patterns.

Hot: Optimized for frequent access of objects
Cool: Optimized for storing large amount of data which is infrequently accessible and stored for at 30 days.
Archive: Optimized for data that can tolerate several hrs of retrieval latency and will remain in the archive tier for at least 180 days.

Queue Storage

Msg store for reliable msging between application components.
Its characteristics are messaging queue for asynchronous communication b/w application components
This stores and retrieves msgs in FIFO order.

Storage for small pieces of data (source will generate) small msgs at regular intervals.
Highly scalable , as this can handle millions of msgs per sec. This can trigger azure services, functions for automation processing.

Usage:
Processing log message, User interaction with app and background tasks

Table storage

- Azure storage table is no sql key-val storage with fast access and schema less design
- Scalable and designed for

File Store

Fully managed file share accessible via SMB(Server Message Block) and NFS(Network File System) protocols.
Server message protocol is a network file sharing protocol that allows applications on cloud to read and write to files and request services from server programmes. This can also mounted to drives of computer.

DATA REDUNDANCY

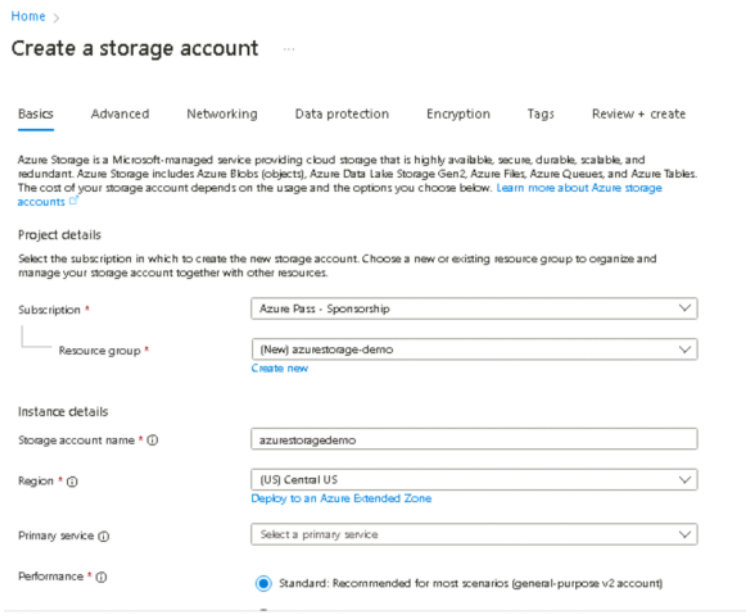
Azure storage that replicates multiple copies of your data. Replication options for a storage include:

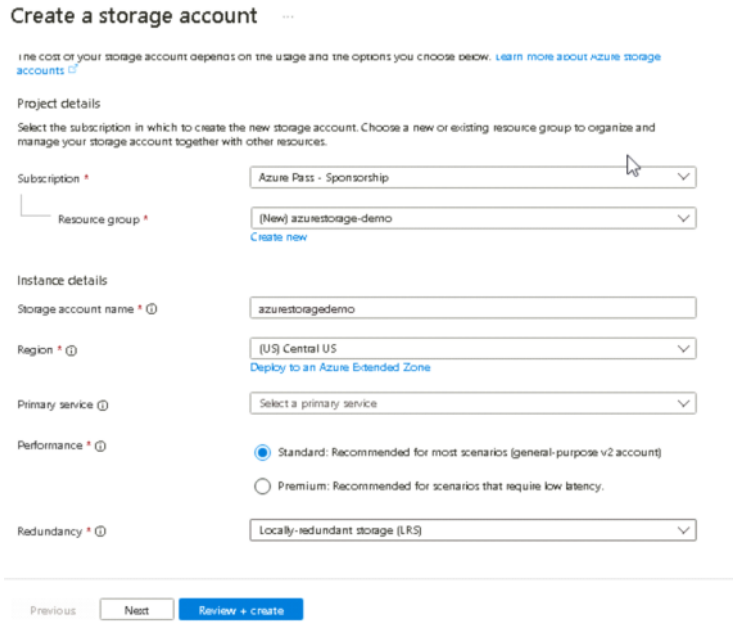
a. Local redundant Storage (LRS)
b. Zone-redundant storage (ZRS)
c. Geo Redundant storage (GRS)

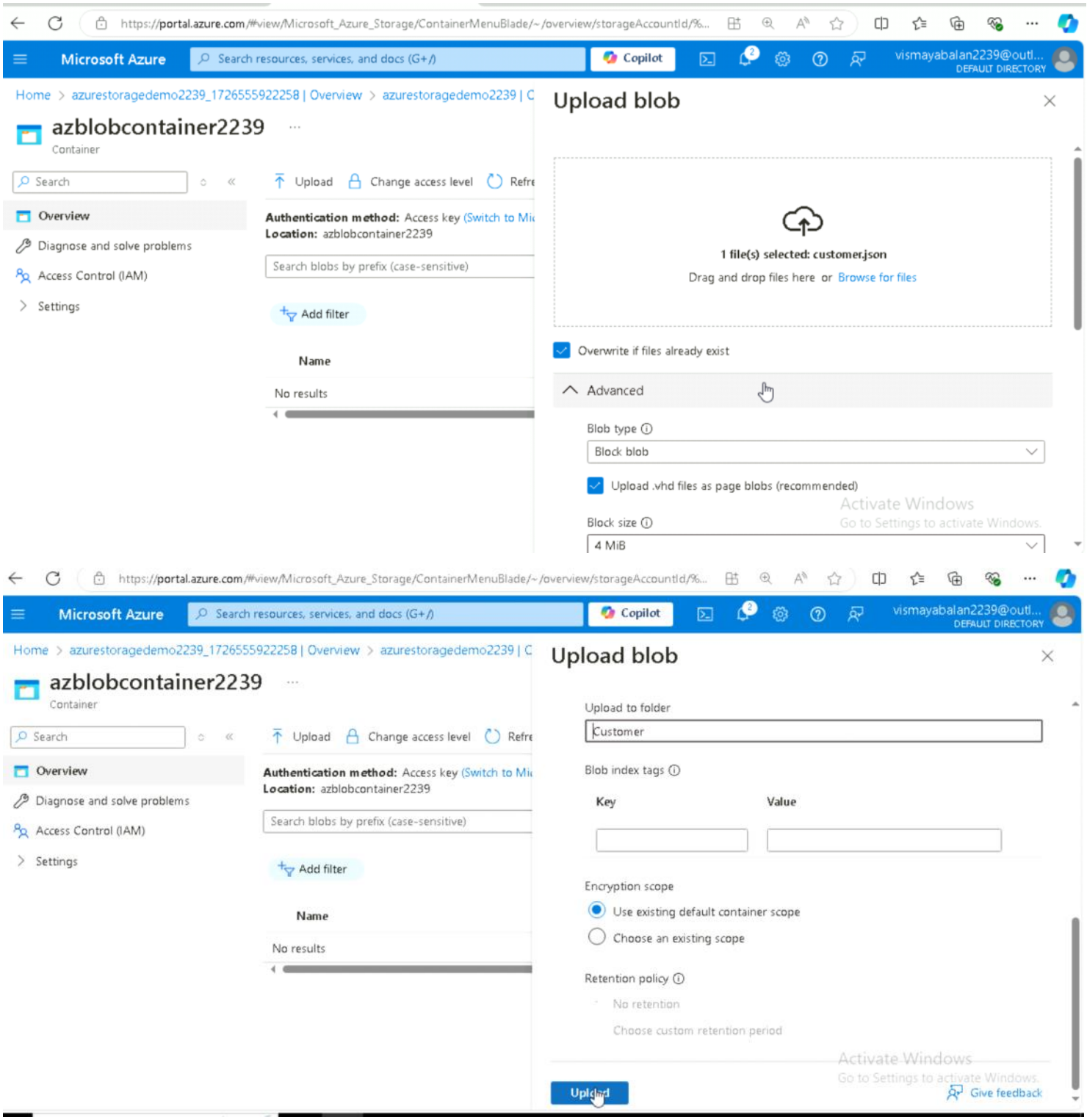
a. **Local redundant Storage (LRS)**
    1. Simple , low cost data replication strategy.
    2. Data is replicated within a single storage scale unit .
b. **Zone-redundant storage (ZRS)**
    1. Use this for high availability
    2. Data is replicated synchronously across 3 available zones
c. **Geo Redundant storage (GRS)**
    1. Cross regional replication to protect against region wise unavailability of data.
d. **Read -Access Geo Redundant Storage (RA-GRS)**
    1. Cross regional replication with read access to the replica

Azure Practicals

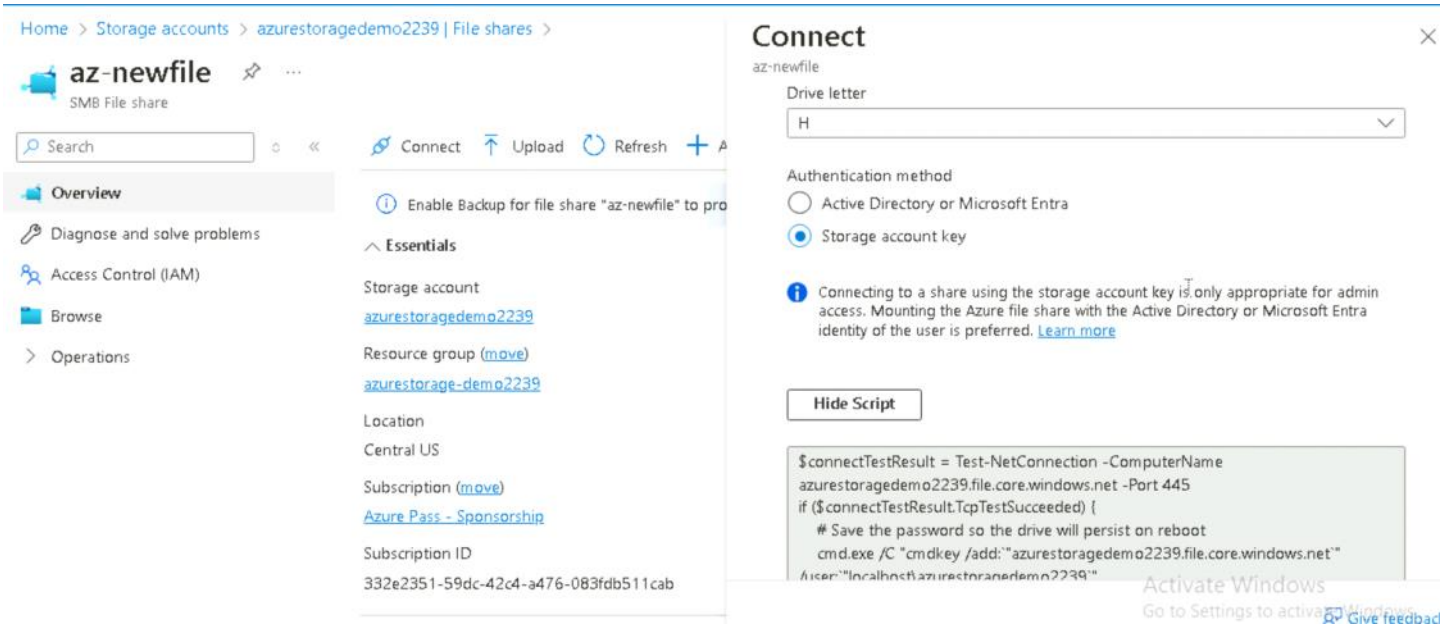
17 September 2024 12:07






Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
vismayabalan2239@outl...
DEFAULT DIRECTORY
Home > azurestoragedemo2239_1726555922258 | Overview > azurestoragedemo2239 | C
azblobcontainer2239
Container
Search
Overview
Diagnose and solve problems
Access Control (IAM)
Settings
Upload
Change access level
Authentication method: Access key (Switch to Mi
Location: azblobcontainer2239
Search blobs by prefix (case-sensitive)
Add filter
Name
No results
Upload blob
1 file(s) selected: customer.json
Drag and drop files here or Browse for files
Overwrite if files already exist
Advanced
Blob type
Block blob
Upload .vhd files as page blobs (recommended)
Block size
4 MiB
Activate Windows
Go to Settings to activate Windows.

Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
vismayabalan2239@outl...
DEFAULT DIRECTORY
Home > azurestoragedemo2239_1726555922258 | Overview > azurestoragedemo2239 | C
azblobcontainer2239
Container
Search
Overview
Diagnose and solve problems
Access Control (IAM)
Settings
Upload
Change access level
Authentication method: Access key (Switch to Mi
Location: azblobcontainer2239
Search blobs by prefix (case-sensitive)
Add filter
Name
No results
Upload blob
Upload to folder
Customer
Blob index tags
Key
Value
Encryption scope
Use existing default container scope
Choose an existing scope
Retention policy
No retention
Choose custom retention period
Upload
Give feedback
Activate Windows
Go to Settings to activate Windows.



Run the script in powershell then u get the mounted fileshare.

