Rev Points

05 February 2024 05:21 PM

1. Recursive cte
2. Self join
3. Join and then select in apache spark
4. CROSS APPLY , writing SP, FUNCTIONS, Table valued function,
5. Recursive CTE
6. DateDiff - for difference in month, days ,week
7. broadcasting live example
8. Direct query

Java study

14 July 2021 11:33 AM

1. Access modifiers
Public - accessible to all classes
Private -accessible to only class in which it is declared
Default - accessible to all classes within package

Super -
use to access the variable or method of parent class in child class. - super. Color // super.method()
Use to access parent class constructor - super()

2 D array input

```
Scanner inp = new Scanner(System.in);
      int n = inp.nextInt();
      int [][] arr = new int[n][n];
      for (int i=0;i<n;i++)
      {
         for (int j=0;j<n;j++)
         {
            arr[i][j]= inp.nextInt();
         }
      }
```

2 D array display

```
public static void display (int [][] arr, int n)
   {
      for(int i=0;i<n;i++)
      {
         for (int j=0;j<n;j++)
         {
            System.out.print(arr[i][j]+" ");
         }
         System.out.println();
      }
   }
```

```
# Program to iterate over string
for (int i = 0; i < str.length(); i++) {
         if (str.charAt(i)=='R')
            count++;
         else if (str.charAt(i)=='L')
            count--;
      }
```

 # program to transpose 2d array

```
 public static void Transpose( int [][]arr ,int n)
```

```
    {
        for(int i=0;i<n;i++)
        {
            for (int j=i;j<n;j++)
            {
                System.out.println(arr[i][j]+" "+arr[j][i]);
                swap(arr,i,j);
                System.out.println(arr[i][j]+" "+arr[j][i]);
            }
        }
    }
```

#swap - we have to pass array variable

```
public static void swap (int[][]arr,int i,int j)
    {
       int temp = arr[i][j];
        arr[i][j] = arr[j][i];
        arr[j][i]= temp;
    }
```

program to print 1d array

```
public static void display(int [] arr, int length)
    {
        for (int j=0;j<length;j++)
            {
                System.out.print(arr[j]+" ");
            }
    }
```

program to reverse an array

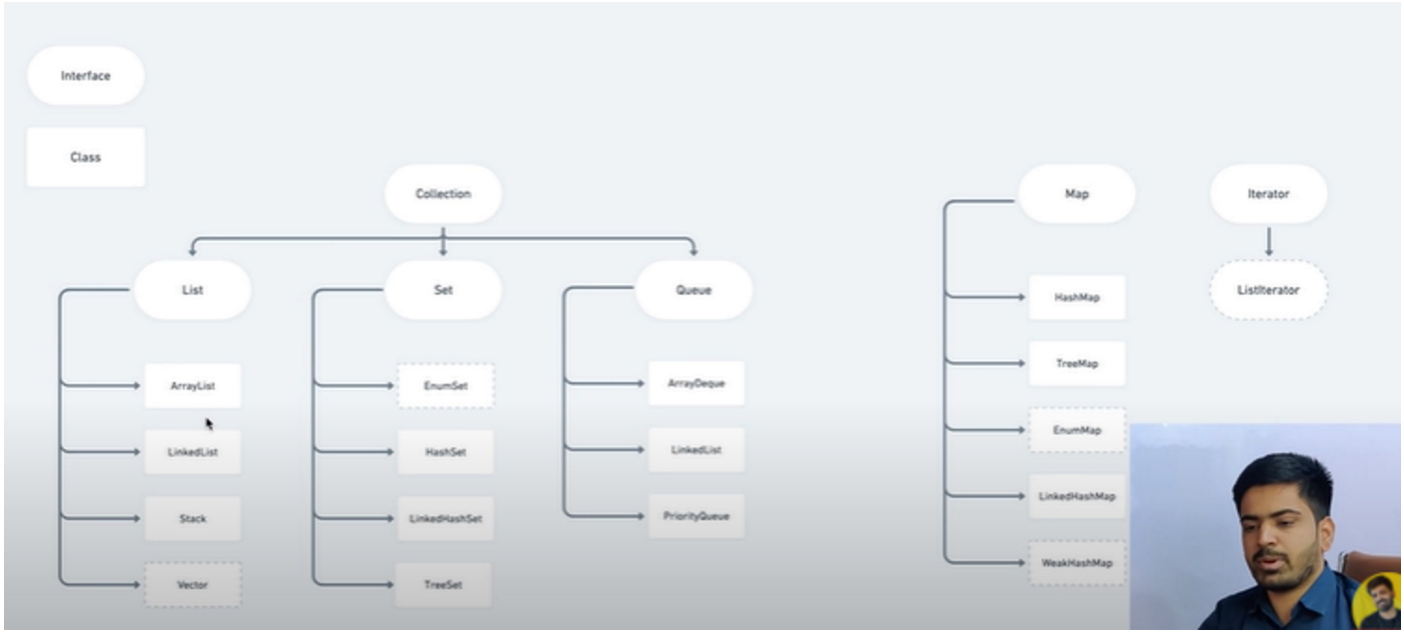
2 sum problem

Linked List find middle

```
import java.util.*;

public class Main {
   public static void main(String[] args) {
     System.out.println("Hello, World!");
     Node temp1 = new Node(1);
     Node temp2 = new Node(2);
     Node temp3 = new Node(3);
     Node head = null;
     head= temp1;
     temp1.next = temp2;
```

```
    temp2.next = temp3;
    printList(head);

  }

  public static void printList(Node head)
  {
    while(head!= null)
    {
     System.out.print(head.data + " ");
     head = head.next;
    }
  }
}

public class Node {
   int data;
   Node next;

   Node(int d) {
      data= d;
      next = null;
   }
}
```

Java collection



List	Add	Remove	Get	Contains	Next	Data Structure
ArrayList	O(1)	O(n)	O(1)	O(n)	O(1)	Array
LinkedList	O(1)	O(1)	O(n)	O(n)	O(1)	Linked List
CopyOnWriteArrayList	O(n)	O(n)	O(1)	O(n)	O(1)	Array

Set	Add	Remove	Contains	Next	Size	Data Structure
HashSet	O(1)	O(1)	O(1)	O(h/n)	O(1)	Hash Table
LinkedHashSet	O(1)	O(1)	O(1)	O(1)	O(1)	Hash Table + Linked List
EnumSet	O(1)	O(1)	O(1)	O(1)	O(1)	Bit Vector
TreeSet	O(log n)	O(log n)	O(log n)	O(log n)	O(1)	Red-black tree
CopyOnWriteArraySet	O(n)	O(n)	O(n)	O(1)	O(1)	Array
ConcurrentSkipListSet	O(log n)	O(log n)	O(log n)	O(1)	O(n)	Skip List

Queue	Offer	Peak	Poll	Remove	Size	Data Structure
PriorityQueue	O(log n)	O(1)	O(log n)	O(n)	O(1)	Priority Heap
LinkedList	O(1)	O(1)	O(1)	O(1)	O(1)	Array
ArrayDequeue	O(1)	O(1)	O(1)	O(n)	O(1)	Linked List
ConcurrentLinkedQueue	O(1)	O(1)	O(1)	O(n)	O(n)	Linked List
ArrayBlockingQueue	O(1)	O(1)	O(1)	O(n)	O(1)	Array
PrioiritityBlockingQueue	O(log n)	O(1)	O(log n)	O(n)	O(1)	Priority Heap
SynchronousQueue	O(1)	O(1)	O(1)	O(n)	O(1)	None!
DelayQueue	O(log n)	O(1)	O(log n)	O(n)	O(1)	Priority Heap
LinkedBlockingQueue	O(1)	O(1)	O(1)	O(n)	O(1)	Linked List

Map	Get	ContainsKey	Next	Data Structure
HashMap	O(1)	O(1)	O(h / n)	Hash Table
LinkedHashMap	O(1)	O(1)	O(1)	Hash Table + Linked List
IdentityHashMap	O(1)	O(1)	O(h / n)	Array
WeakHashMap	O(1)	O(1)	O(h / n)	Hash Table
EnumMap	O(1)	O(1)	O(1)	Array
TreeMap	O(log n)	O(log n)	O(log n)	Red-black tree
ConcurrentHashMap	O(1)	O(1)	O(h / n)	Hash Tables
ConcurrentSkipListMap	O(log n)	O(log n)	O(1)	Skip List

Collection framework

HashMap

Declaration
HashMap<Integer,String> Map=new HashMap<Integer,String>();

1. Map.clear() - remove all mapping
2. Map.isEmpty() - empty check

3. for(Map.Entry m : map.entrySet()){
   System.out.println(m.getKey()+" "+m.getValue());
   }
   Iterating over the map

4. Map can accept 1 null key. It doesn't allow duplicate key - if we put duplicate key then it will update the following value.
5. Map.put(100, "Amar") - adding items in Map.
6. map.remove(102, "Rahul"); - removing element ,
7. Map.get(100) - get value based on key.

8. Map.size() - size of the map

HashSet

HashSet<String> set=new HashSet();
set.add("One"); - add the specific element to set
Set.contains() - search for element
isEmpty() - empty check

Azure Networking

23 July 2023 10:04 AM

Azure Policy
Azure Arc
Azure cli vs azure powershell
Health Advisors - Health advisories are issues that require that you take proactive action to avoid service interruptions, such as service retirements and breaking changes. Service issues are problems such as outages that require immediate actions.