Azure

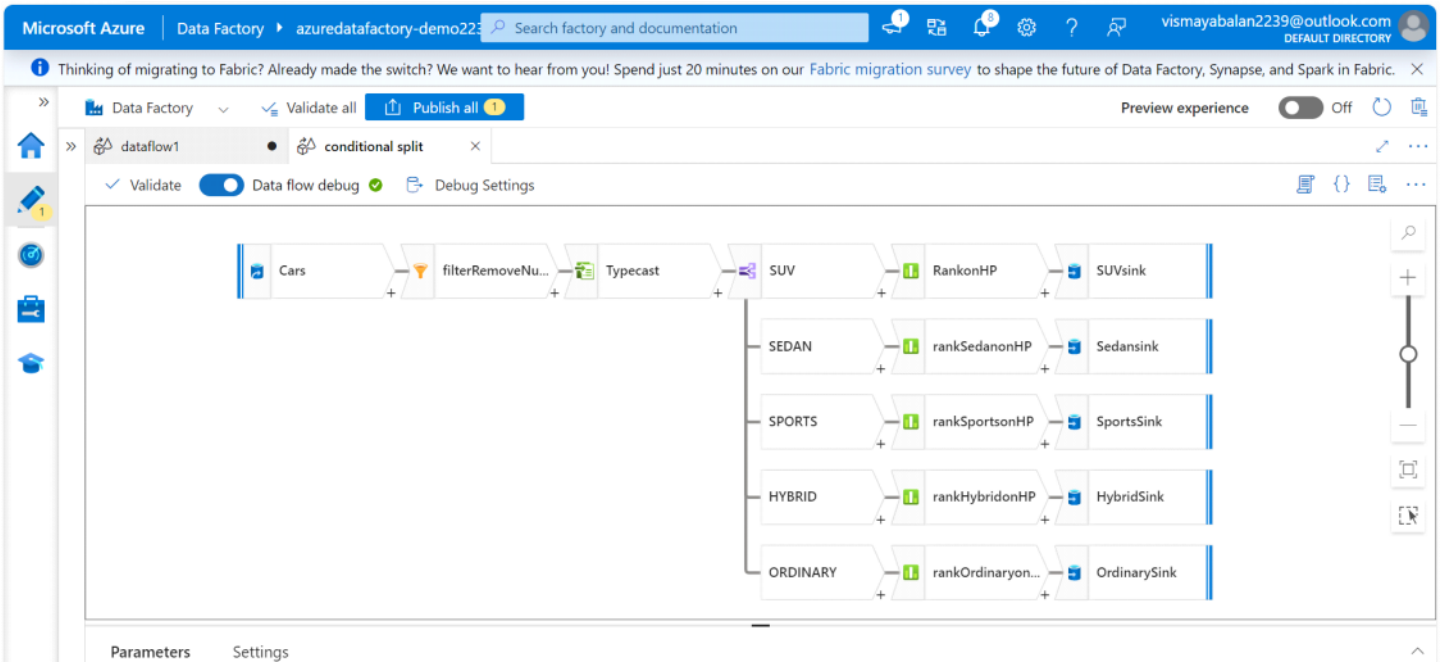
20 September 2024 09:47

ETL Pipeline using Azure Synapse

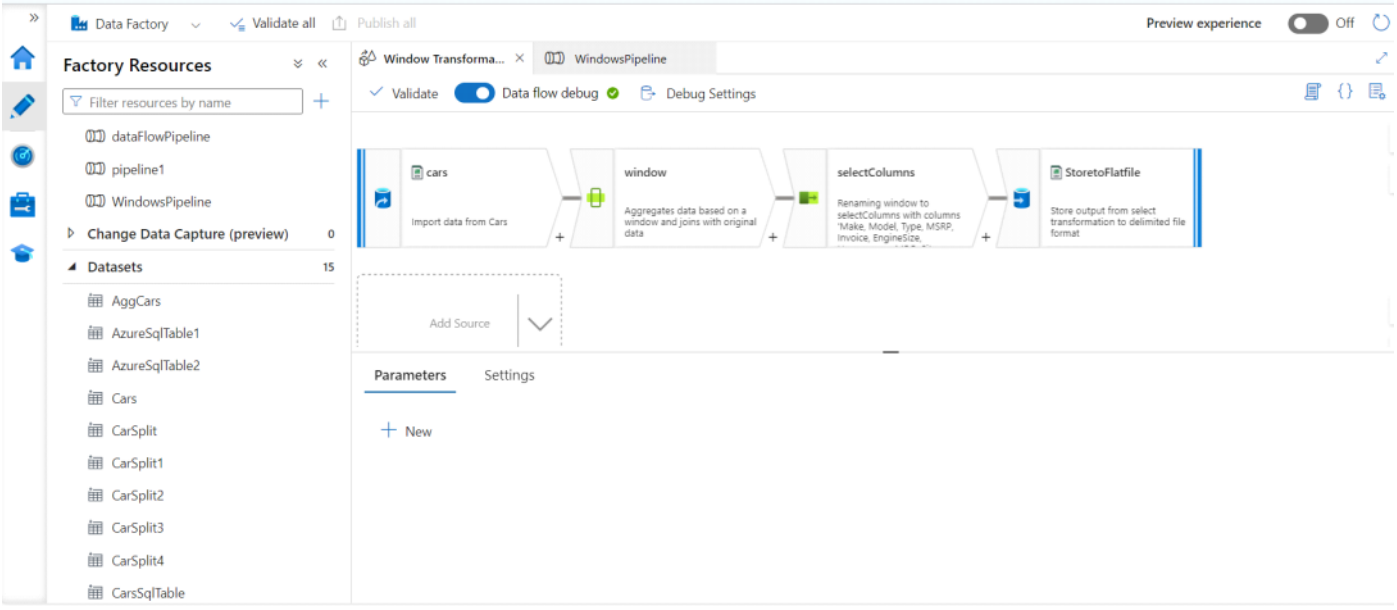
Azure Data Factory

21 September 2024 09:28

1. Conditional split Transformation :
    a. Routes data flow to multiple streams based on matching conditions.
    b. This split transformation is similar to CASE/IF decision structure in any programming language



2. Window Transformation in ADFs
    a. Window transformation in ADF is used to perform operations across a set of rows within a window of data, much like windowing functions in SQL just like OVER().
    b. This allows you to compute values such as running/moving aggregate operations, rankings, averages.
    c. Key aspects:
        i. In the expression builder you can define different types of aggregations

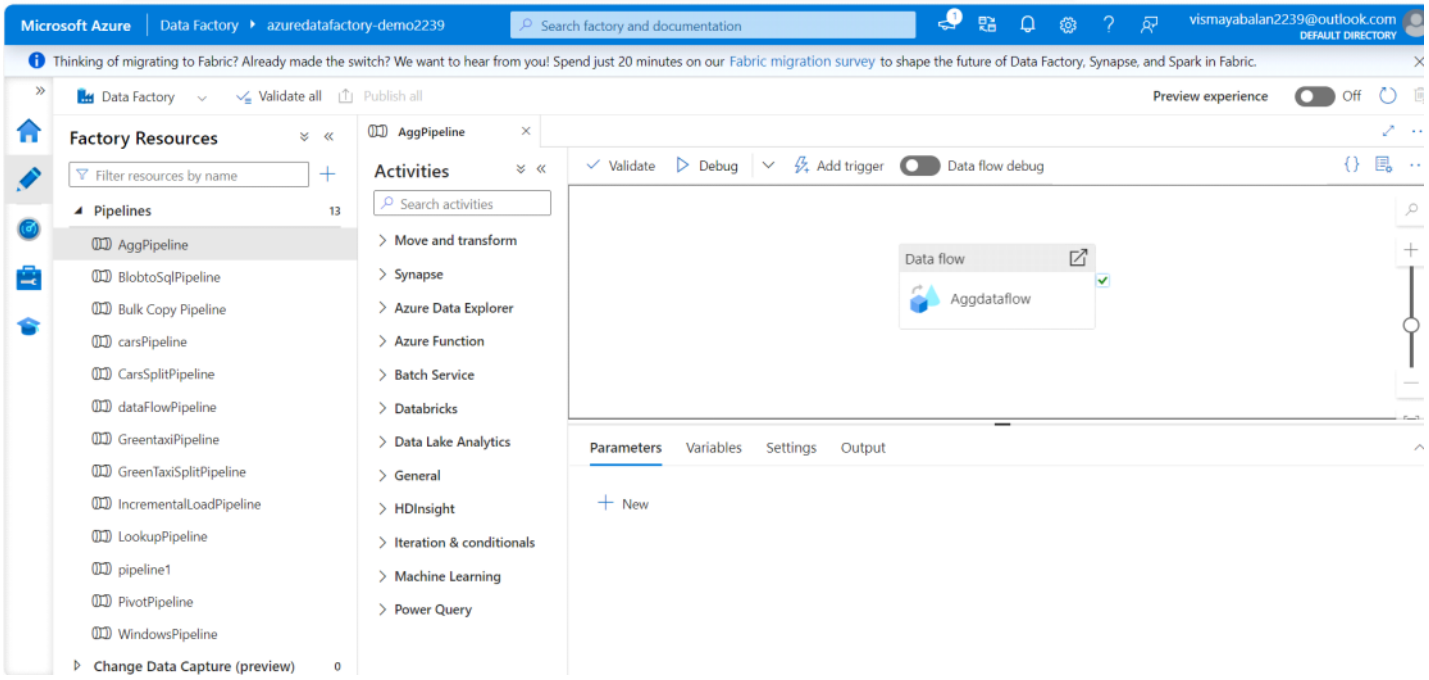


3. Lookup Transformation
    a. Similar to sql left join

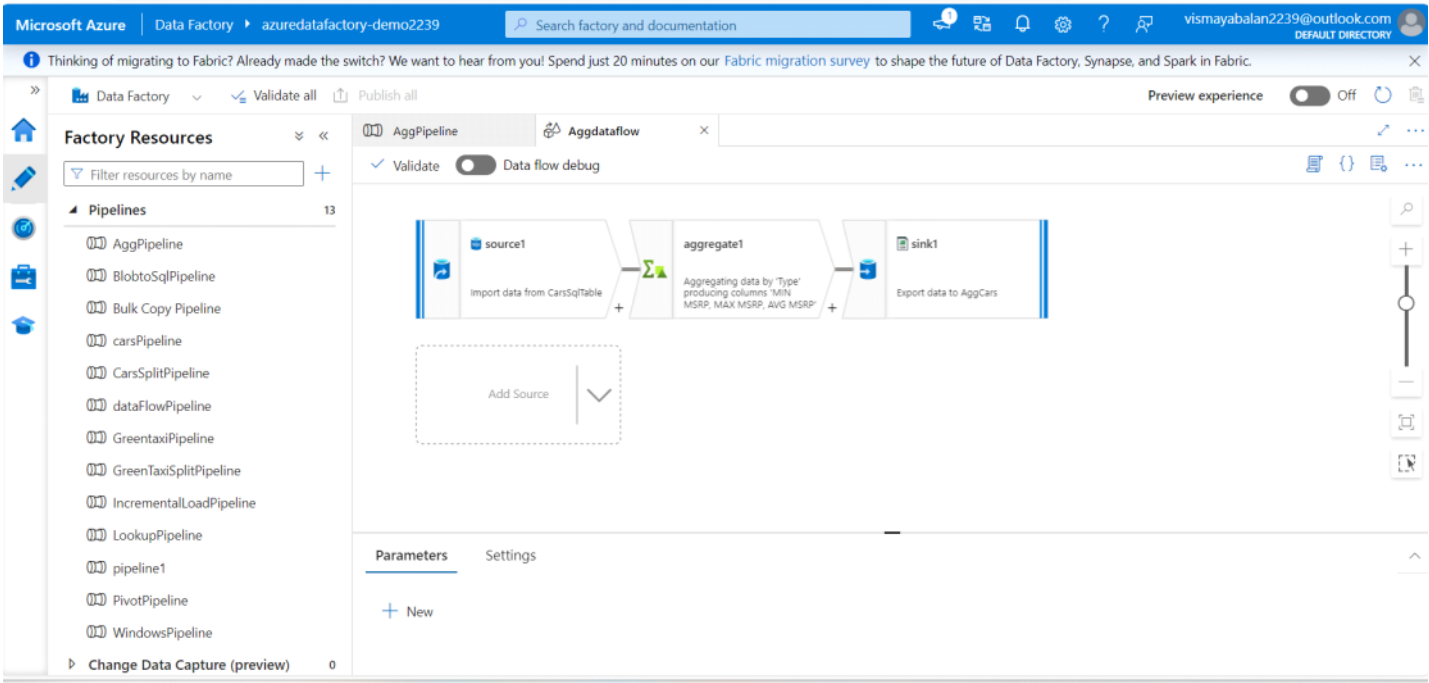
Task

Create multiple tables foreach RatecodeID
except 6 and 99 -> bulk copy all files from DB to BlobStorage

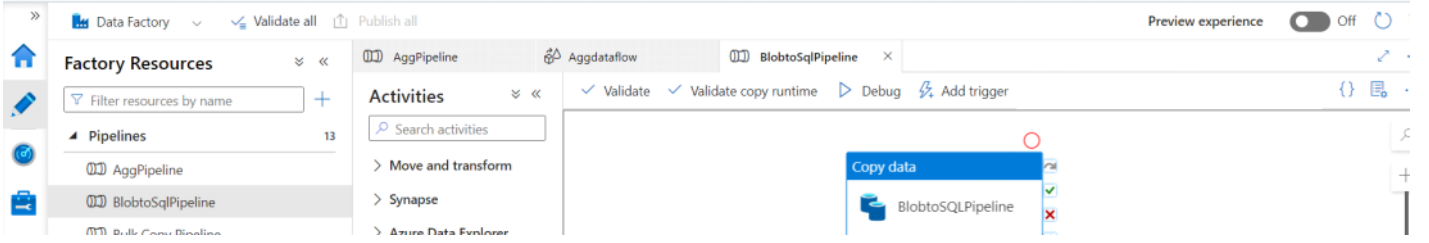
4. Agg Pipeline

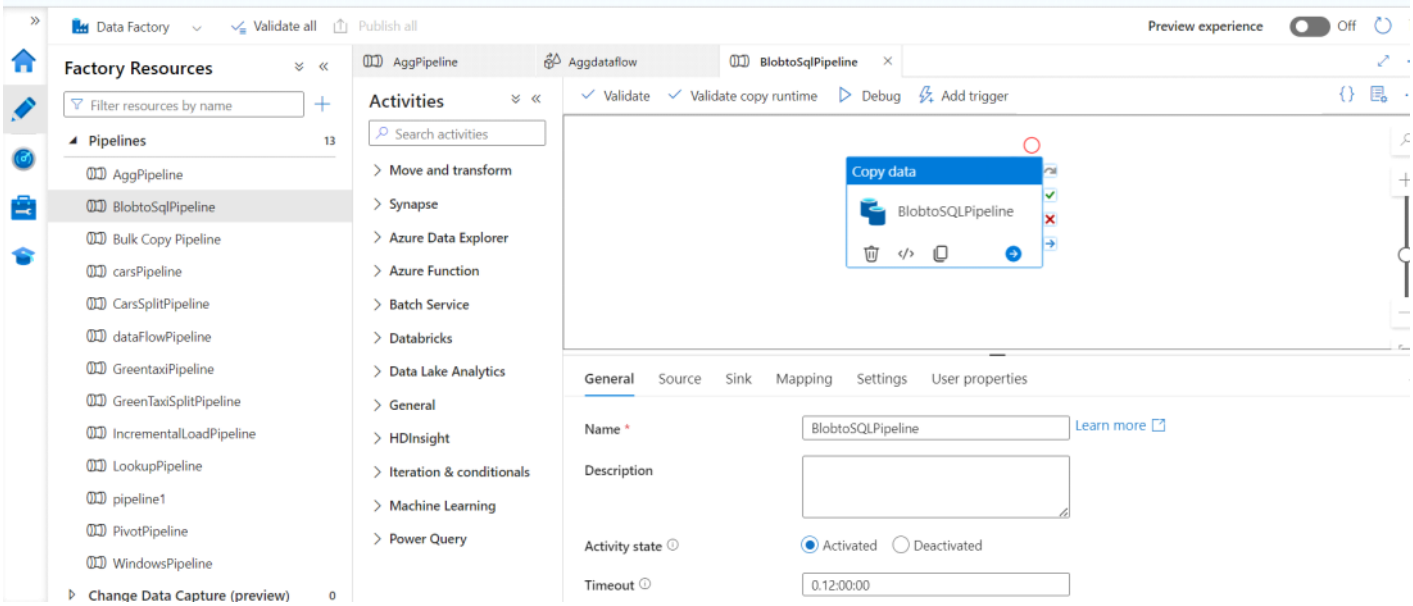


Dataflow



5. Blob to SQL Pipeline




Data Factory
Validate all
Publish all
Preview experience
Off
Factory Resources
Filter resources by name
Pipelines 13
AggPipeline
BlobtoSqlPipeline
Bulk Copy Pipeline
carsPipeline
CarsSplitPipeline
dataFlowPipeline
GreentaxiPipeline
GreenTaxiSplitPipeline
IncrementalLoadPipeline
LookupPipeline
pipeline1
PivotPipeline
WindowsPipeline
Change Data Capture (preview) 0
AggPipeline
Aggdataflow
BlobtoSqlPipeline
Activities
Search activities
Move and transform
Synapse
Azure Data Explorer
Azure Function
Batch Service
Databricks
Data Lake Analytics
General
HDInsight
Iteration & conditionals
Machine Learning
Power Query
Validate
Validate copy runtime
Debug
Add trigger
Copy data
BlobtoSQLPipeline
General
Source
Sink
Mapping
Settings
User properties
Name *
BlobtoSQLPipeline
Learn more
Description
Activity state
Activated
Deactivated
Timeout
0.12:00:00

Databricks

24 September 2024 13:09

https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/2460785922056449/3207523812404718/6795197445702991/latest.html