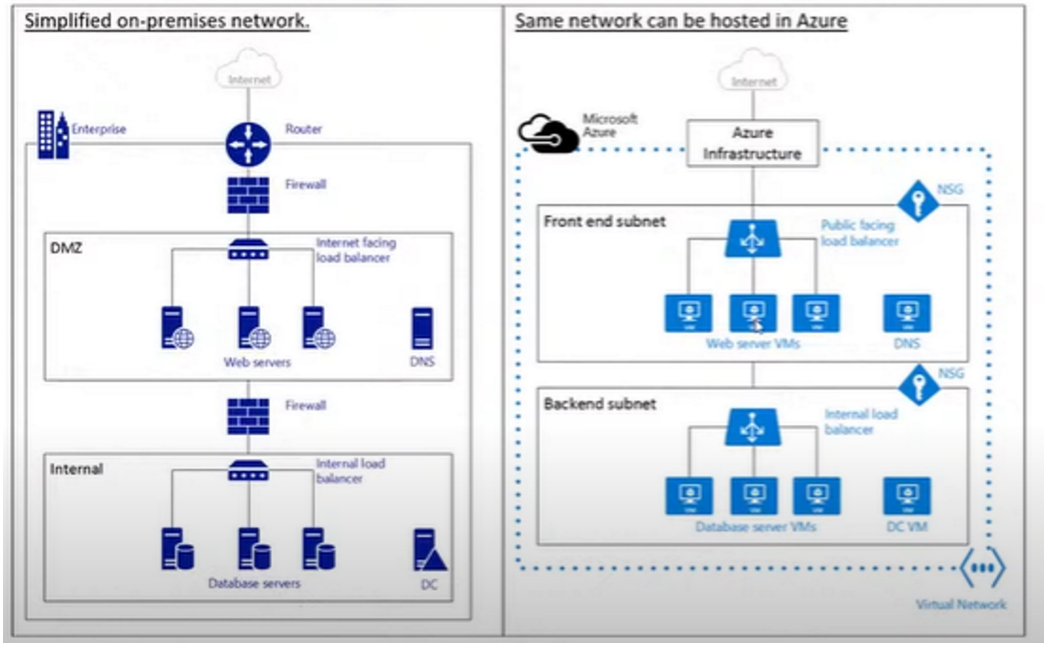
Azure Service Health
Private cloud

management groups, Azure AD user accounts

Azure VPN gateway
Azure Files
Conditional Access

Resource lock , resource tag

Network - interconnection of computers , computers are physically connected.
On these network we have servers.



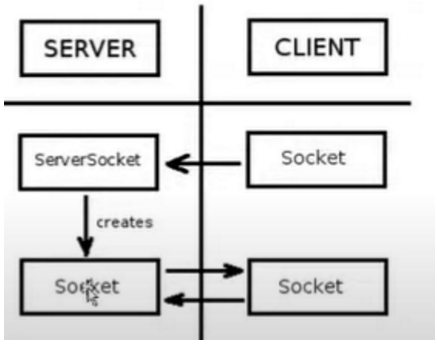
DMZ - demilitarized zone
Virtual network -Network in cloud. doesn't have cost.
NSG - similar to firewall

Benefits

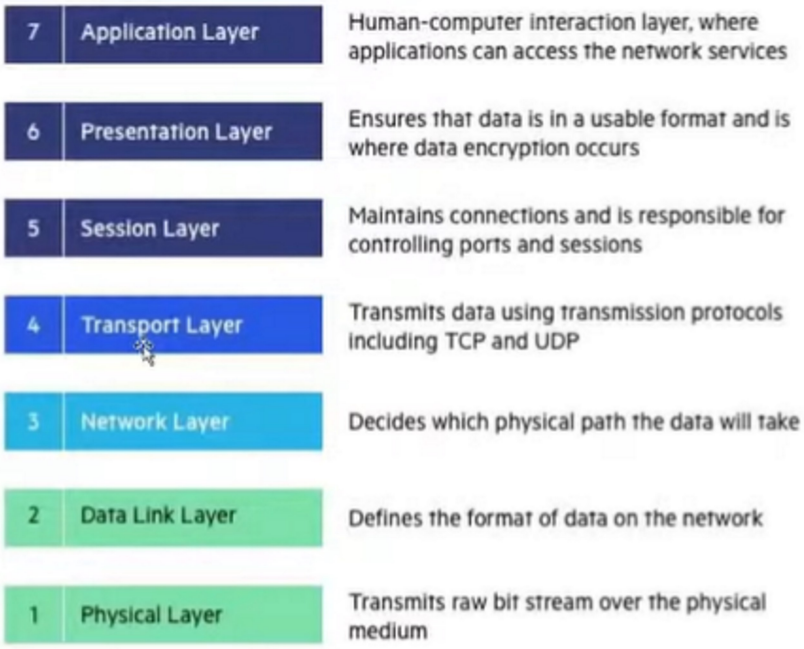
1. Isolation
2. Access to public internet - by default in VM

3. Security - by NSG
4. By default No Access to VM in Virtual Network. Machine in same subnet can connect to each other.
5. DNS is built in every Vnet. Then we can have IP address to domain mapping

Socket - IP(same for machine) + Port No (specific to application) - code is written by developer
Server and client communication happens through sockets .
When clients sends some request to a port i.e. port 80, server will create a new socket and communication will happen from that socket



OSI Layer -
From Application to wire



IP address -

A. Public IP- Name on Aadhar card of person , used outside the Vnet. For machines connecting to Internet.
B. Private IP - nickname of a person , used within same Vnet. Intranet

CIDR notation -
Small network - 192.168.0.X
Larger - 172.16.X.X
Very Large network - 10.X.X.X

A.B.C.D/N - N - no of fixed bytes

202.123.1.4/32 - only 1 IP address

192.168.0.0 to 192.168.0.255 - 192.168.0.0/24 - 24 bits are fixed

Dynamic IP and Static IP -
Static - Resources which will give same IP address when started and stopped
Dynamic - Resource which will give different IP address when started and stopped

Subnet - Range of IP address

VNET - 192.168.0.0 to 192.168.255.255 = 192.168.0.0 / 16

Subnet 0 - 192.168.0.0 to 192.168.0.255 = 192.168.0.0 / 24
Subnet 1 - 192.168.1.0 to 192.168.1.255 = 192.168.1.0 / 24

5 IP address are reserved in subnet 0.0, 0.1,0.2,0.3, 0.255 , we get only (251) IPs.

NIC - Network internet Card - VM doesn't have IP address. VM has NIC and NIC have IP address.

Network Security Group - NSG is defined at Subnet level.
Inbound and Outbound Rule

Azure Load Balancer - Layer 4 (TCP, UDP)

Application Gateway - Layer 7

Traffic manager - Works Outside Vnet.

Front Door -

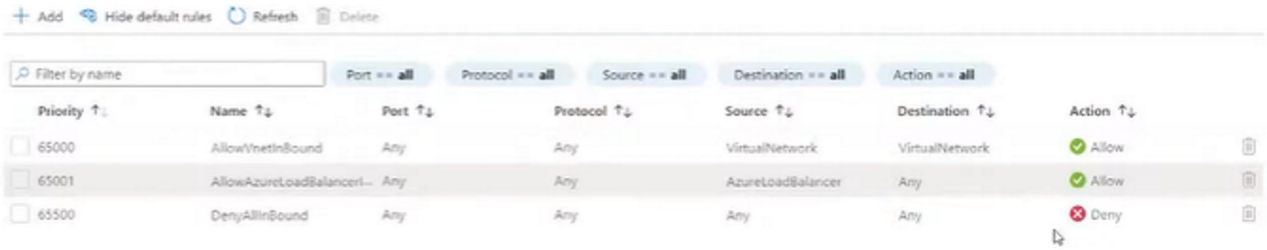
VPN Gateway - On premise to Azure VM

Azure DNS - domain name mapping in azure Vnet.

Practical -

Create VNET, SUBNET , view Diagram

Create NSG



Inbound and Outbound Rules
Rules are executed as per priority .

Virtual machine will get a subnet -

NSG can be done at subnet or VM level.

_____

Create Vnet ->
Give an IP range - 192.168.0.0/16

Inside vnet we have to create subnet
Frontend-subnet - 192.168.1.0/24 - (192.168.1.0 to 192.168.1.255)
Backend-subnet - 192.168.2.0/24 - (192.168.2.0 to 192.168.2.255)

See the diagram of vnet

VM - network interface card
Subnet - subnet ip

Create inbound and outbound rules

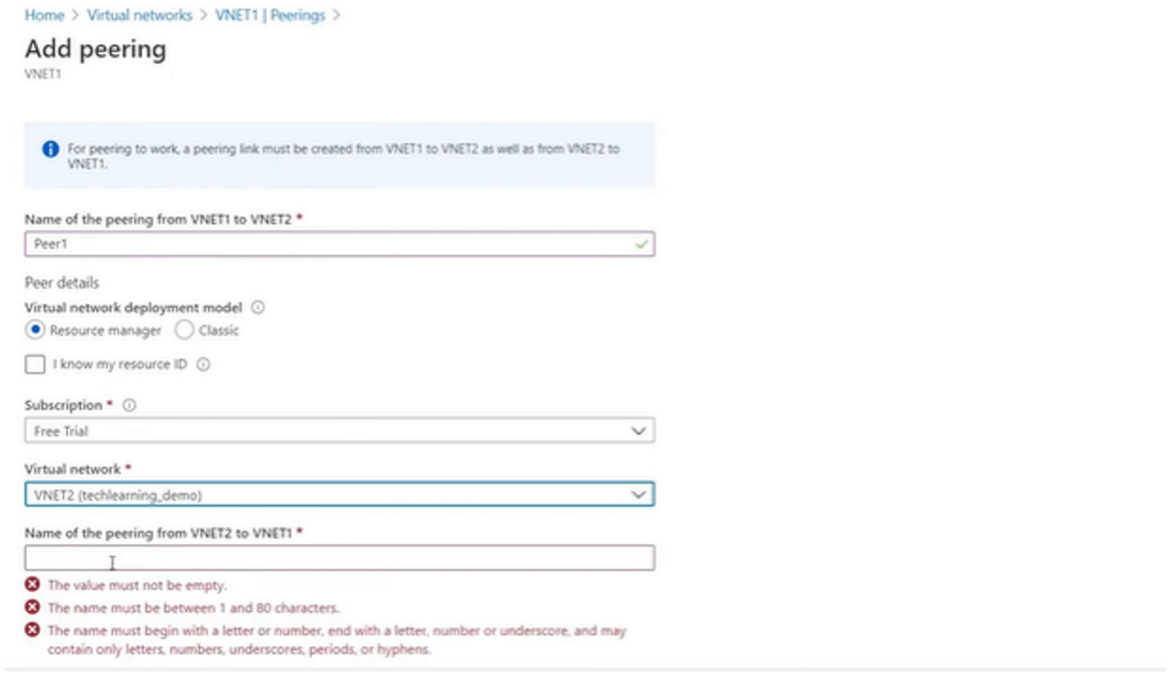
The built in rules cannot be deleted
Same NSG can be used for multiple subnet , but one subnet can have only one nsg