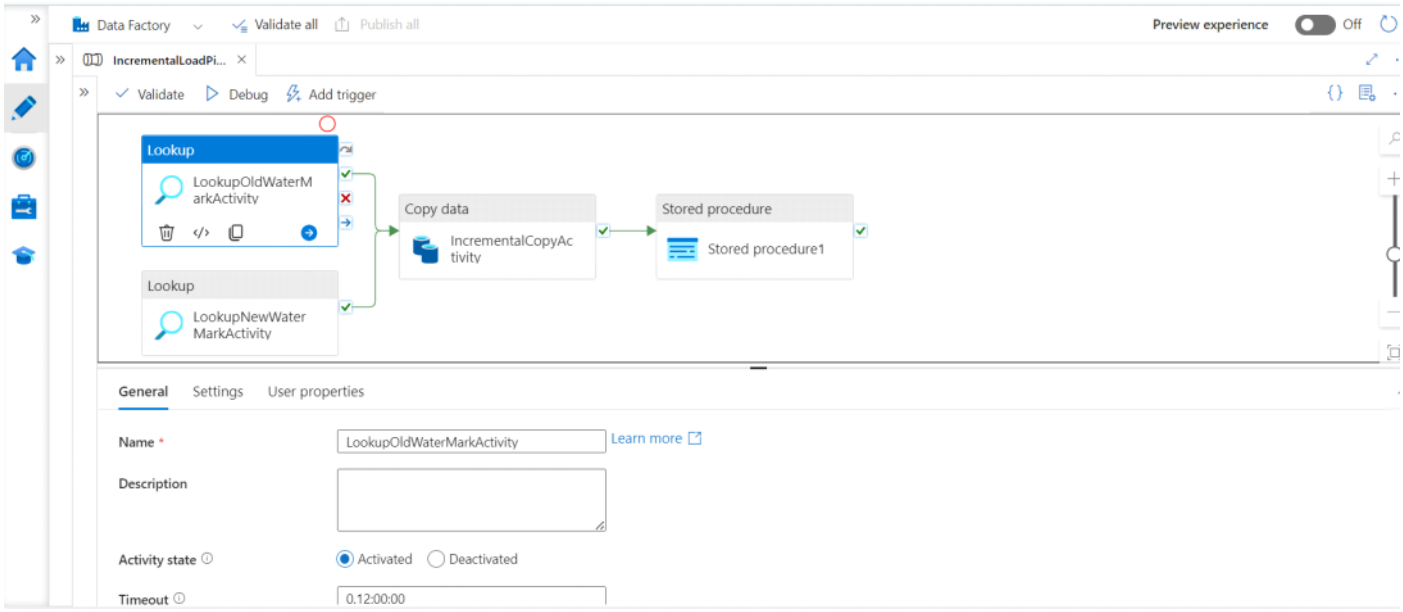
Data Lake vs Delta Lake

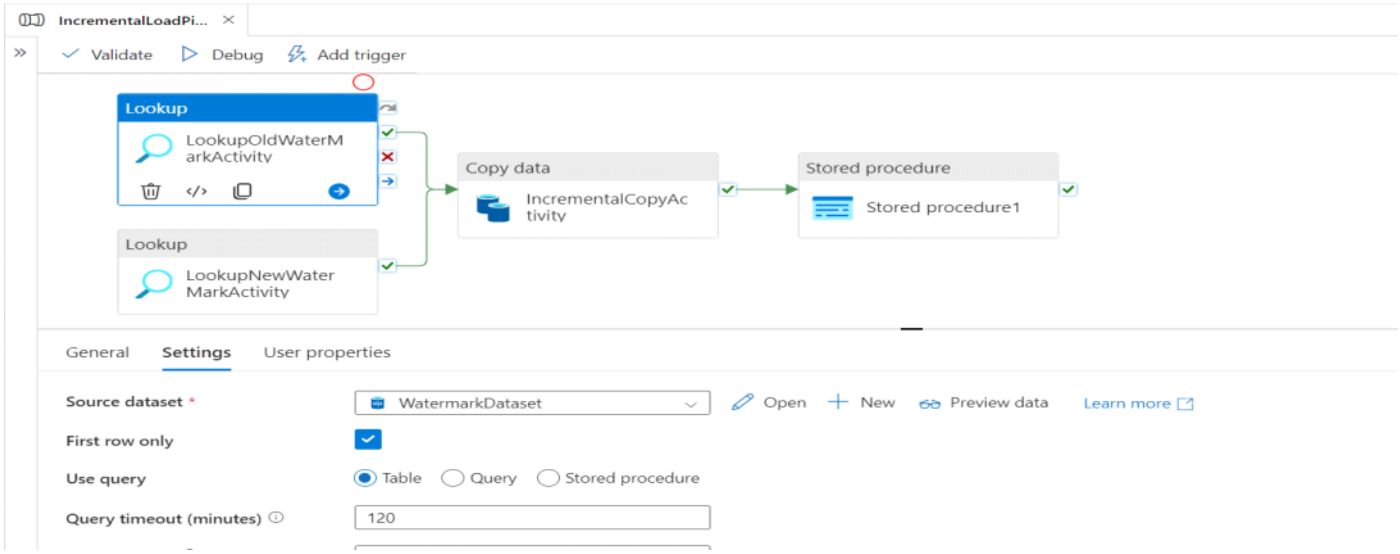
26 September 2024 12:35

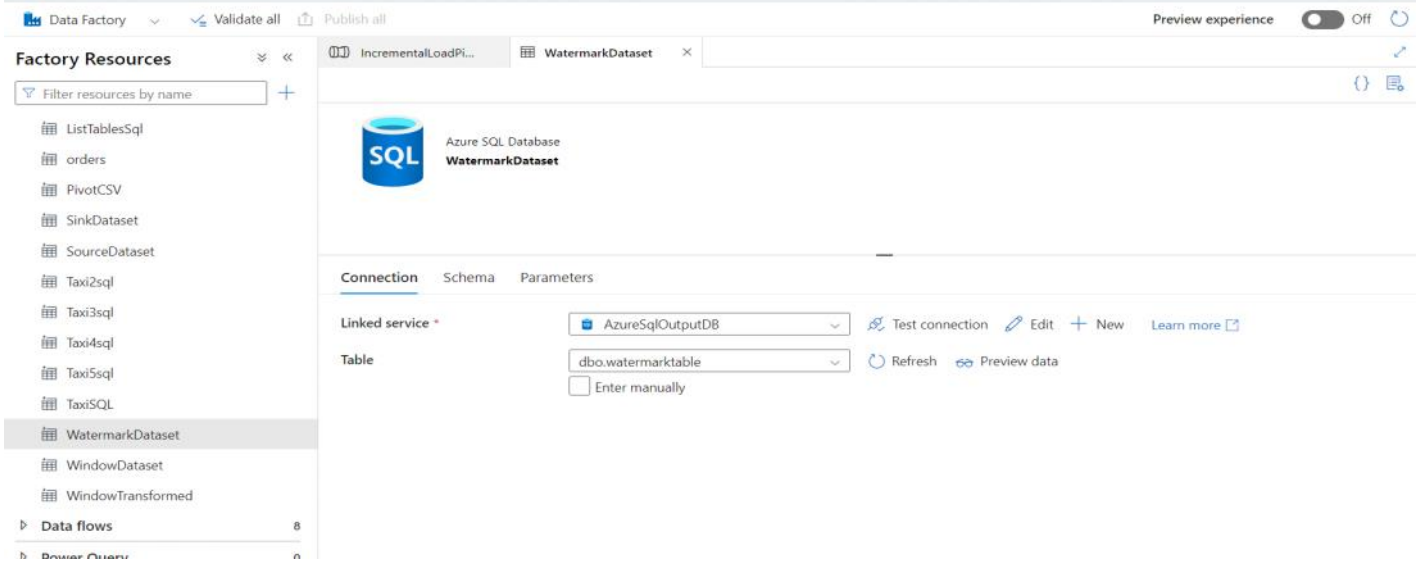
Characteristic	Data Lake	Delta Lake
Definition and Purpose	Raw data storage in native format	Storage layer with ACID transactions
Data Structure	Original format, no enforced schema	Schema-on-write, supports schema evolution
Data Management and Integrity	Lacks integrity features, prone to inconsistencies	Supports ACID transactions for data integrity
Performance	Slower query performance	Performance optimizations (caching, indexing)
Data Governance	Limited governance tools	Built-in governance, metadata management, lineage tracking
Concurrency and Collaboration	Issues with concurrent reads/writes	Handles concurrent operations safely
Data Retrieval and Querying	Requires separate processing frameworks	Unified querying with SQL integration
Integration with Ecosystem	Custom integration required	Designed for compatibility with Spark and modern tools
Cost Considerations	Lower storage costs, potential inefficiencies	Potentially higher costs but better performance and management
Use Cases	Big data analytics, data exploration	Reliable ETL pipelines, real-time analytics

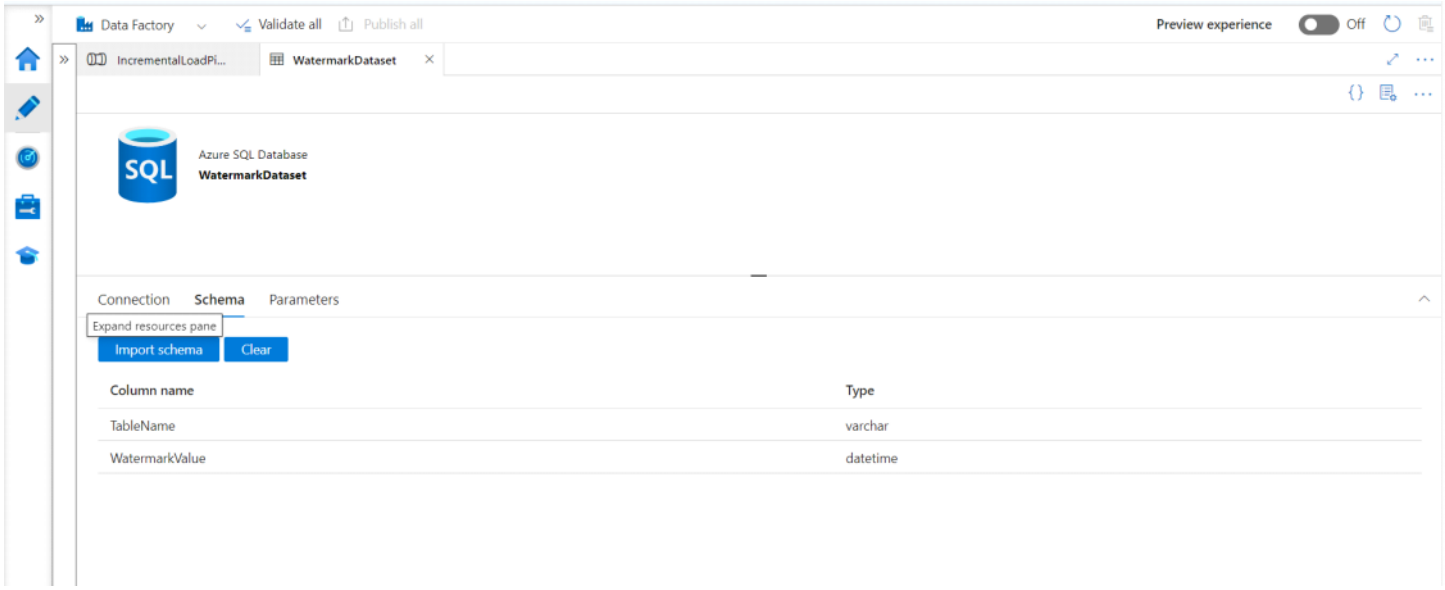
Incremental load

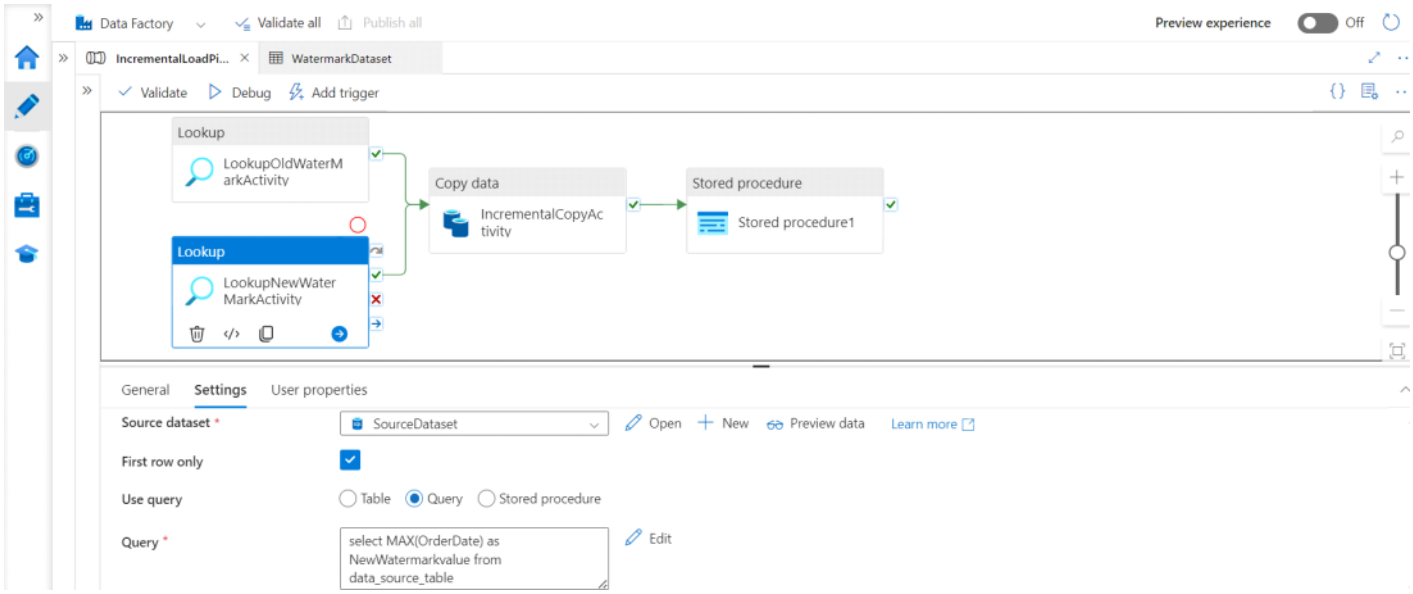
27 September 2024 09:23

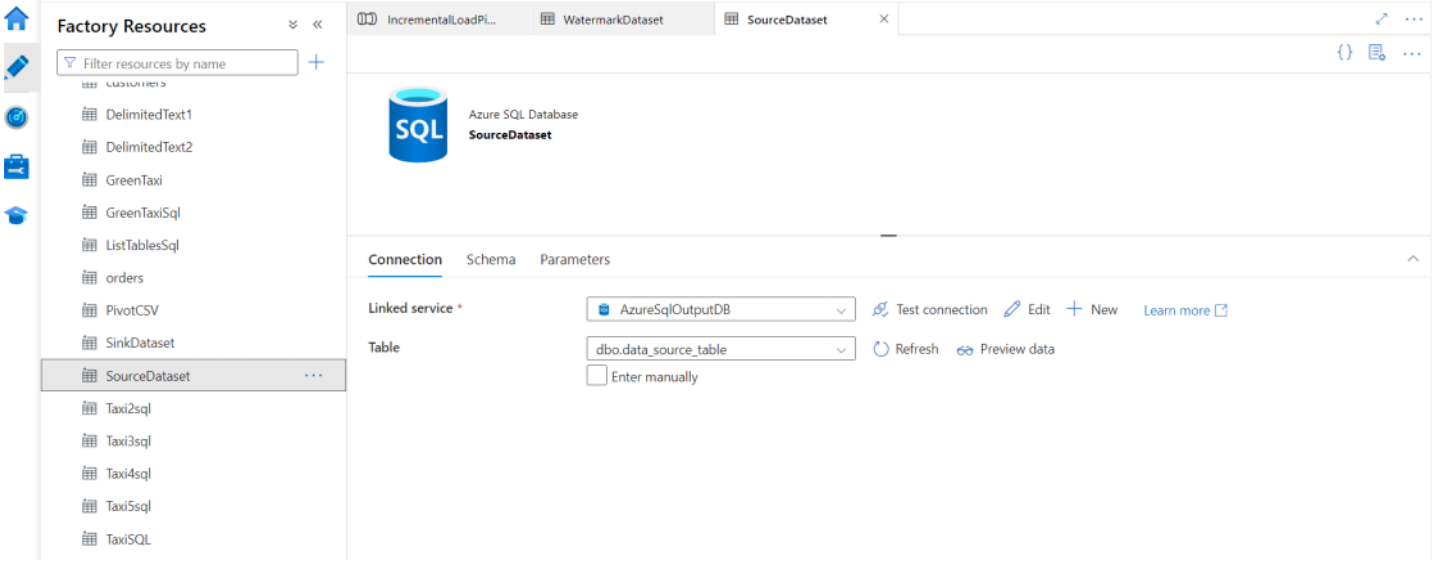


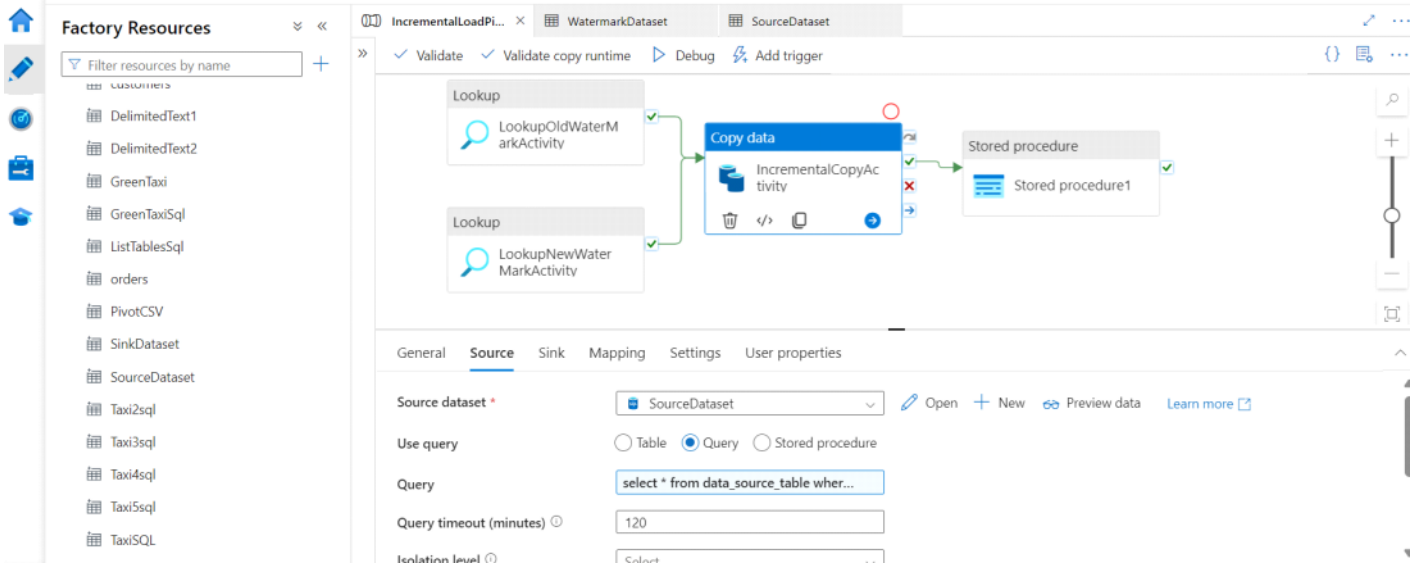


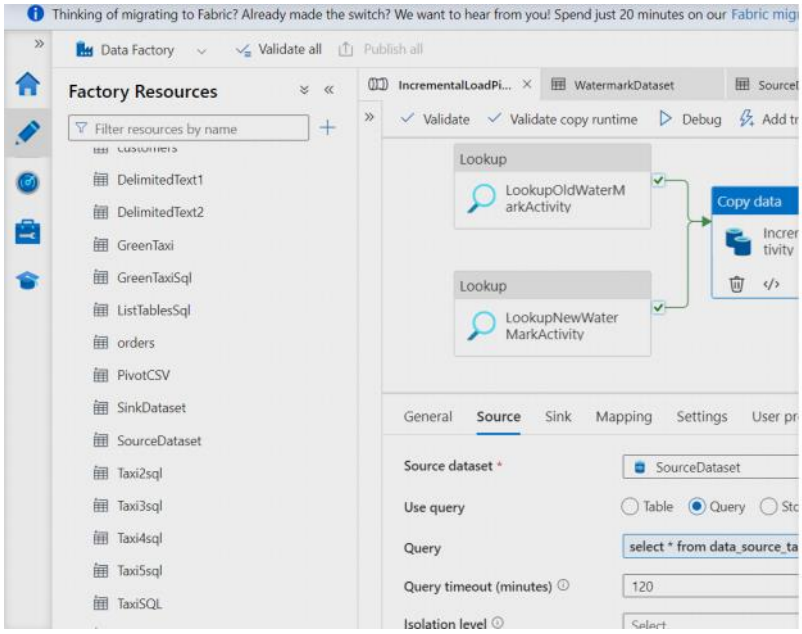


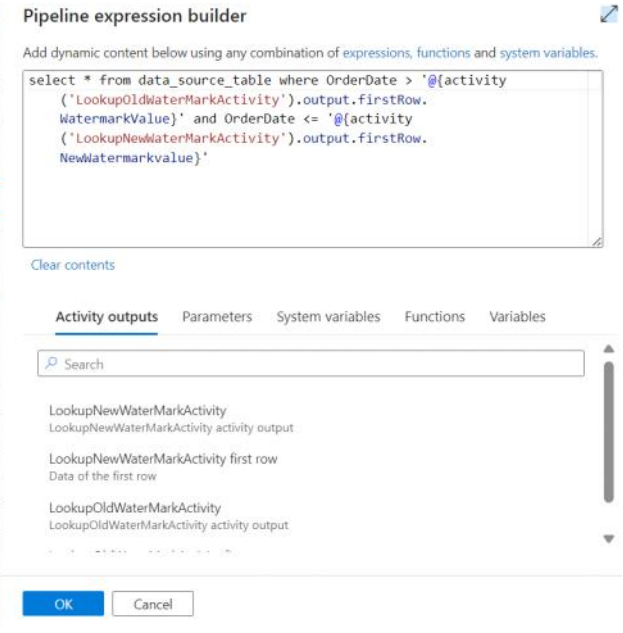

Data Factory
Validate all
Publish all
Preview experience
Off
IncrementalLoadPi...
WatermarkDataset
Azure SQL Database
WatermarkDataset
Connection
Schema
Parameters
Expand resources pane
Import schema
Clear
Column name
Type
TableName
varchar
WatermarkValue
datetime

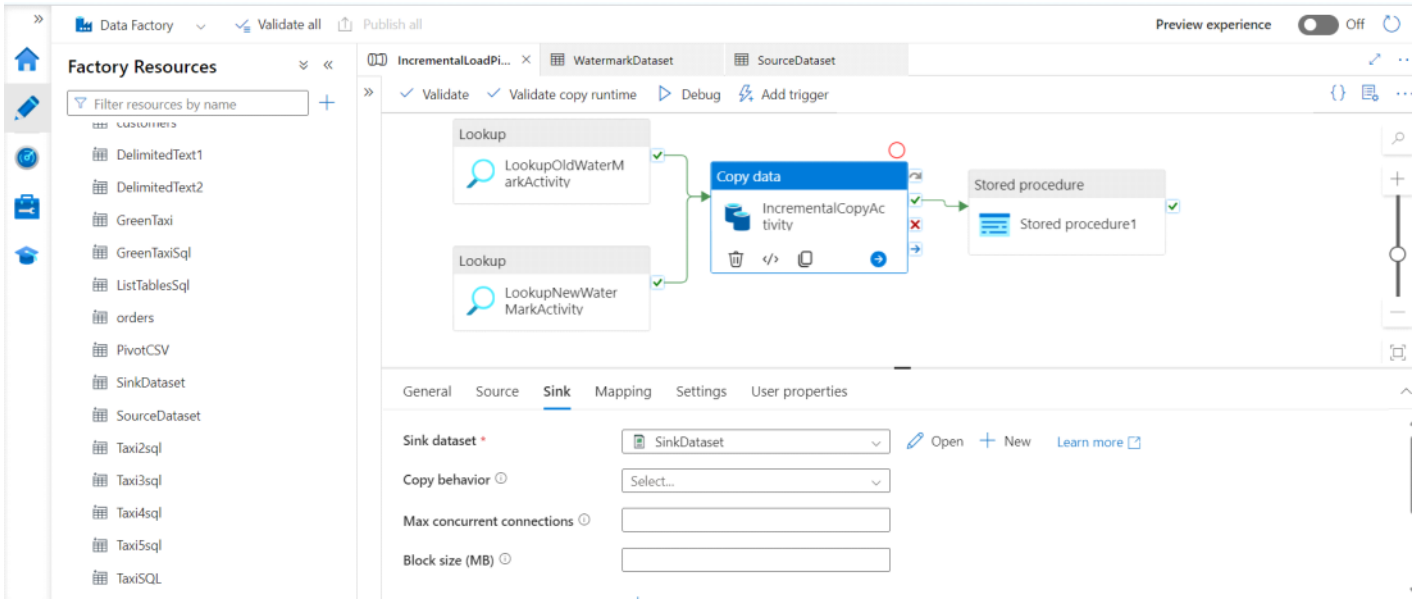

Data Factory
Validate all
Publish all
Preview experience
Off
IncrementalLoadPi...
WatermarkDataset
Validate
Debug
Add trigger
Lookup
LookupOldWaterMarkActivity
Copy data
IncrementalCopyActivity
Stored procedure
Stored procedure1
Lookup
LookupNewWaterMarkActivity
General
Settings
User properties
Source dataset *
SourceDataset
Open
New
Preview data
Learn more
First row only
Use query
Table
Query
Stored procedure
Query *
select MAX(OrderDate) as NewWatermarkvalue from data_source_table
Edit

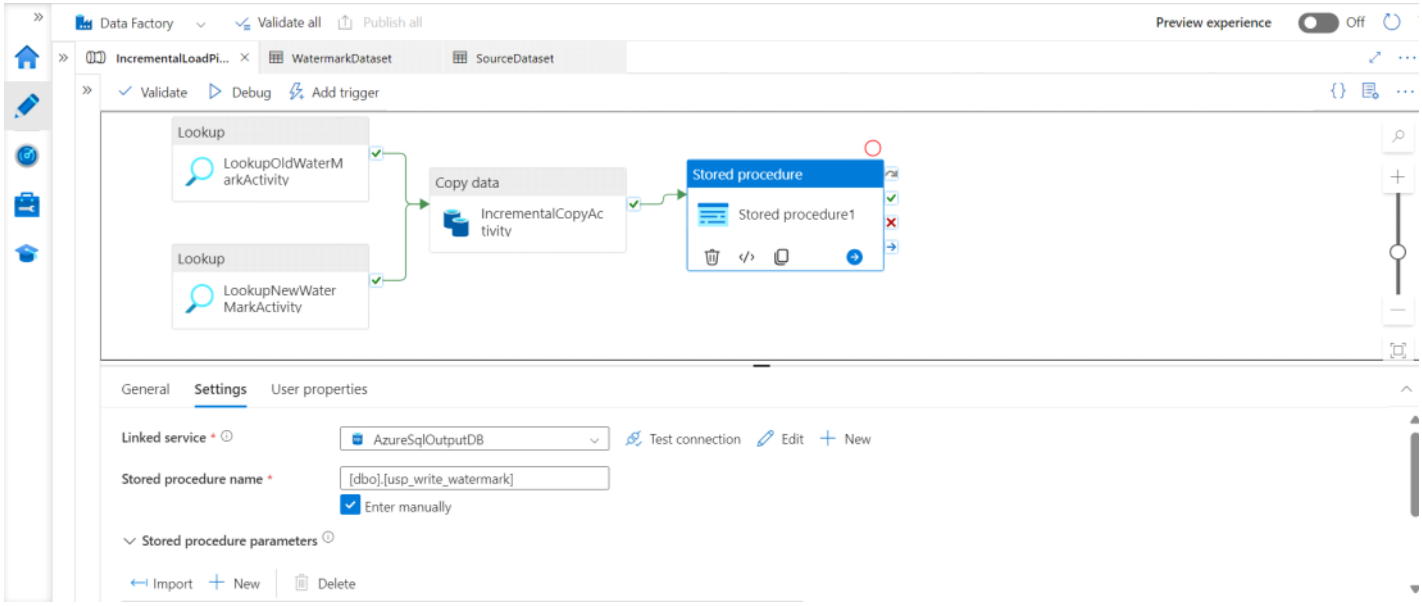

Factory Resources
Filter resources by name
DelimitedText1
DelimitedText2
GreenTaxi
GreenTaxiSql
ListTablesSql
orders
PivotCSV
SinkDataset
SourceDataset
Taxi2sql
Taxi3sql
Taxi4sql
Taxi5sql
TaxiSQL
IncrementalLoadPi...
WatermarkDataset
SourceDataset
Azure SQL Database
SourceDataset
Connection
Schema
Parameters
Linked service *
AzureSqlOutputDB
Test connection
Edit
New
Learn more
Table
dbo.data_source_table
Refresh
Preview data
Enter manually

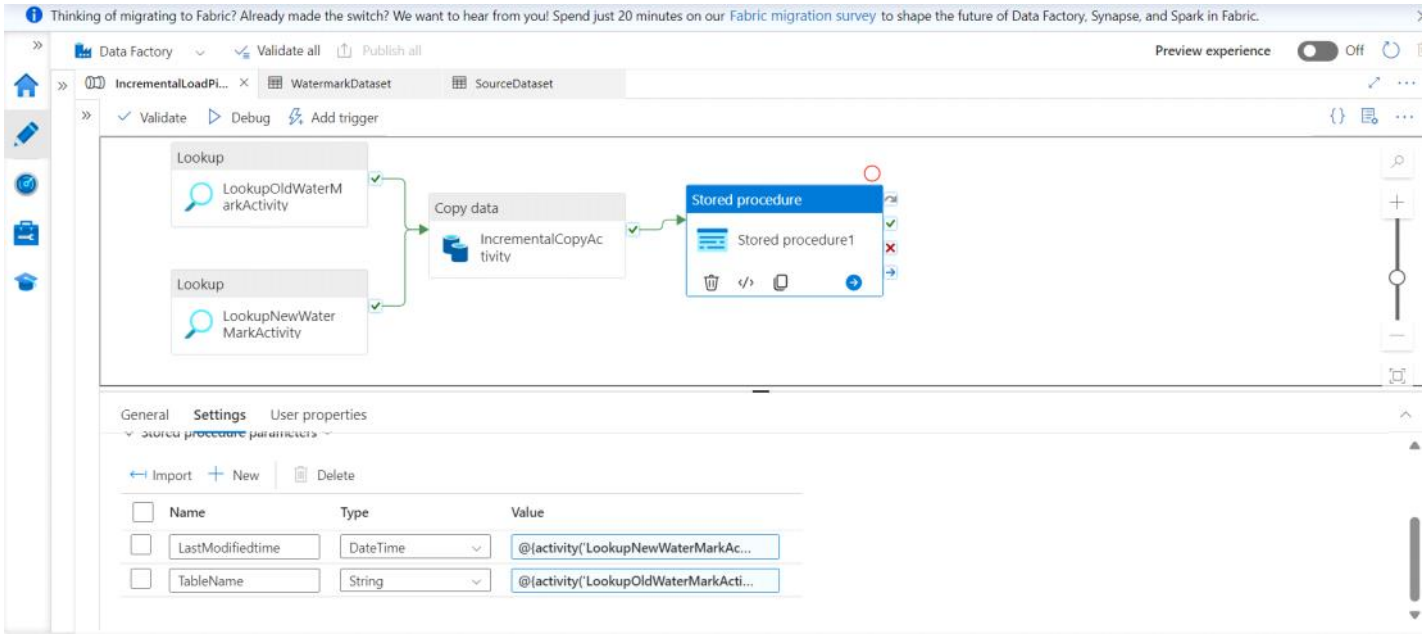

Factory Resources
Filter resources by name
DelimitedText1
DelimitedText2
GreenTaxi
GreenTaxiSql
ListTablesSql
orders
PivotCSV
SinkDataset
SourceDataset
Taxi2sql
Taxi3sql
Taxi4sql
Taxi5sql
TaxiSQL
IncrementalLoadPi...
WatermarkDataset
SourceDataset
Validate
Validate copy runtime
Debug
Add trigger
Lookup
LookupOldWaterMarkActivity
Lookup
LookupNewWaterMarkActivity
Copy data
IncrementalCopyActivity
Stored procedure
Stored procedure1
General
Source
Sink
Mapping
Settings
User properties
Source dataset *
SourceDataset
Open
New
Preview data
Learn more
Use query
Table
Query
Stored procedure
Query
select * from data_source_table wher...
Query timeout (minutes)
120
Isolation level