Create VM under same subnet

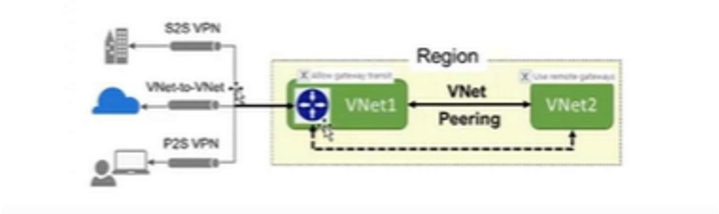
_____

Two resources in same VN can connect to each other
Vnet pairing - for communication between 2 vnet

Regional Vnet peering - if the both Vnet are in same region
Global Vnet peering - if both Vnet are in different Vnet



Network gateway transit - for connection from on prem to VPN



NSG - to control inbound and outbound traffic

Command tools -

v

Azure API management - Gateway service for API.

MB - 1000KB
Mib -
Mb - Megabits

Data Warehousing

17 January 2024 08:41 PM

Its database designed for BI activities. - Query and analysis - read oriented OLAP
Maintain large historical data
Includes ETL and multiple data sources
Single source of truth

Data mart - subset of DW
Operation data store (ODS) - for daily operations , no historical

Characteristics
Subject oriented -
Integrated
Non volatile
Time varient

Adhoc queries
Partially normalized

Schema - collection of database objects -tables , view, indexes.

Integrating Heterogeneous database
Query driven approach - Complex
Update driven approach - Easy and performant

Metadata - data used to represent other data

Data cube - data is represented with dimensions and facts

Star Schema -
Each dimension with one dim table

MongoDB

14 July 2021 11:33 AM

NoSQL Database

Difference between SQL and NoSQL databases
---
1. SQL databases - relational and NoSQL -> non-relational
2. SQL database - structured query language + predefined schema
   NoSQL database - dynamic schemas because of unstructured data
3. SQL database > table based
   NoSQL database -> document based
   format : JSON (JavaScript Object Notation)
4. SQL database - join operation
   NoSQL - joins not required

employee information in json
{"empId" : 100, "empName" : "Ram", "salary" : 34567.80 }

Features of NoSQL :-

NoSQL is for non-structured data
NoSQL is Not Only SQL

suitable for large volume of non-structured data
No Joins
easy to work
no fixed query language
document based
writen in C++

Example of NoSQL DB
1. document store - MongoDB
2. Key-value store - Dynamo
3. Graph store - Polyglott
4. field store - Cassandra

MongoDB structue ->
Database
Collection
Document

Comparasion with Relational databases

NoSQL	RDBMS
Database	Database
Collection	Table / Relation
Document	Tuple/Row/Record
Object Identifier	Primary Key

Document will in JSON format - BSON (Binary representation of JSON format)
Object Identifier

_id - unique
first field in a document
hexadecimal number
12 bytes
by default mongodb will provide
We also can provide

Document will compose of field - value pair
Naming rule for field :-
- cannot start with $
- cannot contain dot .
- cannot contain null character
- field name cannot be _id (reserved for primary key)

Maximum size for BSON document - 16 MB

Work with MongoDB
1. start MongoDB Server
open command prompt and type -> mongod

2. start MongoDb shell
open new command prompt and type -> mongo

3.use <databasename> -> it will switch to the database if databse already there
in case if database is not there, create new database

4. db.getName() -> show current db

5. db.createCollection('friends') -> create collection in current db

6. show collections -> show all the collections in current db

7. db.<collection_name>.drop() > drop the collection
example :- db.friends.drop() -> will drop friends collection

8. db.courses.insert({'name':'Core Java'}) -> insert a new document in a collection

9. db.courses.find() -> display documents of collection

=> create new database moviedb
=> in moviedb, create a collection movies
{'name':'Bahubali 2','category':'Action','year':2017}
10. enter multiple documents
db.movies.insert([{'name':'Bahubali 2','category':'Action','year':2017},
{'name':'Zanjeer','category':'Action','year':1975},
{'name':'Golmal','category':'Comedy','year':1970}])

11. db.movies.find().count() -> count no of documents in a movies collection

12. restrict to add only one document
db.movies.insertOne({'name':'Avengers Endgame','cateogry':'Sci-fi','year':2019})

db.movies.insertOne([{'name':'Bahubali 2','category':'Action','year':2017},
{'name':'Zanjeer','category':'Action','year':1975},
{'name':'Golmal','category':'Comedy','year':1970}])

=> lead to an error -> operation passed in cannot be an Array

13. db.movies.save({'name':'Avengers','cateogry':'Sci-fi','year':2012})

db.movies.save({"name" : "Golmal", "category" : "Comedy", "year" : 1974})

db.movies.save({ "_id" : ObjectId("60f9166515315b0ed226a7f6"), "name" : "Golmal", "category" : "Comedy", "year" : 1980 })

db.movies.save ..... -> If document doesn't exist, It inserts new document,
If document exists, It updates the document.

14. Save with user defined Id
db.movies.save({"_id": 1, "name" : "Quiet Place", "category" : "Thriller", "year" : 2012})

db.movies.save({"_id": 1, "name" : "Quiet Place", "category" : "Horror", "year" : 2012})
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 1 })

db.movies.save({"_id": 1, "name" : "Quiet Place", "category" : "Horror", "year" : 2012})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 })

db.movies.save({"_id": 1, "name" : "Quiet Place", "category" : "Thriller", "year" : 2012})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

15. Add arrays of a field - category
db.movies.save({"_id": 2, "name" : "Tanahji", "category" :["Action","History"], "year" : 2020, "actor":"Ajay Devgan"})

16. db.movies.find().pretty() - to display documents in proper format

Filters :-

17. display those movies which were released in 2020
db.movies.find({'year':2020})

18. display those movies which were released before 2020
db.movies.find({'year':{$lt:2020}})

$lt = less than
$lte = less than or equal to
$gt = greater than
$gte = greater than or equal to
$ne = not equal to

$lt:2020

{'key':{$lt:<value>}}
{'key':{$gt:<value>}}
{'key':{$lte:<value>}}
{'key':{$gte:<value>}}

19. display those movies which were released in or before 1980
db.movies.find({'year':{$lte:1980}})

db.movies.find({'year':{$ne:1980}})

20. display those movies which were released after 2000 and before year 2019
    db.movies.find({'year':{$gt:2000,$lt:2019}})

    db.movies.find({'year':{$gte:2000,$lte:2019}}) // including year 2000 and 2019 in condition

21. display all action movies which were released in 2000 or afterwards
    use logical operator - and, or

    db.movies.find({$and:[{'year':{$gte:2000}},{'category':'Action'}]})

    {$and:[{'year':{$gte:2000}},{'category':'Action'}]}

22. display all Sci-fi movies which were released after 2010
db.movies.find({$and:[{'year':{$gt:2010}},{'category':'Sci-fi'}]})

23. List all the documents where year is not 1980 or category is Comedy
    db.movies.find({$or:[{'year':{$ne:1980}},{'category':'Comedy'}]})

24. List all the documens where release year is not 2000 and category is comedy or movie could be Golmal

db.movies.find({'year':{$ne:2000},$or:[{'category':'Comedy'},{'name':'Golmal'}]})

{'year':{$ne:2000}}
{'category':'Comedy'}
{'name':'Golmal'}

25. Projection - select certain columns
    select * from employee ;
    select empname, dept from employee;

    display moviename for all the documents.

    db.movies.find({},{'name':1})

    db.movies.find({},{'name':0}) - except movie name, display all the columns

    db.movies.find({'year':{$gt:2010}},{name:1})

26. Delete a document/s
    db.movies.find({'category':{$in:['Comedy','Action']}})

    db.movies.deleteMany({'name':'Zanjeer'})

    db.movies.remove({'name':'Golmal'})

27. to display multiple columns
    db.movies.find({},{'name': 1,'year' : 1})