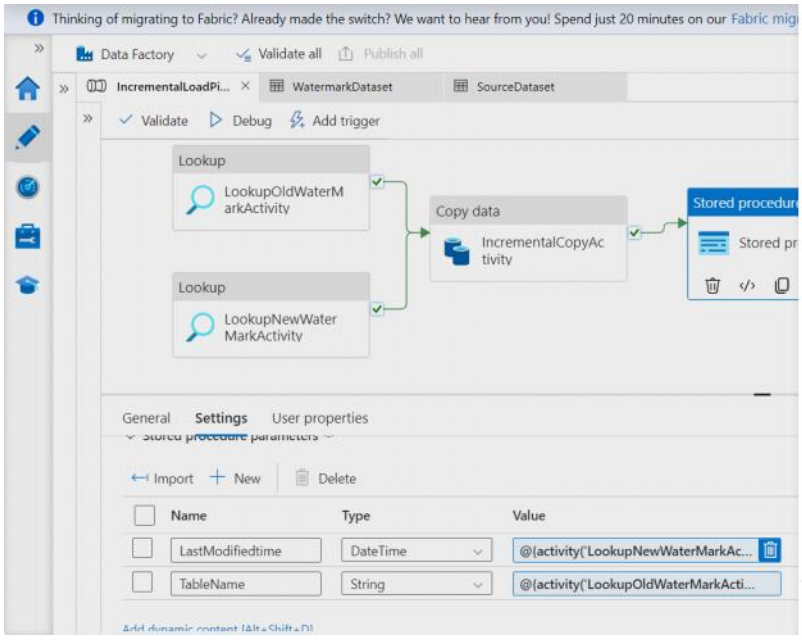

Thinking of migrating to Fabric? Already made the switch? We want to hear from you! Spend just 20 minutes on our Fabric mig
Data Factory
Validate all
Publish all
Factory Resources
Filter resources by name
DelimitedText1
DelimitedText2
GreenTaxi
GreenTaxiSql
ListTablesSql
orders
PivotCSV
SinkDataset
SourceDataset
Taxi2sql
Taxi3sql
Taxi4sql
Taxi5sql
TaxiSQL
IncrementalLoadPi...
WatermarkDataset
Validate
Validate copy runtime
Debug
Lookup
LookupOldWaterMarkActivity
Lookup
LookupNewWaterMarkActivity
Copy data
General
Source
Sink
Mapping
Settings
Source dataset *
SourceDataset
Use query
Table
Query
Query
Query timeout (minutes)
120
Isolation level

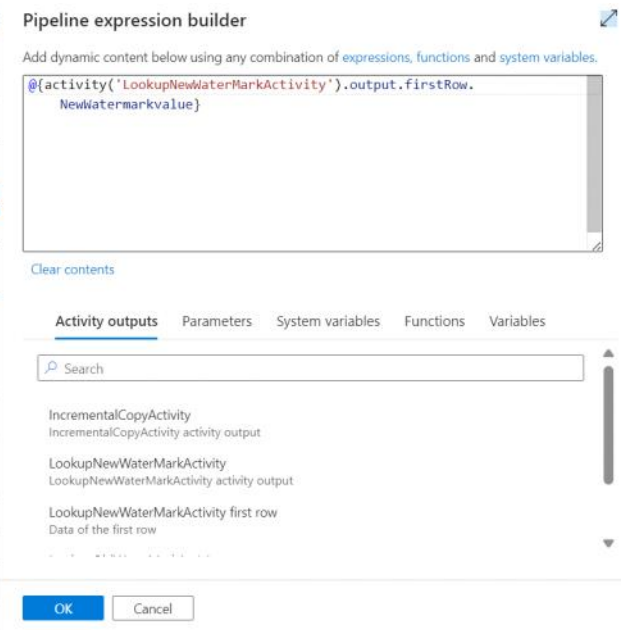

Pipeline expression builder
Add dynamic content below using any combination of expressions, functions and system variables.
select * from data_source_table where OrderDate > '@{activity
('LookupOldWaterMarkActivity').output.firstRow.
WatermarkValue}' and OrderDate <= '@{activity
('LookupNewWaterMarkActivity').output.firstRow.
NewWatermarkvalue}'
Clear contents
Activity outputs
Parameters
System variables
Functions
Variables
Search
LookupNewWaterMarkActivity
LookupNewWaterMarkActivity activity output
LookupNewWaterMarkActivity first row
Data of the first row
LookupOldWaterMarkActivity
LookupOldWaterMarkActivity activity output
OK
Cancel

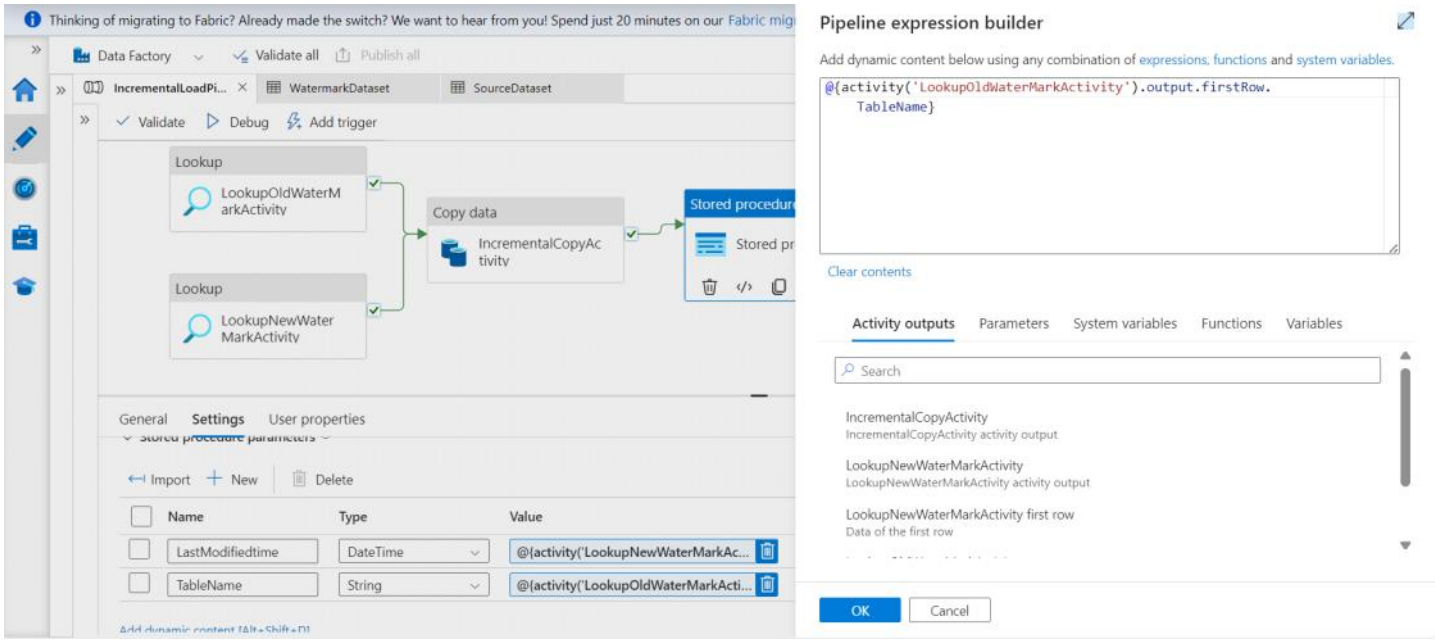

Data Factory
Validate all
Publish all
Preview experience
Off
Factory Resources
Filter resources by name
DelimitedText1
DelimitedText2
GreenTaxi
GreenTaxiSql
ListTablesSql
orders
PivotCSV
SinkDataset
SourceDataset
Taxi2sql
Taxi3sql
Taxi4sql
Taxi5sql
TaxiSQL
IncrementalLoadPi...
WatermarkDataset
SourceDataset
Validate
Validate copy runtime
Debug
Add trigger
Lookup
LookupOldWaterMarkActivity
Lookup
LookupNewWaterMarkActivity
Copy data
IncrementalCopyActivity
Stored procedure
Stored procedure1
General
Source
Sink
Mapping
Settings
User properties
Sink dataset *
SinkDataset
Open
New
Learn more
Copy behavior
Select...
Max concurrent connections
Block size (MB)

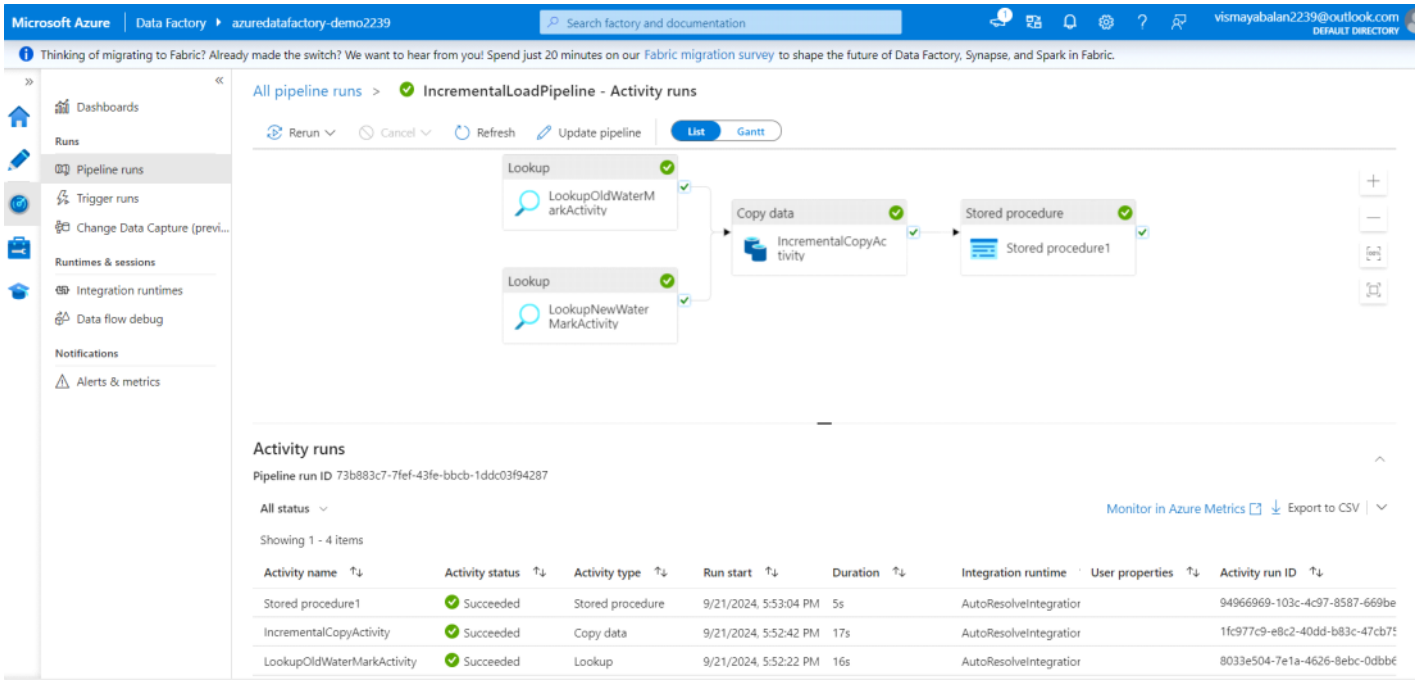

Data Factory
Validate all
Publish all
Preview experience
Off
IncrementalLoadPi...
WatermarkDataset
SourceDataset
Validate
Debug
Add trigger
Lookup
LookupOldWaterMarkActivity
Lookup
LookupNewWaterMarkActivity
Copy data
IncrementalCopyActivity
Stored procedure
Stored procedure1
General
Settings
User properties
Linked service *
AzureSqlOutputDB
Test connection
Edit
New
Stored procedure name *
[dbo].[usp_write_watermark]
Enter manually
Stored procedure parameters
Import
New
Delete


Thinking of migrating to Fabric? Already made the switch? We want to hear from you! Spend just 20 minutes on our Fabric migration survey to shape the future of Data Factory, Synapse, and Spark in Fabric.
Data Factory
Validate all
Publish all
Preview experience
Off
IncrementalLoadPi...
WatermarkDataset
SourceDataset
Validate
Debug
Add trigger
Lookup
LookupOldWaterMarkActivity
Lookup
LookupNewWaterMarkActivity
Copy data
IncrementalCopyActivity
Stored procedure
Stored procedure1
General
Settings
User properties
Import
New
Delete
Name
Type
Value
LastModifiedtime
DateTime
@{activity('LookupNewWaterMarkAc...
TableName
String
@{activity('LookupOldWaterMarkActi...