SQL optimization Query

27 December 2023 09:52 AM

```
--set statistics io on
--set statistics time on
--select c.DepartmentCode, c.CourseNumber, c.CourseTitle,c.Credits,ce.grade
--from CourseEnrollments ce
--inner join CourseOfferings co on co.CourseOfferingId = ce.CourseOfferingId
--inner join Courses c on co.DepartmentCode = c.DepartmentCode and co.CourseNumber =
c.CourseNumber
--where ce.StudentId = 29717

select

select co.*
from CourseOfferings co
left join CourseEnrollments ce on co.CourseOfferingId = ce.CourseOfferingId
where co.TermCode = 'SP2016' and ce.CourseOfferingId is null


--select top record only
select co.*
from CourseOfferings co
where  not exists
(select 1 from CourseEnrollments ce where co.CourseOfferingId = ce.CourseOfferingId)
and co.TermCode = 'SP2016';



CREATE INDEX IX_Applicants_FirstNameLastName
on Applicants (LastName,FirstName, State );


select * from Applicants where LastName ='Davis' and state ='CO';


-----------------------------------


SQL Complex 1 - ICC tournament

create table icc_world_cup ( Team_1 Varchar(20), Team_2 Varchar(20), Winner Varchar(20) ); INSERT
INTO icc_world_cup values('India','SL','India'); INSERT INTO icc_world_cup values('SL','Aus','Aus'); INSERT
INTO icc_world_cup values('SA','Eng','Eng'); INSERT INTO icc_world_cup values('Eng','NZ','NZ'); INSERT
INTO icc_world_cup values('Aus','India','India'); select * from icc_world_cup;


select
```

```
team_1 as team,
case when winner = team_1 then 'Y' ELSE 'N' END as ct
from icc_world_cup
),
cte2 as
(
select
team_2 as team,
case when winner = team_2 then 'Y' ELSE 'N' END as ct
from icc_world_cup
),
cte3 as
(
select * from cte1 union all SELECT * from cte2
)
SELECT team,
count(1),
COUNT(CASE WHEN ct = 'Y' THEN 1 END) as SUCCESS,
COUNT(CASE WHEN ct = 'N' THEN 1 END) As FAILURE
from cte3 GROUP by team
```

Complex 2

```
select order_date,
count(1) as total,
count(case when rn =1 then 1 end) as new,
count(1) - count(case when rn =1 then 1 end) as old
from
(
select *,
row_number() over( partition by customer_id order by order_date asc) as rn
from customer_orders
)z group by order_date
```

```
MERGE PRODUCT_LIST AS TARGET
	USING UPDATE_LIST AS SOURCE
	ON (TARGET.P_ID = SOURCE.P_ID)
	WHEN MATCHED
		AND TARGET.P_NAME <> SOURCE.P_NAME
		OR TARGET.P_PRICE <> SOURCE.P_PRICE
	THEN UPDATE
		SET TARGET.P_NAME = SOURCE.P_NAME,
		TARGET.P_PRICE = SOURCE.P_PRICE
	WHEN NOT MATCHED BY TARGET
	THEN INSERT (P_ID, P_NAME, P_PRICE)
```

VALUES (SOURCE.P_ID, SOURCE.P_NAME, SOURCE.P_PRICE)

Recursive CTE

Clustured Index - primary key

```
CREATE TABLE production.part_prices(
   part_id int,
   valid_from date,
   price decimal(18,4) not null,
   PRIMARY KEY(part_id, valid_from)
);
```

Non clustured index -

```
CREATE [NONCLUSTERED] INDEX index_name
ON table_name(column_list);
```

Code language: SQL (Structured Query Language) (sql)
In this syntax:

```
CREATE INDEX ix_customers_city
ON sales.customers(city);
```

SQL Server

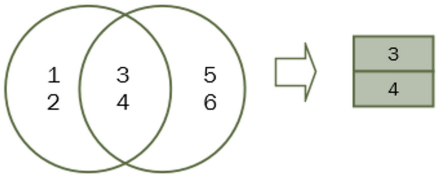
23 July 2021 11:38 AM

1. Structured query language for all RDBMS databases such as
2. MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.

Table - data in RDBMS is stored in form of tables
Field - column heads
Record - row
Column - vertical entry
Null - no value ,left blank

As - for giving different name to database

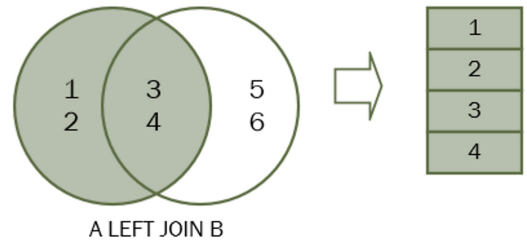
Inner Joins




```
SELECT
    first_name,
    last_name,
    employees.department_id,
    departments.department_id,
    department_name
FROM
    employees
        INNER JOIN
    departments ON departments.department_id = employees.department_id
WHERE
    employees.department_id IN (1 , 2, 3);f
```

```
SELECT
        first_name,
        last_name,
        job_title,
        department_name
FROM
        employees e
INNER JOIN departments d ON d.department_id = e.department_id
INNER JOIN jobs j ON j.job_id = e.job_id
WHERE
        e.department_id IN (1, 2, 3);
```

Left Join



A LEFT JOIN B

Full outer join

A

n
1
2
3

B

c
x
y
z

```
SELECT *
FROM A
CROSS JOIN B
```

A x B

n	c
1	x
1	y
1	z
2	x
2	y
2	z
3	x
3	y
3	z

Type	Great fit for	Watch out if
Replicated	✓ Small dimension tables in a star schema with less than 2 GB of storage after compression (~5x compression)	❑ Many DML operations are on table (such as insert, merge, delete, update) ❑ You change Data Warehouse Units (DWU) so frequently ❑ You only use 2-3 columns but your table has many columns ❑ You have index in replicated table
Round Robin (default)	✓ Temporary/staging table ✓ No obvious joining key or good candidate column	❑ Performance is slow due to data movement
Hash	✓ Fact tables ✓ Large dimension tables greater than 2 GB	❑ The distribution key cannot be updated

Subtitles/closed captions unavailable

Hash - use hash function , large fact table, table with frequent insert , update and delete
Round Robin - temporary , staging table - when you have no idea of hash key,
Replicated - for small dim tables,

SQL server Tutorial

Data types

NULL - absence of data, represents nothing ctrl+ 0 to insert null
Identity Column - numeric & auto increment - it should not be nullable - not reliable
Unique Key - cannot have duplicate , can have one null -
Primary Key - used to locate a record , want to locate record , no null , cannot be changed
Composite key - both are primary and their composite is unique

Foreign Key - to maintain referential integrity , Reference of one table from another -
we can put constraint on table using foreign key

Normalization - > we try to split the table depend on logic
Data should not be bad, redundant, and duplicate
One column should store only one value

Non Atomic - Pirates , Clash

Design mistake	Problem			
No primary key	Duplicate			
Atomic value	Duplicate, update ,delete			
Repeating groups	Redundant	1nf (key)	Only atomic values	

Partially depend on primary key	Redundant, Integrity	2nf (Full key)	No partial dependency, primary key	
Transitive dependency	Redundant, Integrity	3nf (only on the key)	No transient dependency	

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

1NF - single valued attribute , no duplicacy

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

2NF - FNF and no partial dependency (There should be single column primary key)

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

3NF - No transitive dependency (non-key column Full Name may change Salutation.)

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

DENORMALIZATION

	OLTP	OLAP
Design	Normalized. (1st normal form, second normal form and third normal form).	Denormalized (Dimension and Fact design).
Source	Daily transactions.	OLTP.
Motive	Faster insert, updates, deletes and improve data quality by reducing redundancy.	Faster analysis and search by combining tables.
SQL complexity	Simple and Medium.	Highly complex due to analysis and forecasting.

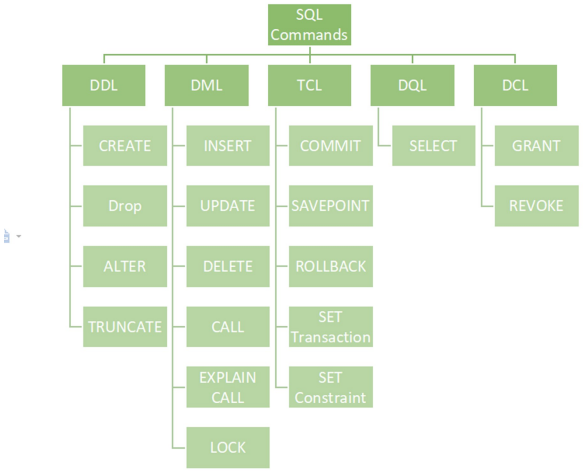
STAR and SNOW FLAKE DESIGN

Dimensions - Dimensions of data ,stored in dimension table
Measures - values, stored in fact table

Star design - pure denormalized
Snow flake - normalized

	Snowflake Schema	Star Schema
Normalization	Can have normalized dimension tables.	Pure denormalized dimension tables.
Maintenance	Less redundancy so less maintenance.	More redundancy due to denormalized format so more maintenance.
Query	Complex Queries due to normalized dimension tables.	Simple queries due to pure denormalized design.
Joins	More joins due to normalization.	Less joins.
Usage guidelines	If you are concerned about integrity and duplication.	More than data integrity speed and performance is concern here.