Thinking of migrating to Fabric? Already made the switch? We want to hear from you! Spend just 20 minutes on our Fabric mig
Data Factory
Validate all
Publish all
IncrementalLoadPi...
WatermarkDataset
SourceDataset
Validate
Debug
Add trigger
Lookup
LookupOldWaterMarkActivity
Lookup
LookupNewWaterMarkActivity
Copy data
IncrementalCopyActivity
Stored procedur
Stored pr
General
Settings
User properties
Import
New
Delete
Name
Type
Value
LastModifiedtime
DateTime
@{activity('LookupNewWaterMarkAc...
TableName
String
@{activity('LookupOldWaterMarkActi...
Add dynamic content [Alt+Shift+D]

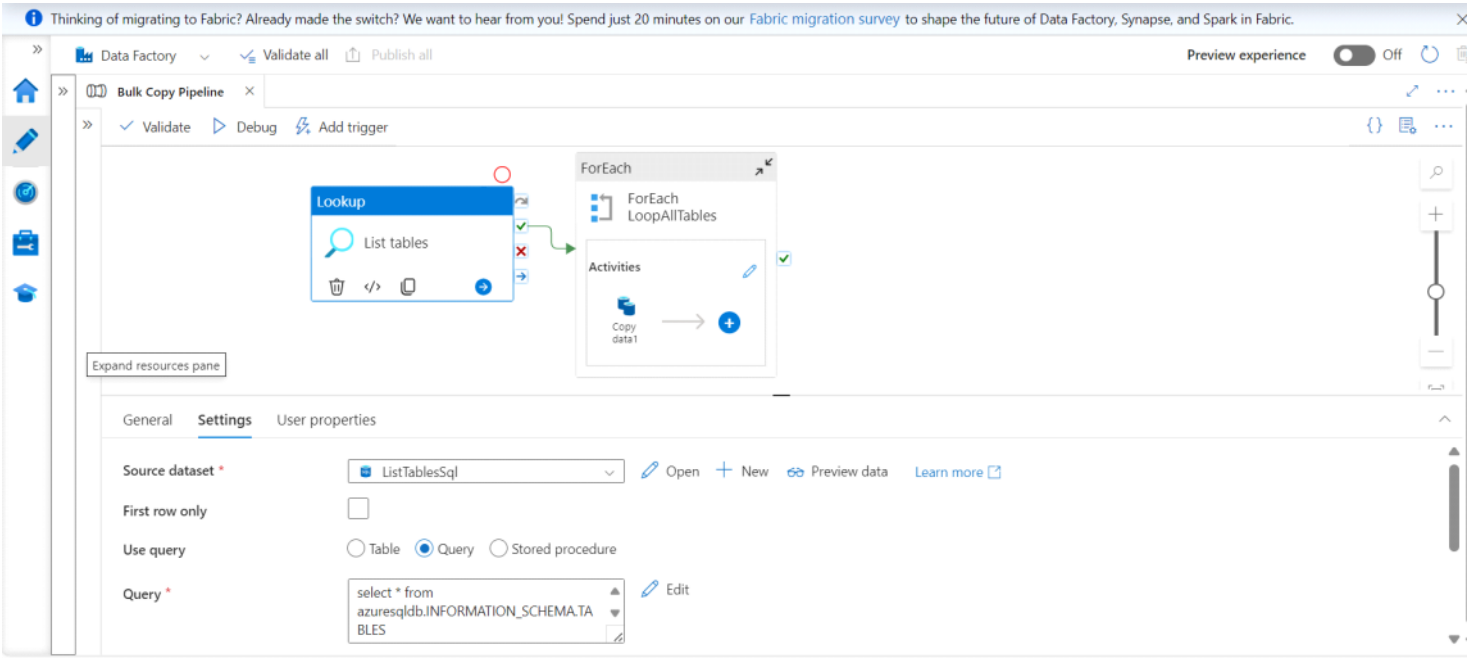

Pipeline expression builder
Add dynamic content below using any combination of expressions, functions and system variables.
@{activity('LookupNewWaterMarkActivity').output.firstRow.
NewWatermarkvalue}
Clear contents
Activity outputs
Parameters
System variables
Functions
Variables
Search
IncrementalCopyActivity
IncrementalCopyActivity activity output
LookupNewWaterMarkActivity
LookupNewWaterMarkActivity activity output
LookupNewWaterMarkActivity first row
Data of the first row
OK
Cancel

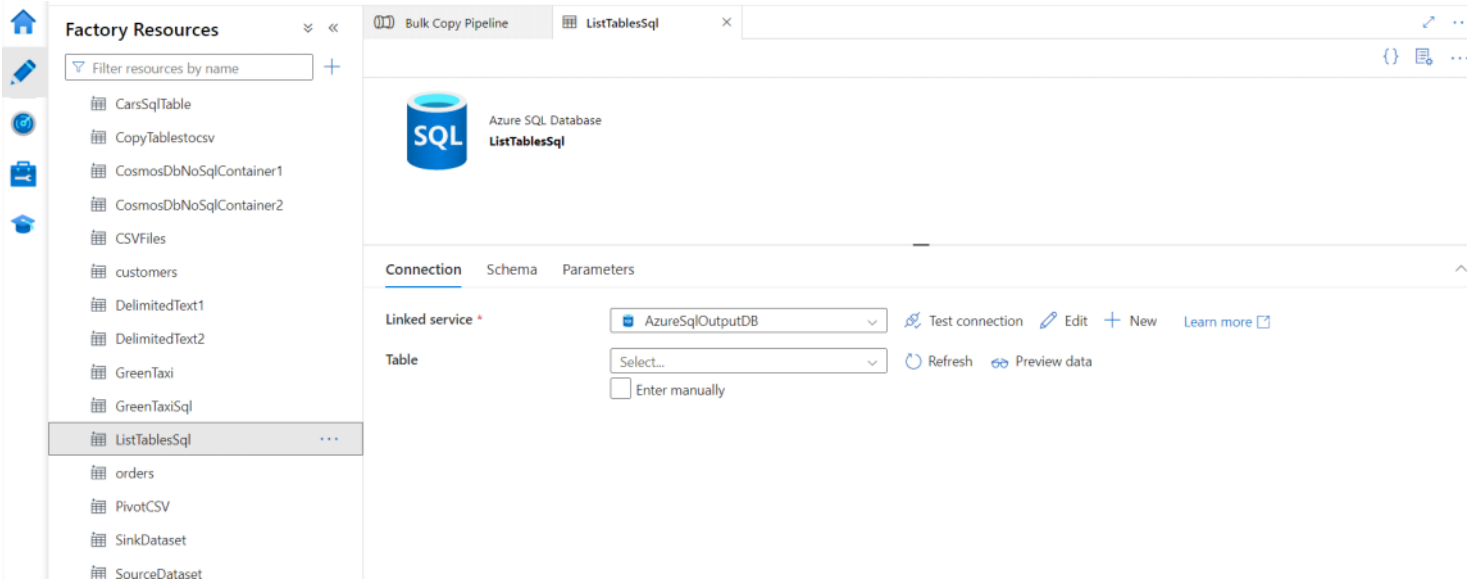

Thinking of migrating to Fabric? Already made the switch? We want to hear from you! Spend just 20 minutes on our Fabric mig
Data Factory
Validate all
Publish all
IncrementalLoadPi...
WatermarkDataset
SourceDataset
Validate
Debug
Add trigger
Lookup
LookupOldWaterMarkActivity
Copy data
IncrementalCopyActivity
Stored procedur
Stored pr
Lookup
LookupNewWaterMarkActivity
General
Settings
User properties
Import
New
Delete
Name
Type
Value
LastModifiedtime
DateTime
@{activity('LookupNewWaterMarkAc...
TableName
String
@{activity('LookupOldWaterMarkActi...
Pipeline expression builder
Add dynamic content below using any combination of expressions, functions and system variables.
@{activity('LookupOldWaterMarkActivity').output.firstRow.
TableName}
Clear contents
Activity outputs
Parameters
System variables
Functions
Variables
Search
IncrementalCopyActivity
IncrementalCopyActivity activity output
LookupNewWaterMarkActivity
LookupNewWaterMarkActivity activity output
LookupNewWaterMarkActivity first row
Data of the first row
OK
Cancel

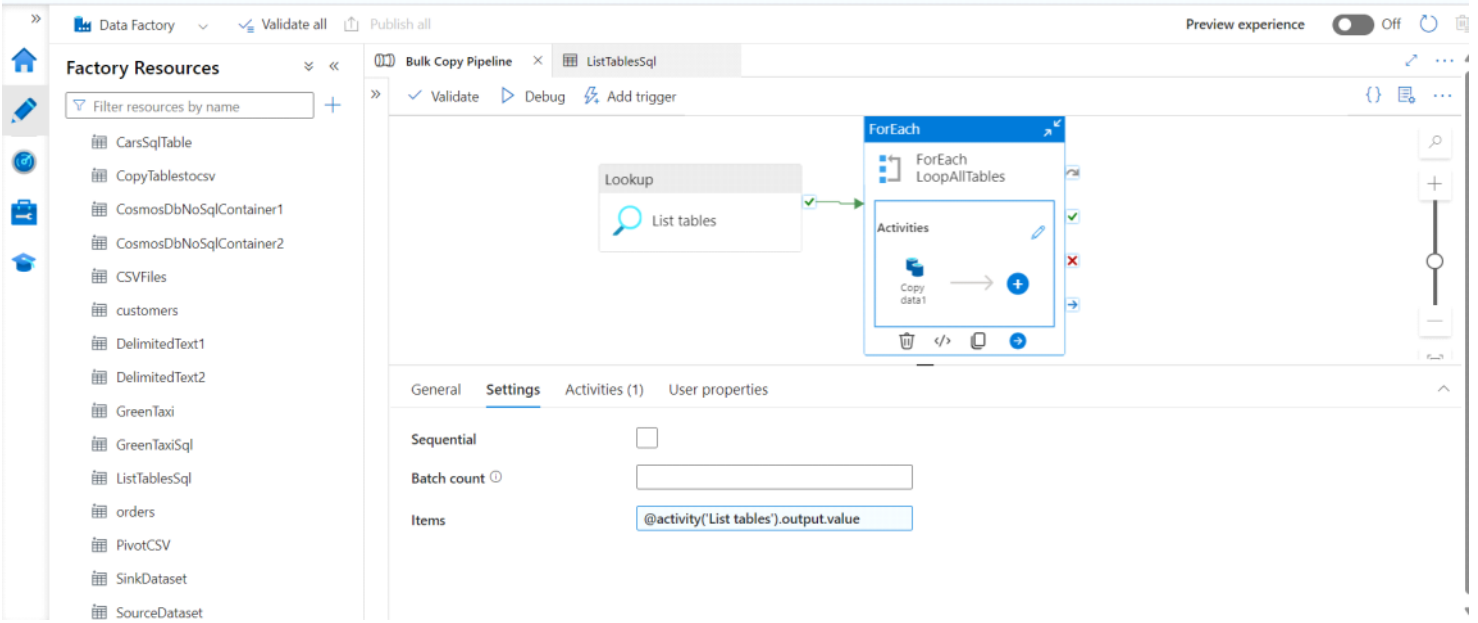

Microsoft Azure
Data Factory
azuredatafactory-demo2239
Search factory and documentation
vismayabalan2239@outlook.com
DEFAULT DIRECTORY
Thinking of migrating to Fabric? Already made the switch? We want to hear from you! Spend just 20 minutes on our Fabric migration survey to shape the future of Data Factory, Synapse, and Spark in Fabric.
Dashboards
Runs
Pipeline runs
Trigger runs
Change Data Capture (previ...
Runtimes & sessions
Integration runtimes
Data flow debug
Notifications
Alerts & metrics
All pipeline runs
IncrementalLoadPipeline - Activity runs
Rerun
Cancel
Refresh
Update pipeline
List
Gantt
Lookup
LookupOldWaterMarkActivity
Copy data
IncrementalCopyActivity
Stored procedure
Stored procedure1
Lookup
LookupNewWaterMarkActivity
Activity runs
Pipeline run ID 73b883c7-7fef-43fe-bbcb-1ddc03f94287
All status
Monitor in Azure Metrics
Export to CSV
Showing 1 - 4 items
Activity name
Activity status
Activity type
Run start
Duration
Integration runtime
User properties
Activity run ID
Stored procedure1
Succeeded
Stored procedure
9/21/2024, 5:53:04 PM
5s
AutoResolveIntegratior
94966969-103c-4c97-8587-669be
IncrementalCopyActivity
Succeeded
Copy data
9/21/2024, 5:52:42 PM
17s
AutoResolveIntegratior
1fc977c9-e8c2-40dd-b83c-47cb7
LookupOldWaterMarkActivity
Succeeded
Lookup
9/21/2024, 5:52:22 PM
16s
AutoResolveIntegratior
8033e504-7e1a-4626-8ebc-0dbb