SQL command



DML

1. Order by c1,c2 - sort on c1+c2
2. Distinct c1,c2 - distinct on c1+c2
3. Like - % - any number of letter , _ - one lettter , ^ not
   select * from demo where Name like '[K]%' - values starting with K
   select * from demo where Name like '[^K]%' - values not starting with K
4.
5. Case -
   select *,
   case
   when CustomerId >= 100 and CustomerId< 105 then 'Medium'
   when CustomerId >= 105 and CustomerId< 110 then 'High'
   else 'NA'
   end as TRE
   from demo
6. Union - gives unique rows

select id, Name from demo
union
select CustomerId, CustomerName from Customer

7. Union all - includes duplicate rows also

Inner Join -

```
select  demo.id, demo.Name, demo.City, demo.CustomerId, Address.Address
from demo Inner join  Address on demo.CustomerId = Address.CustomerId
```

Left Join -

```
select  demo.id, demo.Name, demo.City, demo.CustomerId, Address.Address
from demo left join  Address on demo.CustomerId = Address.CustomerId
```

Right Join -

```
select  demo.id, demo.Name, demo.City, demo.CustomerId, Address.Address
from demo right join  Address on demo.CustomerId = Address.CustomerId
```

Full outer Join

```
select  demo.id, demo.Name, demo.City, demo.CustomerId, Address.Address
from demo full outer join  Address on demo.CustomerId = Address.CustomerId
```

Cross Join - all rows of both table permutation and combination
select * from demo cross join Address

Having - apply filter on group - >

Self Join - Join on same table

```
SELECT column_name(s)
FROM table1 T1, table1 T2
WHERE condition;
```

ISNULL
select isnull(CustomerPhone,-1) from demo

SUBQUERY

```
select * from demo
where id in (select CustomerCode from demo)
```

CORELATED QUERY - get first, second, third highest

```
select * from demo as d
where 1 = (select count(1) from demo d2 where d2.CustomerId > d.CustomerId )
```

BETWEEN () - both inclusive

Max, Min , Avg - aggregated process

SEELECT * INTO -
create new table with existing data without create syntax
select * into Customer from demo

```
Insert into Address(CustomerId, Address) values ( 105, 'Pune')
```

BULK insert

```
Insert into Address select * from customer
```

```
update Address set Address = 'Luknow' where CustomerId =101
```

```
Delete from Address
```

DDL

CREATE database Customer

- Creates 2 files mdf - Row data , ldf - log file

Create table Customer

```
create table Employee2
(
	id int unique,
	name nvarchar(50) unique not null
)
```

Alter table

```
Alter table Address
ADD Email varchar(255)
DROP COLUMN column_name
RENAME COLUMN old_name to new_name
ALTER COLUMN column_name datatype;
```

```
TRUNCATE TABLE Student_details;
```

Add Constraint

```
alter table Employee
add constraint constraint_salary2
 check (salary > 10000)
```

Default

```
add constraint def_const default 0 for dependents
```

Transaction

- set of tasks , all will get executed or all will be rolled back

```
begin tran
insert into Employee values(12,'fread',45000,0)
insert into Employee values(11,'fread',45000,0)
if(@@ERROR > 0)
begin
	rollback tran
end
begin
	commit tran
end
```

@@error - global variable
@@trancount - gives the current transaction count

Concurrency

A. By default the transaction of SQL server are locked at **row level. Only the Particular row is locked. The user cannot see the volatility of transaction.**

```
begin transaction
	update Employee set name='Vikas' where id=7
	waitfor delay '00:00:10'
	update Employee set name='Ramesh' where id=7
rollback transaction
```

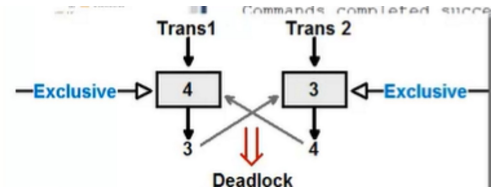
When you try to query data in between transaction
, query will go in wait state and you wont see any output
Until the transaction is complete

B. If you don't want to go in blocking mode, you want to see volatility of data,
Then you can use nolock.

```
select * from Employee with (nolock)
```

Blocking and deadlock

A deadlock occurs when two or more processes or transactions block each other from continuing because each has locked a database resource that the other transaction needs. SQL Server handles deadlocks by terminating and rolling back transactions that were started after the first transaction.

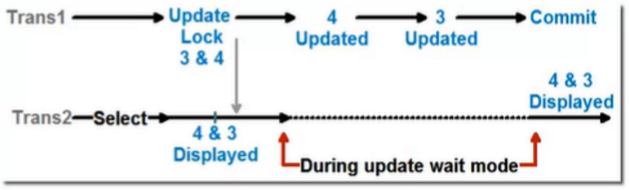


```
--transaction 1
begin transaction
update Employee set name='kada' where id=5
waitfor delay '00:00:20'
update Employee set name='Vada' where id=6
commit transaction


--transaction 2
begin transaction
update Employee set name='kada' where id=6
waitfor delay '00:00:20'
update Employee set name='Vada' where id=5
commit transaction
```

How to avoid deadlock -
Keep the time small
Maintain same chronological order

Updlock -
It will lock all the locks in deadlock mode

```
begin transaction
select * from Employee with (updlock) where id=5 or id =6
update Employee set name='kada' where id=5
waitfor delay '00:00:20'
update Employee set name='Vada' where id=6
commit transaction
```



Shared lock

it is applied only during select query
Update tblCustomer set name ='sometax' where id=4

When we execute any update query in sql server, internally it applies all 3 locks

Select (shared)
Calculation (update lock)
Actual update (exclusive lock)

Lock can be applied on
Rows - Pages - Table - DB

spid	dbid	ObjId	IndId	Type	Resource	Mode	Status
51	5	0	0	DB		S	GRANT
52	5	0	0	DB		S	GRANT
52	1	1467152272	0	TAB		IS	GRANT
52	32767	-571204656	0	TAB		Sch-S	GRANT
54	5	0	0	DB		S	GRANT
54	5	1029578706	0	TAB		IX	GRANT
54	5	1029578706	1	PAG	0.313888889	IX	GRANT
54	5	1029578706	1	KEY	(a0c936a3c965)	X	GRANT

I - Intent
IS - Intent shared lock
IX - Intent Exclusive lock

ISOLATIONAL LEVEL

Its very important when we have multiple transactional in sql server

Problems with nolock
1. Repeatable read -

We see new data in different reads within same transaction

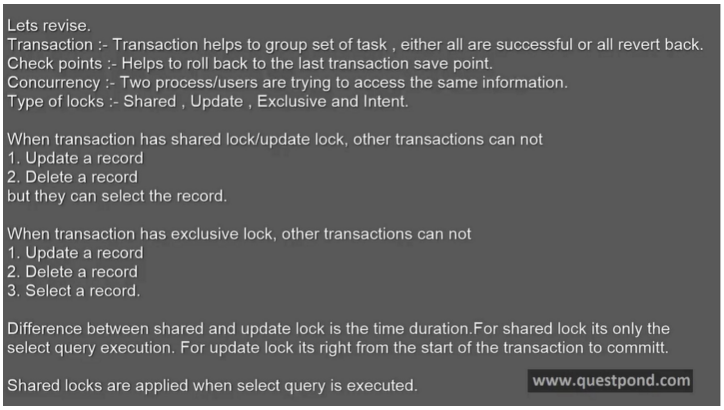
2. Dirty read
   We see data which never got committed.

3. Phantom read (Ghost)
   We don't see data at first time , but the next time we see that record

Isolation Level	Dirty Read	Non-Repeatable Read	Phantom
Read uncommitted	Yes	Yes	Yes
Read committed	No	Yes	Yes
Repeatable read	No	No	Yes
Snapshot	No	No	No
Serializable	No	No	No

ACID

Atomicity - All transaction should commit or roll back
Begin tran , commit tran
Consistency - transaction should create a valid state of new data
Save tran
Isolation - transaction must be isolated from other transaction
Isolation level
Durability - commited data must be stored in correct state , back up
Back up, mirroring



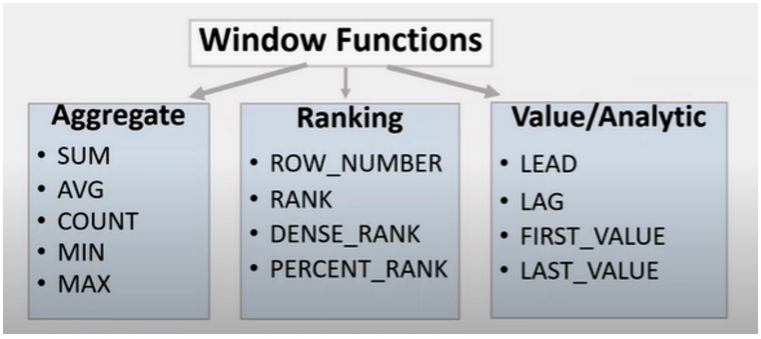
---

1. Isnull - can take 2 column
   select isnull(Name, 'Vada') as t from demo
2. Coalesce - returns the first non null column from the list of column
   Select coalesce(FirstName, lastName, PetName) as Name from tblPerson

---

Windows Function

A window function performs a calculation across a set of table rows that are somehow related to the current row. This is comparable to the type of calculation that can be done with an aggregate function. But unlike regular aggregate functions, use of a window function does not cause rows to become grouped into a single output row — the rows retain their separate identities. Behind the scenes, the window function is able to access more than just the current row of the query result



3. row_number() - generates a unique numbering for each row in a column

4. Partition - if we want different row numbering for different groups , then we use partition
5. Rank() - generates same number for same records and do numbering ,but skips the continuous numbers
6. Dense_rank()- generates same number for same records and do numbering ,but skips the continuous numbers

```
select ROW_NUMBER() over (order by customer) as row_num,
ROW_NUMBER() over (partition by Vendor order by Vendor) as row_num_partition,
rank() over (order by customer) as rank_customer,
dense_rank() over (order by customer) as dense_rank_customer,
* from productDetails
```

row_num	row_num_partition	rank_customer	dense_rank_customer	customer	Product	Amount	Vendor
1	1	1	1	Abhay	Bag	200	Atlas
6	2	6	4	Vasu	Pant	300	Atlas
2	1	2	2	Amar	Shoes	100	Data
4	1	4	3	Ramesh	Shirt	150	Polo
5	2	4	3	Ramesh	Shirt	250	Polo
3	1	2	2	Amar	Shoes	100	Reebook

```
select * from Employee
-- Get total aggregates
select id , name,
sum(id) over () as 'total',
avg(id) over () as 'avg' ,
count(id) over () as 'count',
min(id) over () as 'min'
from Employee
```

	id	name	total	avg	count	min
1	1	Ajay	66	6	11	1
2	2	Vikas	66	6	11	1
3	3	Kukas	66	6	11	1
4	4	Freak	66	6	11	1
5	5	Ful	66	6	11	1
6	6	Ful	66	6	11	1
7	7	fread	66	6	11	1
8	8	fread	66	6	11	1
9	9	fread	66	6	11	1
10	10	fread	66	6	11	1
11	11	fread	66	6	11	1

```
-- Get partition by aggregates
select id , name,
sum(id) over (partition by name) as 'total',
avg(id) over (partition by name) as 'avg' ,
count(id) over (partition by name) as 'count',
min(id) over (partition by name) as 'min'
from Employee
```

	id	name	total	avg	count	min
1	1	Ajay	1	1	1	1
2	7	fread	45	9	5	7
3	8	fread	45	9	5	7
4	9	fread	45	9	5	7
5	10	fread	45	9	5	7
6	11	fread	45	9	5	7
7	4	Freak	4	4	1	4
8	5	Ful	11	5	2	5
9	6	Ful	11	5	2	5
10	3	Kukas	3	3	1	3
11	2	Vikas	2	2	1	2

```
-- Get Running aggregates
select id , name,
sum(id) over (order by id) as 'total',
avg(id) over (order by id) as 'avg' ,
count(id) over (order by id) as 'count',
min(id) over (order by id) as 'min'
from Employee
```

	id	name	total	avg	count	min
1	1	Ajay	1	1	1	1
2	2	Vikas	3	1	2	1
3	3	Kukas	6	2	3	1
4	4	Freak	10	2	4	1
5	5	Ful	15	3	5	1
6	6	Ful	21	3	6	1
7	7	fread	28	4	7	1
8	8	fread	36	4	8	1
9	9	fread	45	5	9	1
10	10	fread	55	5	10	1
11	11	fread	66	6	11	1

```
-- Get partitioned running aggregates
select id , name,
sum(id) over (partition by name order by id) as 'total',
avg(id) over (partition by name order by id) as 'avg' ,
count(id) over (partition by name order by id) as 'count',
min(id) over (partition by name order by id) as 'min'
from Employee
```

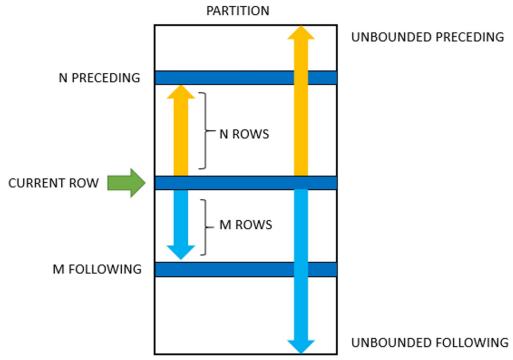
	id	name	total	avg	count	min
1	1	Ajay	1	1	1	1
2	7	fread	7	7	1	7
3	8	fread	15	7	2	7
4	9	fread	24	8	3	7
5	10	fread	34	8	4	7
6	11	fread	45	9	5	7
7	4	Freak	4	4	1	4
8	5	Ful	5	5	1	5
9	6	Ful	11	5	2	5
10	3	Kukas	3	3	1	3
11	2	Vikas	2	2	1	2

```
-- Get Ranking Functions
select id , name,
row_number() over (order by id) as 'row_number',
rank() over (order by id) as 'rank' ,
dense_rank() over (order by id) as 'dense_rank',
PERCENT_RANK() over (order by id) as 'PERCENT_RANK'
from Employee
```

	id	name	row_number	rank	dense_rank	PERCENT_RAN
1	1	Ajay	1	1	1	0
2	2	Vikas	2	2	2	0.1
3	3	Kukas	3	3	3	0.2
4	4	Freak	4	4	4	0.3
5	5	Ful	5	5	5	0.4
6	6	Ful	6	6	6	0.5
7	7	fread	7	7	7	0.6
8	8	fread	8	8	8	0.7
9	9	fread	9	9	9	0.8
10	10	fread	10	10	10	0.9
11	11	fread	11	11	11	1

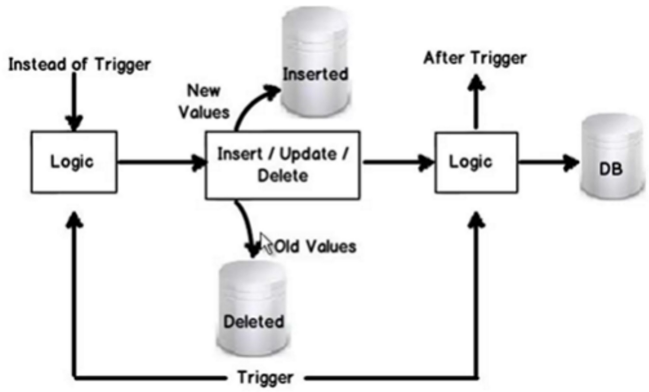
```
-- Get Analytic Functions
select id , name,
first_value(id) over (order by id) as 'first_value',
last_value(id) over (order by id ROWS BETWEEN UNBOUNDED PRECEDING  AND UNBOUNDED FOLLOWING ) as 'last_value' ,
lead(id) over (order by id) as 'lead',
lag(id) over (order by id) as 'lag'
from Employee
```

	id	name	first_value	last_value	lead	lag
1	1	Ajay	1	11	2	NULL
2	2	Vikas	1	11	3	1
3	3	Kukas	1	11	4	2
4	4	Freak	1	11	5	3
5	5	Ful	1	11	6	4
6	6	Ful	1	11	7	5
7	7	fread	1	11	8	6
8	8	fread	1	11	9	7
9	9	fread	1	11	10	8
10	10	fread	1	11	11	9
11	11	fread	1	11	NULL	10




```
select CustomerKey, GeographyKey,
 AVG(GeographyKey) OVER(ORDER BY CustomerKey ROWS BETWEEN 2 PRECEDING AND 0 FOLLOWING) as avg,
 SUM(GeographyKey) OVER(ORDER BY CustomerKey ROWS BETWEEN 2 PRECEDING AND 0 FOLLOWING) as sum,
 first_value(GeographyKey) OVER(ORDER BY CustomerKey ROWS BETWEEN 2 PRECEDING AND 0 FOLLOWING) as f_value,
 last_value(GeographyKey) OVER(ORDER BY CustomerKey ROWS BETWEEN 2 PRECEDING AND 0 FOLLOWING) as l_value
from dbo.DimCustomer
```

Triggers , inserted, deleted tables



Trigger -

```
CREATE TRIGGER trigger
  ON  triggerCustomer
  AFTER  INSERT,DELETE,UPDATE
AS
BEGIN
        Insert into triggerAudit values (GETDATE())
END
GO
```

Inserted & Deleted table

1) Temporary table created by SQL server
2) Table structure is same as original table

Query for getting trigger in SSMS

```
SELECT     o.[name],
         c.[text]
FROM       sys.objects AS o
INNER JOIN  sys.syscomments AS c
ON     o.object_id = c.id
WHERE   o.[type] = 'TR'
```

Declare variables

Inserted and deleted table

```
ALTER TRIGGER [dbo].[TriggerUpdate]
  ON [dbo].[triggerCustomer]
  AFTER INSERT,DELETE,UPDATE
AS
```

```
BEGIN

        declare @OldValue varchar(50)
        declare @NewValue varchar(50)

        --fetch old value
        select @OldValue =customerName from deleted
        --fetch new value
        select @NewValue=customerName from inserted

    -- Insert statements for trigger here
        Insert into triggerAudit(LastUpdatedDate,OldName, NewName) values (GETDATE(),@OldValue , @NewValue )
END
```

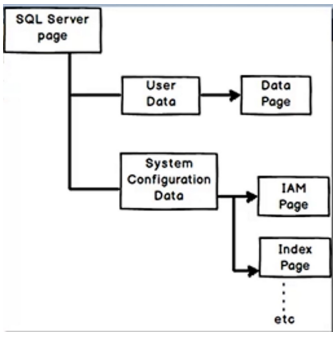
INSTEAD OF AND AFTER TRIGGER

After trigger - happens after the operation on main table
Instead of - happens before the operation on main table and it happens instead of Insert, update and delete

---

SQL Server 8 KB page

When you write or read, SQL server does it into the 8 kb page.



Data page - User data
IAM page - Index allocation map (it stores info about all pages)

To get the page list
DBCC IND('InvoiceDb',Employee, -1)

PageFID	PagePID	IAMFID	IAMPID	ObjectID	IndexID	PartitionNumber	PartitionID	iam_chain_type	PageType	IndexLevel	NextPageFID	NextPagePID	PrevPageFID	PrevPagePID
1	385	NULL	NULL	3.38E+08	0	1	7.21E+16	In-row data	10	NULL	0	0	0	0
1	400	1	385	3.38E+08	0	1	7.21E+16	In-row data	1	0	0	0	0	0

Page 10 - IAM page
Page 1 - data page

To get page details

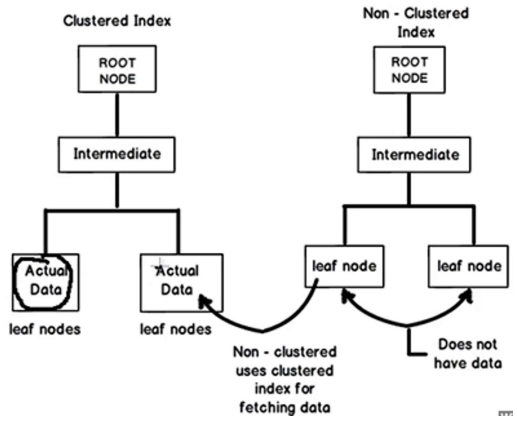
DBCC TRACEON(3604)
DBCC PAGE('InvoiceDb',1,400,1)

INDEX and Performances

Index - makes search faster by making binary tree structure. But it slows down performance of insert , update and delete