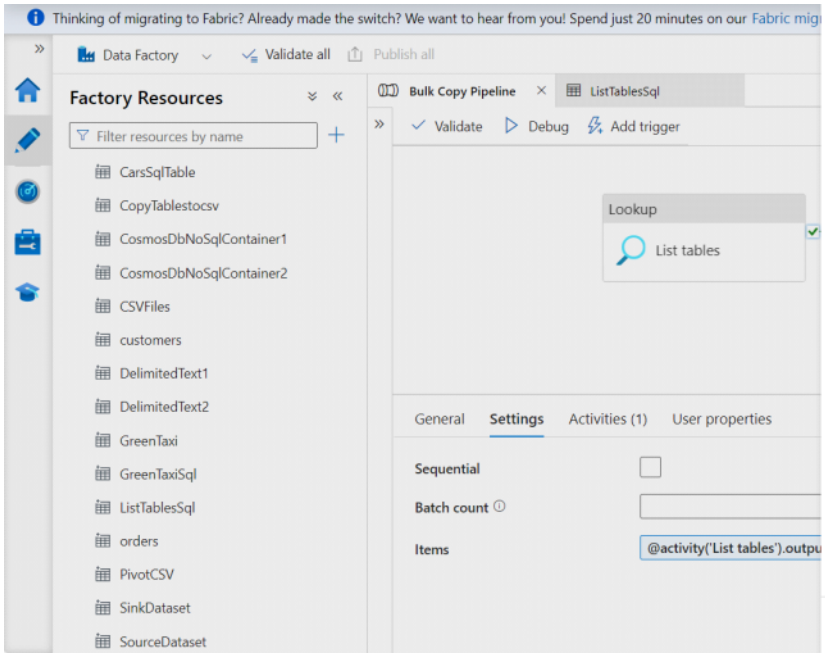
Bulk Copy Pipeline

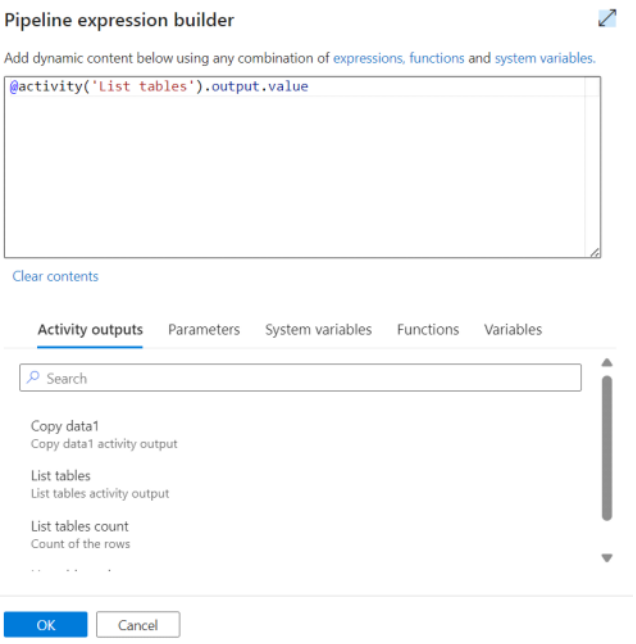
27 September 2024 10:43

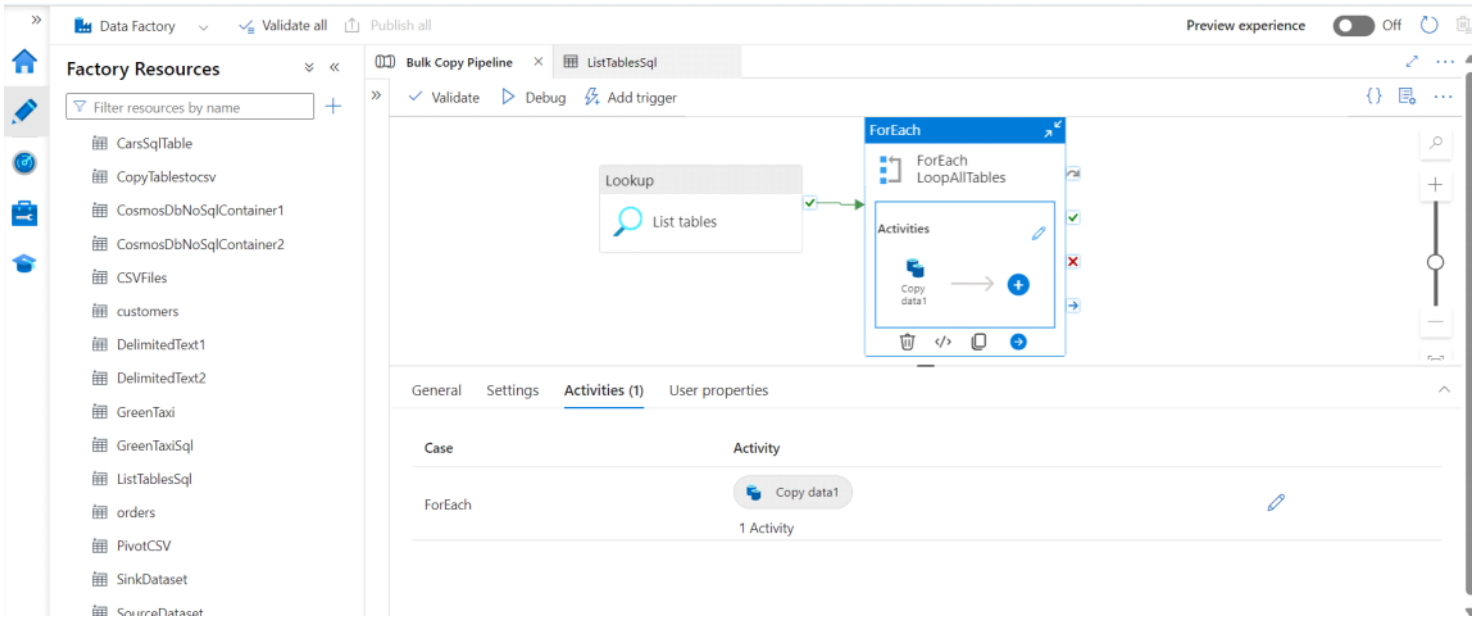


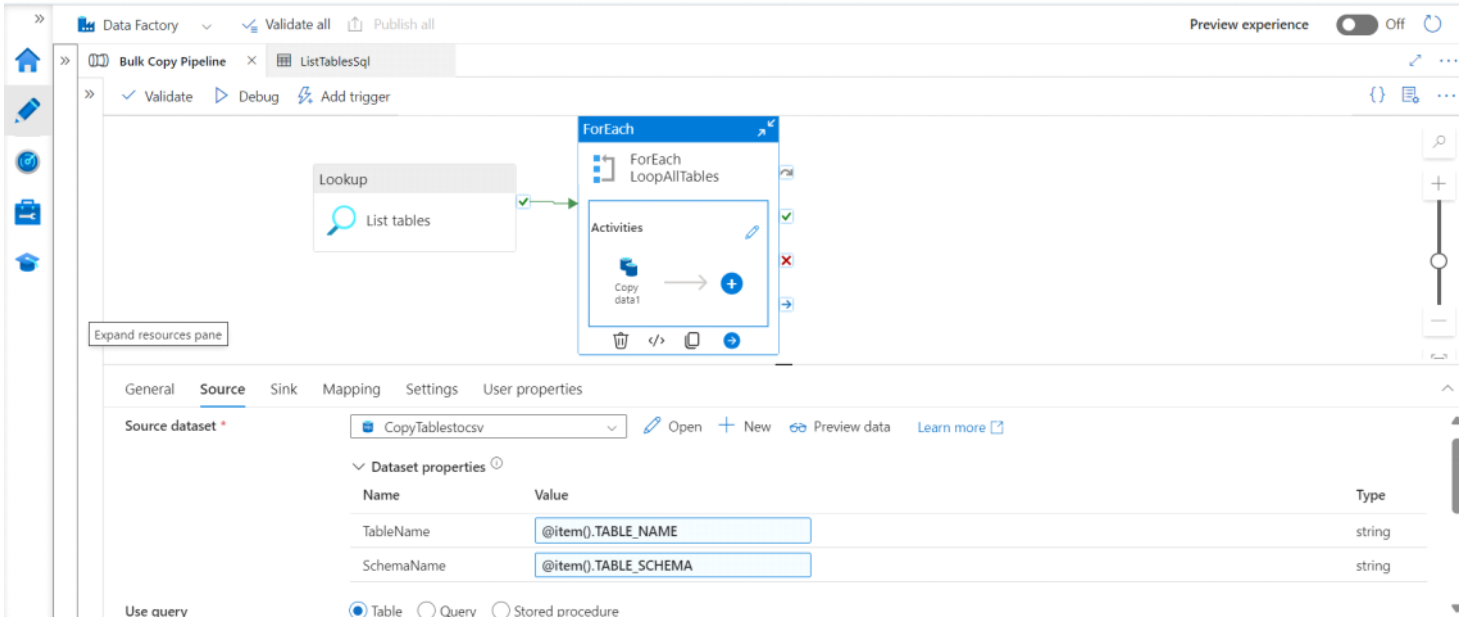


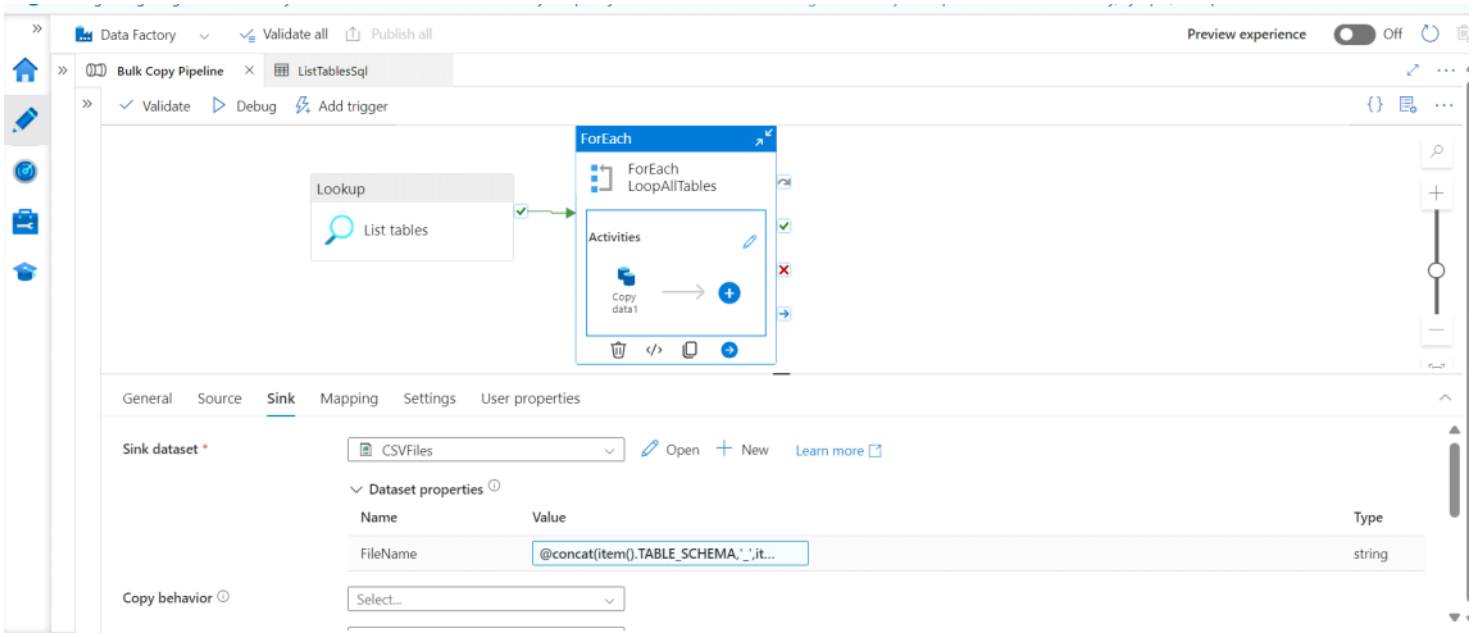


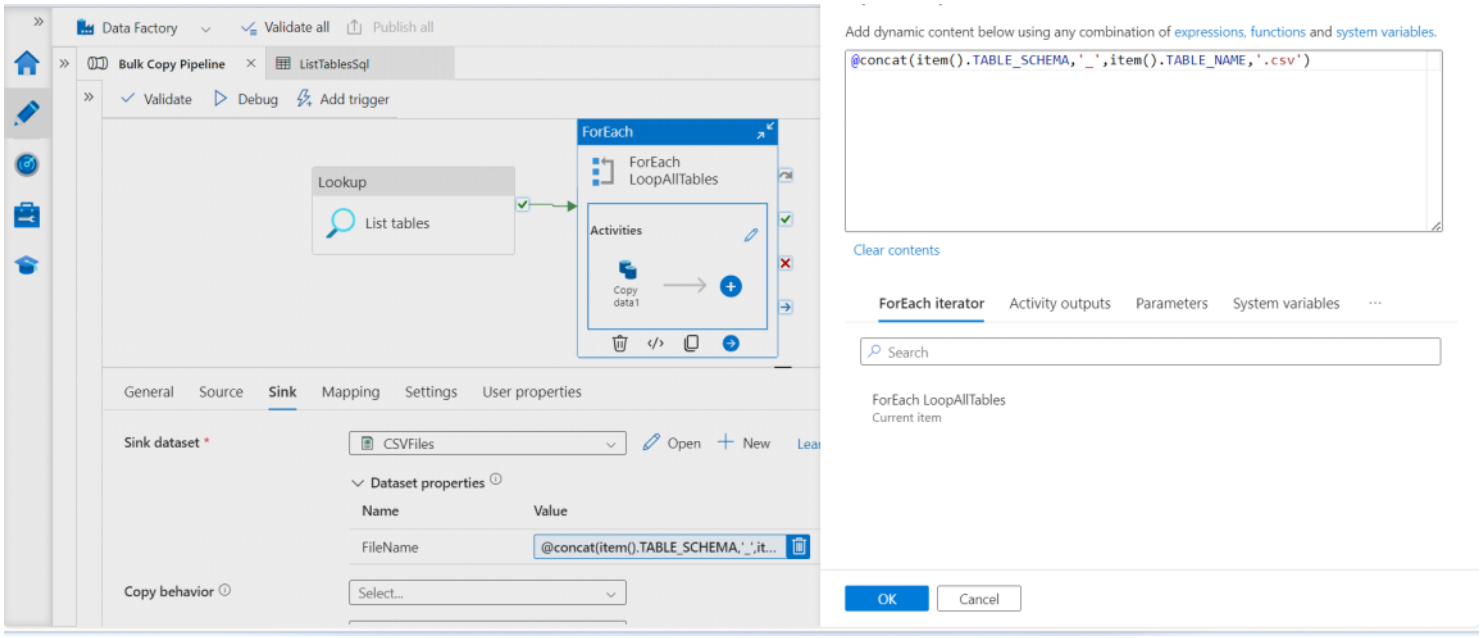

Thinking of migrating to Fabric? Already made the switch? We want to hear from you! Spend just 20 minutes on our Fabric mig
Data Factory
Validate all
Publish all
Factory Resources
Filter resources by name
CarsSqlTable
CopyTablestocsv
CosmosDbNoSqlContainer1
CosmosDbNoSqlContainer2
CSVFiles
customers
DelimitedText1
DelimitedText2
GreenTaxi
GreenTaxiSql
ListTablesSql
orders
PivotCSV
SinkDataset
SourceDataset
Bulk Copy Pipeline
ListTablesSql
Validate
Debug
Add trigger
Lookup
List tables
General
Settings
Activities (1)
User properties
Sequential
Batch count
Items
@activity('List tables').outpu

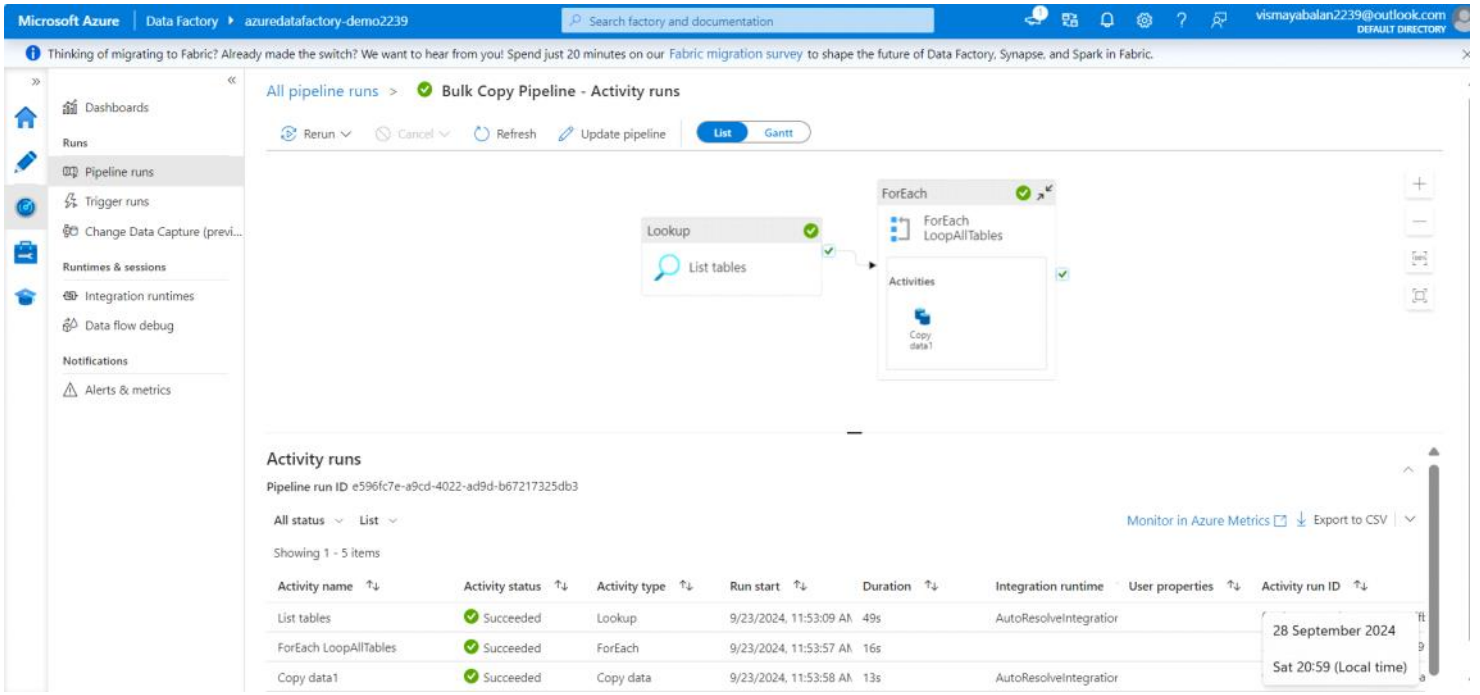

Pipeline expression builder
Add dynamic content below using any combination of expressions, functions and system variables.
@activity('List tables').output.value
Clear contents
Activity outputs
Parameters
System variables
Functions
Variables
Search
Copy data1
Copy data1 activity output
List tables
List tables activity output
List tables count
Count of the rows
OK
Cancel


Data Factory
Validate all
Publish all
Preview experience
Off
Factory Resources
Filter resources by name
CarsSqlTable
CopyTablestocsv
CosmosDbNoSqlContainer1
CosmosDbNoSqlContainer2
CSVFiles
customers
DelimitedText1
DelimitedText2
GreenTaxi
GreenTaxiSql
ListTablesSql
orders
PivotCSV
SinkDataset
Bulk Copy Pipeline
ListTablesSql
Validate
Debug
Add trigger
Lookup
List tables
ForEach
ForEach
LoopAllTables
Activities
Copy data1
General
Settings
Activities (1)
User properties
Case
Activity
ForEach
Copy data1
1 Activity