Page split and Index

Clustured and Non clustured index -



	char	nchar	varchar	nvarchar
Character Data Type	Non-Unicode fixed-length	Unicode fixed-length can store both non-Unicode and Unicode characters (i.e. Japanese, Korean etc.)	Non-Unicode variable length	Unicode variable length can store both non-Unicode and Unicode characters (i.e. Japanese, Korean etc.)
Maximum Length	up to 8,000 characters	up to 4,000 characters	up to 8,000 characters	up to 4,000 characters
Character Size	takes up 1 byte per character	takes up 2 bytes per Unicode/Non-Unicode character	takes up 1 byte per character	takes up 2 bytes per Unicode/Non-Unicode character
Storage Size	n bytes	2 times n bytes	Actual Length (in bytes)	2 times Actual Length (in bytes)
Usage	use when data length is constant or fixed length columns	use only if you need Unicode support such as the Japanese Kanji or Korean Hangul characters due to storage overhead	used when data length is variable or variable length columns and if actual data is always way less than capacity	use only if you need Unicode support such as the Japanese Kanji or Korean Hangul characters due to storage overhead
			query that uses a varchar parameter does an index seek due to column collation sets	query that uses a nvarchar parameter does an index scan due to column collation sets

Stored Procedure -

Using the cache
Centralized query

It has 3 parameters

```
CREATE procedure selectCustomer
 @CustomerName nvarchar(100),  --input parameter
 @CurrentDate Datetime output --output parameter

AS
BEGIN
 set @CurrentDate = GETDATE()
 select * from demo where name = @CustomerName
 return @@rowcount    --return parameter
END
```

___

```
declare @DatetimeCurrent datetime
declare @myrowcount int
execute @myrowcount = selectCustomer 'Vaibhav', @DatetimeCurrent output
select  @DatetimeCurrent
print @myrowcount
```

___

CTE

CTE is a temporary result set which can be used in
subsequent execution scope only once.
In CTE we can use recursive query - self referencing - employee manager hierarchy

	id	name	managerId	sales
1	1	Jitendra	0	500
2	2	Saurabh	1	100
3	3	Rahul	1	300
4	4	Shoumik	1	300
5	5	Mounika	4	220
6	6	Kalyani	3	150

For single recursion

```
select z.manager, count(1), sum(z.sales)
from
(select A.id managerId, A.name  manager, B.id, B.name , B.sales
from SaleMap A , SaleMap B  where A.id= B.managerId )Z group by z.manager
```

	manager	(No column name)	(No column name)
1	Jitendra	3	700
2	Mudit	1	500
3	Rahul	1	150
4	Shoumik	1	220

For multiple recursion

```
WITH UserCTE
    AS (SELECT id,
            name,
            managerId,
                            sales,
            0 AS EmpLevel
        FROM   SaleMap
        WHERE  managerId = 0
        UNION ALL
        SELECT usr.id,
            usr.name,
            usr.managerId,
                            usr.sales,
            mgr.[EmpLevel] + 1
        FROM   SaleMap AS usr
            INNER JOIN UserCTE AS mgr
                ON usr.managerId = mgr.id
        WHERE  usr.managerId IS NOT NULL)
SELECT *
FROM   UserCTE AS u
ORDER  BY EmpLevel;
```

Can we perform insert , update and delete in cte - yes and it will also affect the physical table

Temporary table

Temporary table are physical table which gets created for one session.
Once session is closed table is deleted.

CTE	Temporary table
Created in RAM	Created in hard disk
Lifetime - only for one subsequent execution	Lifetime - till the connection is closed
Used for recursive call	Store temp data

```
create table #temporary
(
name nvarchar(50) not null
)
```

SUBQUERY and Corelated Query

```
Select * from employee where id = (select id from salary)

select * from demo as d
where 1 = (select count(1) from demo d2 where d2.CustomerId > d.CustomerId )
```

JOIN vs Subquery

Subquery - series of filtering criterion
Join - select common part

Each query will be distributed into 60 distributions
Distribution -

Hash -using hash function - very large tables
Round_Robin - circular distribution - staging tables
Replicated - data in all nodes - small tables

Index option
Row store - transactional indexes
1. Heap - table without clustered index, no order, faster index,

Column store - analytics

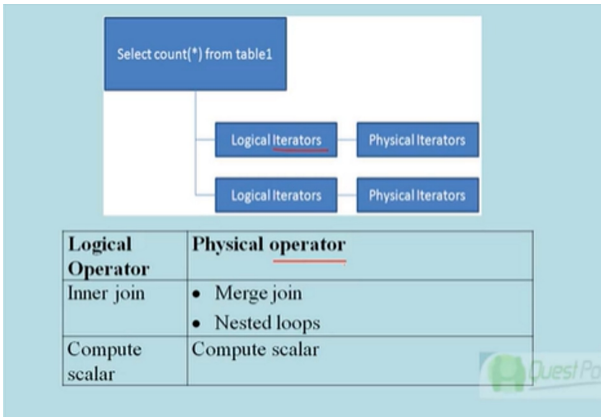
Heap - base row store
Clustured Index - Base row store maintained as B-Tree
Clustured columnstore index - base column stored

Partitions ,indexes , distributions,
Partition are created on date column - queries will generally take recent data - mini dataset

What is polybase , copy, bulk insert in adf ,

SQL Performance tuning

Iterator/ operator - logic for execution

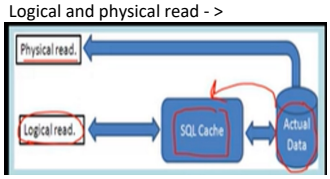


Logical Operator	Physical operator
Inner join	• Merge join • Nested loops
Compute scalar	Compute scalar

Query execution

Check syntax - > Query processor tree -> Data statistics -> Estimated plan -> execute -> Actual plan

Logical and physical read - >



Logical read - read from memory
Physical read - read from actual data

How to find logical and physical read -
set statistics io on
SELECT * from demo

1. Unique key improves table scan performance. Always create primary key on table.

2. Choose table scan for small records and seek scan for large records.
   Table scan - row wise scan
   Seek scan - B Tree scan - only with indexes

3. Use covering index to reduce row id lookup (RID)
   Both clustured and non clustured index has B Tree structure

   Clustured indexes - > leaf node point to actual data - >
   Clustered indexes physically order the data on the disk. - > primary key

   Non clustured index - > leaf node point to RID
   Heap table - table without indexes
   Use composite key
   Max non clustered index per table - 999

4. Keep index size as small as possible

   Keep pages less - >

_____

Data in SQL server is stored as tree structure based on clustered Index(key).

Table scan - if you want to get data by column other than clustered index, you need to perform whole table scan.

Index - it contains data pointer in sorted order basis of specific column..

While searching on index- we need to perform 2 operations , index search and seed search.

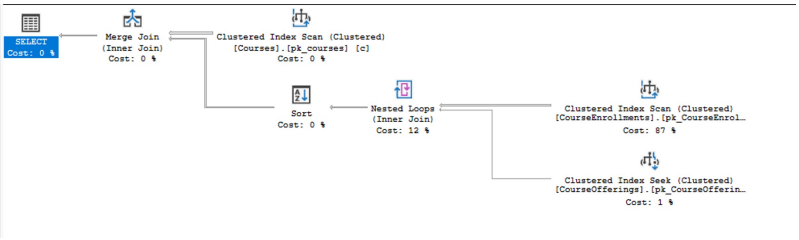
Estimated execution plan - occurs right to left and top to bottom

Query - >

```
select c.DepartmentCode, c.CourseNumber, c.CourseTitle,c.Credits,ce.grade
from CourseEnrollments ce
inner join CourseOfferings co on co.CourseOfferingId = ce.CourseOfferingId
inner join Courses c on co.DepartmentCode = c.DepartmentCode and co.CourseNumber = c.CourseNumber
```

where ce.StudentId = 29717

1. Estimated execution plan



It doesn't actually execute statement.
Read from right to left , top to bottom.

2. Get Statistics

To get detailed IO and CPU statistics about a query, need to actually execute query

set statistics io on
set statistics time on

Result
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.
Table 'CourseOfferings'. Scan count 0, logical reads 91, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.
Table 'CourseEnrollments'. Scan count 1, logical reads 12023, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.
Table 'Courses'. Scan count 1, logical reads 7, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

SQL Server Execution Times:
CPU time = 78 ms, elapsed time = 92 ms.

Logical read - reading each 8 kb page in sql server - minimize it

3. Adding an Index

   Create Index on studentId in courseEnrollments, it reduces performance drastically.

   Point of comparison
- Data access operation
- Statement cost - Subtree cost
- Logical read
- CPU time
- Elapsed time

4. Rewrite Sql statement
   Use exists, not exists to get out of matching loop.

5. Common execution plan operations

   Clustered Index scan -> Read all rows as clustered index
   Table scan -> Read all rows as heap (Normal reading)
   Clustered index Seek - traverse tree structure of table as clustered index to find matching row
   Index scan - Read all key values to find matching data
   Index seek - Traverse tree structure of index as clustered index to find matching row

   Reduce scan and increase seek

   Nested loop join -
   Merge Join -
   Hash Join -

6. Build effective indexes -

Command -

```
CREATE INDEX IX_Students_State
on Students (State);
```

Clustered index - primary key
Non Clustered index - other than primary key

Add Indexes for where clause
Primary key are indexed by default in SQL server
Index foreign key
For combination of column as index, include the most frequently used column first
Make selective index - try to get unique combination of columns with least records, preferably 1.

Hint - force SQL server to use some feature, don't use this as optimizer is already smart

LIKE - for prefix %, SQL server will not be able to use indexes as indexes are based on sorting logic. Always specify some minimum characters in LIKE search

FUNCTION operating on column value - not able to use the index, create a computed column for function

INCLUDE column - > it will stored the column values in index, creating covering index
Covering index - to reduce key lookup operation, it will get all data from index seek only

```
CREATE INDEX IX_Students_Email
on Students (Email)
INCLUDE (FIrstName, LastName) ;

select Email, FirstName, LAstName from Students
where Email = 'PaulLWilson@superrito.com'
```

7. Over Indexing
   Why not create index on every column ?
   Index has maintenance cost , for DML (insert, update, delete) it will have to update the index.
   Index are investments. Invest wisely.