Data Factory
Validate all
Publish all
Preview experience
Off
Bulk Copy Pipeline
ListTablesSql
Validate
Debug
Add trigger
Lookup
List tables
ForEach
ForEach
LoopAllTables
Activities
Copy data1
Expand resources pane
General
Source
Sink
Mapping
Settings
User properties
Source dataset *
CopyTablestocsv
Open
New
Preview data
Learn more
Dataset properties
Name
Value
Type
TableName
@item().TABLE_NAME
string
SchemaName
@item().TABLE_SCHEMA
string
Use query
Table
Query
Stored procedure

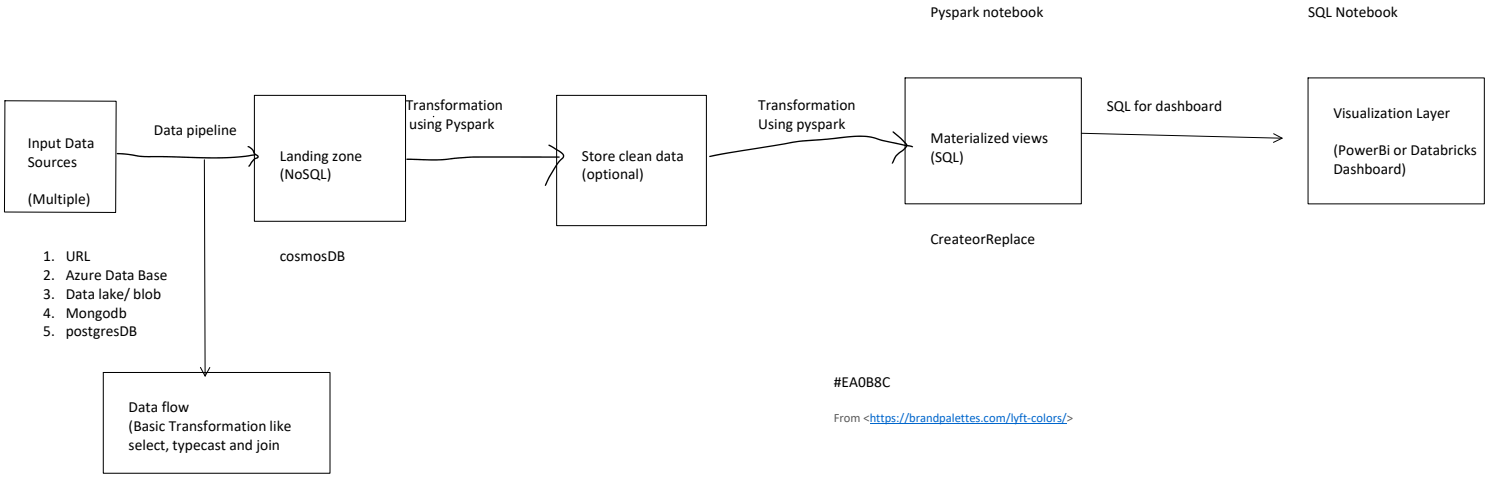

Data Factory
Validate all
Publish all
Preview experience
Off
Bulk Copy Pipeline
ListTablesSql
Validate
Debug
Add trigger
Lookup
List tables
ForEach
ForEach LoopAllTables
Activities
Copy data1
General
Source
Sink
Mapping
Settings
User properties
Sink dataset *
CSVFiles
Open
New
Learn more
Dataset properties
Name
Value
Type
FileName
@concat(item().TABLE_SCHEMA,'_',it...
string
Copy behavior
Select...


Data Factory
Validate all
Publish all
Bulk Copy Pipeline
ListTablesSql
Validate
Debug
Add trigger
Lookup
List tables
ForEach
ForEach LoopAllTables
Activities
Copy data1
General
Source
Sink
Mapping
Settings
User properties
Sink dataset *
CSVFiles
Open
New
Dataset properties
Name
Value
FileName
@concat(item().TABLE_SCHEMA,'_',it...
Copy behavior
Select...
Add dynamic content below using any combination of expressions, functions and system variables.
@concat(item().TABLE_SCHEMA,'_',item().TABLE_NAME,'.csv')
Clear contents
ForEach iterator
Activity outputs
Parameters
System variables
Search
ForEach LoopAllTables
Current item
OK
Cancel


Microsoft Azure
Data Factory
azuredatafactory-demo2239
Search factory and documentation
vismayabalan2239@outlook.com
DEFAULT DIRECTORY
Thinking of migrating to Fabric? Already made the switch? We want to hear from you! Spend just 20 minutes on our Fabric migration survey to shape the future of Data Factory, Synapse, and Spark in Fabric.
Dashboards
Runs
Pipeline runs
Trigger runs
Change Data Capture (previ...
Runtimes & sessions
Integration runtimes
Data flow debug
Notifications
Alerts & metrics
All pipeline runs > Bulk Copy Pipeline - Activity runs
Rerun
Cancel
Refresh
Update pipeline
List
Gantt
Lookup
List tables
ForEach
ForEach LoopAllTables
Activities
Copy data1
Activity runs
Pipeline run ID e596fc7e-a9cd-4022-ad9d-b67217325db3
All status
List
Monitor in Azure Metrics
Export to CSV
Showing 1 - 5 items
Activity name
Activity status
Activity type
Run start
Duration
Integration runtime
User properties
Activity run ID
List tables
Succeeded
Lookup
9/23/2024, 11:53:09 AM
49s
AutoResolveIntegration
ForEach LoopAllTables
Succeeded
ForEach
9/23/2024, 11:53:57 AM
16s
Copy data1
Succeeded
Copy data
9/23/2024, 11:53:58 AM
13s
AutoResolveIntegration
28 September 2024
Sat 20:59 (Local time)

Capstone

27 September 2024 09:36



#EA0B8C

From <https://brandpalettes.com/lyft-colors/>

We can use any sources except cosmos
Minimum 3 sources have to be picked url, datalake or blob , azure database
For some of them we have to use dataflow for joining the dataset from different sources

Corr
Quartiles
Ntile
Window functions
Lag lead

https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/2460785922056449/3258091060545073/6795197445702991/latest.html

Decline
corr

+----------+--------+-------------+--------+-----+--------------------+------+--------+-----+ |time_stamp|pressure|location| rain| wind| id|clouds|humidity| temp|
+----------+--------+-------------+--------+-----+--------------------+------+--------+-----+ |1543798374| 1002.54|North Station| null| 2.61|b2393cff-e890-471...| 1| 0.93|48.69| |1543462329| 998.65|North Station| null|11.26|dbcdccf9-2910-424...| 0.42| 0.72| 37.5| |1543268108| 1009.04|North Station|3.00E-04| 9.41|8822cff1-b98d-4c2...| 0.92| 0.86|44.48| |1543408327| 991.29|North Station| null| 7.72 |baee2fb3-4d84-42a...| 0.73| 0.72|36.05| |1543765974| 1012.22|North Station| 0.003| 6.33 |f4f851ac-a954-445...| 1| 0.95|44.95| |1543209464| 1013.78|North Station| null| 2.96|f76755aa-bba1-492...| 0.93| 0.92|40.58| my all values in the rain got converted to 0 instead of null values
+----------+--------+-------------+--------+-----+--------------------+------+--------+-----+ |time_stamp|pressure|location|rain|wind| id|clouds|humidity| temp|
+----------+--------+-------------+--------+-----+--------------------+------+--------+-----+ |1543203644| 1014.35|Northeastern Univ...| 0|1.36|d2e446ae-6b46-42b...| 0.89| 0.93|40.81| |1543203644| 1014.39| South Station| 0| 1.6|2c35d436-9c68-4d9...| 0.87| 0.93|40.86| |1543203644| 1014.4| Haymarket Square| 0 |1.57|9ba7ea28-f8ea-42d...| 0.87| 0.92|40.98| |1543203644| 1014.4| West End| 0|1.52|d0372e03-c94c-498...| 0.87| 0.93|40.84| |1543203645| 1014.35| Financial District| 0|1.49|de11dbc3-ea90-426... | 0.89| 0.93|40.95| |1543203645| 1014.35| North Station| 0|1.43|6d7cf3a6-f741-4fb...| 0.89| 0.92| 40.94| |1543203645| 1014.39| Back Bay| 0|1.46|24dc0aa7-00f8-4a7...| 0.87| 0.92|41.04|

```
from pyspark.sql.functions import col

# Cast the 'rain' column to float type (if it's not already)
df = df.withColumn("rain", col("rain").cast("float"))

# Replace only null values in the 'rain' column with 0
df = df.fillna({'rain': 0})

# Verify the result
df.show()
```

while casting it into float it is becoming 0 . do i have to use double or something?

Time Period	Uber Color (Shades of Gray)	Lyft Color (Shades of Pink/Purple)
Morning	#D3D3D3 (Light Gray)	#FFB6C1 (Light Pink)
Afternoon	#A9A9A9 (Medium Gray)	#FF69B4 (Soft Pink)
Evening	#696969 (Dark Gray)	#C71585 (Medium Purple)
Night	#000000 (Black)	#800080 (Dark Purple)

It looks like you have a well-structured dataset for analyzing Uber/Lyft cab prices and the impact of weather conditions on surcharges! Here are some analytical insights and visualizations you might consider implementing in Databricks and Power BI:

Suggested Analytical Approaches

1. **Price Trends Analysis**:
   - Analyze how cab prices fluctuate over time. You can group the data by date or time period and plot average prices, highlighting peak times.

2. **Surge Pricing Analysis**:
   - Investigate surge multiplier trends relative to different weather conditions. Create visualizations showing average surge multipliers on rainy vs. non-rainy days.

3. **Correlation Analysis**:
   - Use correlation metrics to assess the relationship between cab prices and weather variables (e.g., temperature, humidity, wind speed). A heatmap could visualize these correlations effectively.

4. **Price Per Mile Analysis**:
   - Calculate and visualize the average price per mile for different cab types. This can help identify which services offer better value.

5. **Weather Impact Analysis**:
   - Compare average prices and surge multipliers for different weather conditions (sunny, rainy, windy) to understand how weather affects pricing strategies.

6. **Time of Day Analysis**:
   - Segment the data by time of day (Morning, Afternoon, Evening) to analyze how prices and surge multipliers vary throughout the day.

7. **Destination-Based Analysis**:
   - Examine pricing patterns for specific destinations (e.g., Theatre District, North Station) to identify high-demand areas.

Suggested Visualizations in Power BI

- **Line Charts**: To visualize trends over time for prices and surge multipliers.
- **Bar Charts**: For comparing average prices by cab type, weather condition, or time of day.
- **Heatmaps**: To show correlations between price and weather variables.
- **Scatter Plots**: To illustrate the relationship between price per mile and distance traveled.
- **Slicer Filters**: Allow users to filter by date, weather condition, or cab type for more focused analysis.

Advanced Techniques

- **Window Functions**: Use these for calculating moving averages or lag/lead comparisons over time.
- **Quartile Analysis**: Identify price ranges and outliers to understand pricing strategies better.

Example Queries in Databricks

You might use the following SQL-like queries in Databricks for your analyses:

```sql
-- Average price by hour and day of the week
SELECT hour, day_of_week, AVG(price) as avg_price
```

```
FROM cab_data
GROUP BY hour, day_of_week
ORDER BY hour, day_of_week;

-- Surge multipliers based on weather conditions
SELECT rain, AVG(surge_multiplier) as avg_surge
FROM cab_data
GROUP BY rain;

-- Price per mile by cab type
SELECT cab_type, AVG(price_per_mile) as avg_price_per_mile
FROM cab_data
GROUP BY cab_type;
```

Conclusion

With these analyses and visualizations, you can gain valuable insights into how weather affects Uber and Lyft pricing strategies. Let me know if you need help with specific queries or visualizations!

On fail
On success
On completion
Skip

Linked service

Azure visualisation - powerbi, hdinsight

Scala - Traits

DAG

HDInsight and databricks
Azure synapse & ADF& Databricks
Pool in synapse
Etl pipeline
cache and persistence storage level
Spark job flow
Hadoop job flow
Pyspark
ADF

Transf
https://adb-3129472675843117.17.azuredatabricks.net/?o=3129472675843117#notebook/1955993164789931

dashboard
https://adb-3129472675843117.17.azuredatabricks.net/?o=3129472675843117#notebook/1955993164789987

https://adb-3129472675843117.17.azuredatabricks.net/?o=3129472675843117#notebook/1955993164789987

visualization
https://adb-3129472675843117.17.azuredatabricks.net/?o=3129472675843117#notebook/1955993164789987

Interview

03 October 2024 18:52

Pyspark

designed to process massive volumes of data quickly and efficiently.

From <https://www.datacamp.com/blog/pyspark-interview-questions>

Lazy Evaluation in PySpark
Lazy evaluation is one of the core features of Apache Spark, including PySpark. It means that Spark doesn't immediately execute operations when you define them. Instead, it builds an **execution plan (DAG - Directed Acyclic Graph)** and postpones the actual computation until an **action** (such as count(), collect(), show(), etc.) is called.

Coalesce - reduce the partition size without shuffling

the **Spark Driver** is a critical component of the Spark application architecture. It acts as the **master node** of a Spark application, responsible for orchestrating the execution of tasks on the cluster. The driver program is the entry point of the Spark application, running the main function that defines the application logic.

Components of the Driver:

1. **SparkSession**: The entry point to programming with the Spark API. It is used to initiate a SparkContext, which is necessary for any Spark application to interact with the cluster.
2. **DAG Scheduler**: The Directed Acyclic Graph (DAG) Scheduler in the driver breaks down the logical plan of the job into stages of tasks, which are then submitted to the cluster for execution.
3. **Task Scheduler**: Responsible for task submission and execution across executors. It tracks the completion of tasks and retries any failed tasks.
4. **Cluster Manager Communication**: The driver interacts with the cluster manager (e.g., YARN, Mesos, Kubernetes, or Spark's standalone cluster manager) to request resources for execution.

- **Driver (Master)**:
  - Schedules Jobs
  - Breaks Jobs into Tasks
  - Sends Tasks to Executors
  - Collects Results
- **Cluster Manager**:
  - Allocates Resources
  - Communicates with the Driver

- **Executors (Workers)**:
  - Executes Tasks
  - Reports back to the Driver

Hadoop is an open-source framework designed for distributed storage and processing of large datasets across clusters of computers. It enables the scalable and fault-tolerant processing of massive data volumes by dividing workloads across multiple machines.

Key Components of the Hadoop Ecosystem:

1. Hadoop Distributed File System (HDFS)

HDFS is the storage layer of Hadoop. It splits large files into blocks (default size is 128 MB or 64 MB) and distributes these blocks across the nodes in a cluster. HDFS

is designed to provide fault tolerance by replicating data blocks across multiple nodes, typically with a default replication factor of 3.

- **NameNode**: The master node that manages the metadata of HDFS (file system namespace, block locations). It does not store the actual data.
- **DataNodes**: The worker nodes responsible for storing data blocks. They periodically report back to the NameNode about their status.

2. MapReduce

MapReduce is the original programming model in Hadoop for processing large datasets. It works by breaking down a job into two phases:

- **Map Phase**: Processes input data in parallel by converting it into key-value pairs.
- **Reduce Phase**: Aggregates the results of the Map phase and combines the
- data to generate the final output.

MapReduce allows for distributed processing by running map and reduce tasks across multiple nodes, which makes it efficient for large-scale batch processing.

3. YARN (Yet Another Resource Negotiator)

YARN is the resource management layer in Hadoop. It allocates system resources (CPU, memory) and schedules jobs for execution in a distributed cluster.

- **ResourceManager**: The master component responsible for tracking available resources and assigning them to applications.
- **NodeManager**: The worker component that manages resources on individual cluster nodes.

YARN allows Hadoop to run multiple applications (like MapReduce, Apache Spark, or Apache Flink) in a single cluster

Azure's services, organizations can reduce operational complexity, gain scalability, and improve their ability to integrate with cutting-edge technologies.