   Dynamic management views

8. Finding Performance bottleneck

   SQL server is always collecting data.
   DMV - dynamic management views

   Essentials-of-Sql-Server-Performance-for-Every-Developer/Exercise Files/Module 4/Part 4/DMV.txt at master · iCodeMechanic/Essentials-of-Sql-Server-Performance-for-Every-Developer · GitHub

1. Getting Information about the sessions.

```
SELECT
   database_id,   -- SQL Server 2012 and after only
   session_id,
   status,
   login_time,
   cpu_time,
   memory_usage,
   reads,
   writes,
   logical_reads,
   host_name,
   program_name,
   host_process_id,
   client_interface_name,
   login_name as database_login_name,
   last_request_start_time
FROM sys.dm_exec_sessions
WHERE is_user_process = 1
ORDER BY cpu_time DESC;
```

2. What SQL statements are currently executing

```
-- Finding statements running in the database right now (including if a statement is blocked by another)
-- ------------------------------------------------------------------------------------------------
SELECT
      [DatabaseName] = db_name(rq.database_id),
      s.session_id,
      rq.status,
      [SqlStatement] = SUBSTRING (qt.text,rq.statement_start_offset/2,
         (CASE WHEN rq.statement_end_offset = -1 THEN LEN(CONVERT(NVARCHAR(MAX),
         qt.text)) * 2 ELSE rq.statement_end_offset END - rq.statement_start_offset)/2),
      [ClientHost] = s.host_name,
      [ClientProgram] = s.program_name,
      [ClientProcessId] = s.host_process_id,
      [SqlLoginUser] = s.login_name,
      [DurationInSeconds] = datediff(s,rq.start_time,getdate()),
      rq.start_time,
      rq.cpu_time,
      rq.logical_reads,
      rq.writes,
      [ParentStatement] = qt.text,
      p.query_plan,
      rq.wait_type,
      [BlockingSessionId] = bs.session_id,
      [BlockingHostname] = bs.host_name,
      [BlockingProgram] = bs.program_name,
      [BlockingClientProcessId] = bs.host_process_id,
      [BlockingSql] = SUBSTRING (bt.text, brq.statement_start_offset/2,
         (CASE WHEN brq.statement_end_offset = -1 THEN LEN(CONVERT(NVARCHAR(MAX),
         bt.text)) * 2 ELSE brq.statement_end_offset END - brq.statement_start_offset)/2)
   FROM sys.dm_exec_sessions s
   INNER JOIN sys.dm_exec_requests rq
      ON s.session_id = rq.session_id
   CROSS APPLY sys.dm_exec_sql_text(rq.sql_handle) as qt
   OUTER APPLY sys.dm_exec_query_plan(rq.plan_handle) p
   LEFT OUTER JOIN sys.dm_exec_sessions bs
      ON rq.blocking_session_id = bs.session_id
   LEFT OUTER JOIN sys.dm_exec_requests brq
      ON rq.blocking_session_id = brq.session_id
   OUTER APPLY sys.dm_exec_sql_text(brq.sql_handle) as bt
   WHERE s.is_user_process =1
      AND s.session_id <> @@spid
 AND rq.database_id = DB_ID()  -- Comment out to look at all databases
   ORDER BY rq.start_time ASC;
```

3. Find the slowest , Most expensive query

```
-- Finding the most expensive statements in your database
-- ------------------------------------------------------------------------------------------------
SELECT TOP 20
      DatabaseName = DB_NAME(CONVERT(int, epa.value)),
      [Execution count] = qs.execution_count,
      [CpuPerExecution] = total_worker_time / qs.execution_count ,
      [TotalCPU] = total_worker_time,
      [IOPerExecution] = (total_logical_reads + total_logical_writes) / qs.execution_count ,
      [TotalIO] = (total_logical_reads + total_logical_writes) ,
      [AverageElapsedTime] = total_elapsed_time / qs.execution_count,
      [AverageTimeBlocked] = (total_elapsed_time - total_worker_time) / qs.execution_count,
    [AverageRowsReturned] = total_rows / qs.execution_count,
    [Query Text] = SUBSTRING(qt.text,qs.statement_start_offset/2 +1,
         (CASE WHEN qs.statement_end_offset = -1
            THEN LEN(CONVERT(nvarchar(max), qt.text)) * 2
            ELSE qs.statement_end_offset end - qs.statement_start_offset)
         /2),
      [Parent Query] = qt.text,
      [Execution Plan] = p.query_plan,
    [Creation Time] = qs.creation_time,
    [Last Execution Time] = qs.last_execution_time
   FROM sys.dm_exec_query_stats qs
   CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) as qt
   OUTER APPLY sys.dm_exec_query_plan(qs.plan_handle) p
   OUTER APPLY sys.dm_exec_plan_attributes(plan_handle) AS epa
   WHERE epa.attribute = 'dbid'
      AND epa.value = db_id()
   ORDER BY [AverageElapsedTime] DESC; --Other column aliases can be used-- Finding the most expensive statements in your database
-- ------------------------------------------------------------------------------------------------
SELECT TOP 20
      DatabaseName = DB_NAME(CONVERT(int, epa.value)),
      [Execution count] = qs.execution_count,
      [CpuPerExecution] = total_worker_time / qs.execution_count ,
      [TotalCPU] = total_worker_time,
```

```
    [IOPerExecution] = (total_logical_reads + total_logical_writes) / qs.execution_count ,
    [TotalIO] = (total_logical_reads + total_logical_writes) ,
    [AverageElapsedTime] = total_elapsed_time / qs.execution_count,
    [AverageTimeBlocked] = (total_elapsed_time - total_worker_time) / qs.execution_count,
  [AverageRowsReturned] = total_rows / qs.execution_count,
  [Query Text] = SUBSTRING(qt.text,qs.statement_start_offset/2 +1,
       (CASE WHEN qs.statement_end_offset = -1
          THEN LEN(CONVERT(nvarchar(max), qt.text)) * 2
          ELSE qs.statement_end_offset end - qs.statement_start_offset)
       /2),
    [Parent Query] = qt.text,
    [Execution Plan] = p.query_plan,
  [Creation Time] = qs.creation_time,
  [Last Execution Time] = qs.last_execution_time
  FROM sys.dm_exec_query_stats qs
  CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) as qt
  OUTER APPLY sys.dm_exec_query_plan(qs.plan_handle) p
  OUTER APPLY sys.dm_exec_plan_attributes(plan_handle) AS epa
  WHERE epa.attribute = 'dbid'
    AND epa.value = db_id()
  ORDER BY [AverageElapsedTime] DESC; --Other column aliases can be used
```

4. Getting SQL Server Recommendation

```
-- Looking for Missing Indexes
-- ------------------------------------------------------------------------------------------
SELECT
   TableName = d.statement,
   d.equality_columns,
   d.inequality_columns,
   d.included_columns,
   s.user_scans,
   s.user_seeks,
   s.avg_total_user_cost,
   s.avg_user_impact,
   AverageCostSavings = ROUND(s.avg_total_user_cost * (s.avg_user_impact/100.0), 3),
   TotalCostSavings = ROUND(s.avg_total_user_cost * (s.avg_user_impact/100.0) * (s.user_seeks + s.user_scans),3)
FROM sys.dm_db_missing_index_groups g
INNER JOIN sys.dm_db_missing_index_group_stats s
   ON s.group_handle = g.index_group_handle
INNER JOIN sys.dm_db_missing_index_details d
   ON d.index_handle = g.index_handle
WHERE d.database_id = db_id()
ORDER BY TableName, TotalCostSavings DESC;
```

SQL best practices

1. Create Execution plan
2. Monitor resource usage
3. Use SQL DMV
4. Select column instead of select *
5. Avoid select Distinct
6. Avoid cartesian join
7. Avoid prefix like , use suffix like always
8. Use limit for sampling query result
9. Function in sql

Execution plan

1. Estimated execution plan - before execution
2. Actual execution plan - after execution
3. Live query statistics - on execution

You can download the execution plan in xml, graphical, text

_____

SQL server profiler - > try to get tunning recommendation

SQL server

Non clustered index takes help of clustered index to get data

Data page and index page

DBCC - database consistency checker

Type 1 - data page

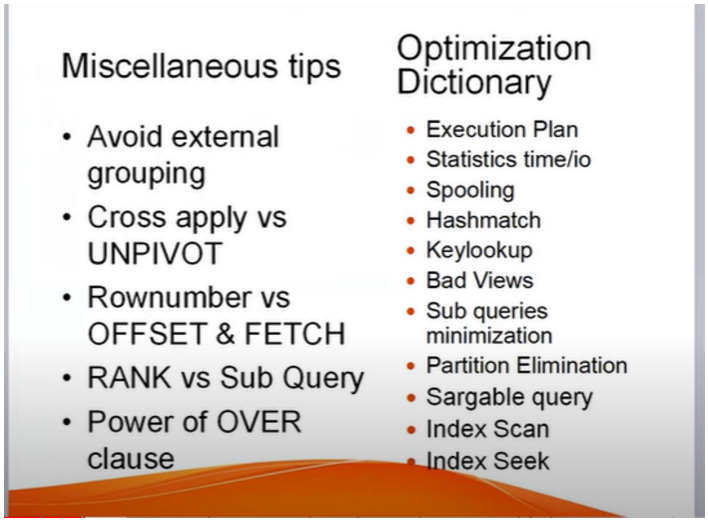
Type 2 - index page

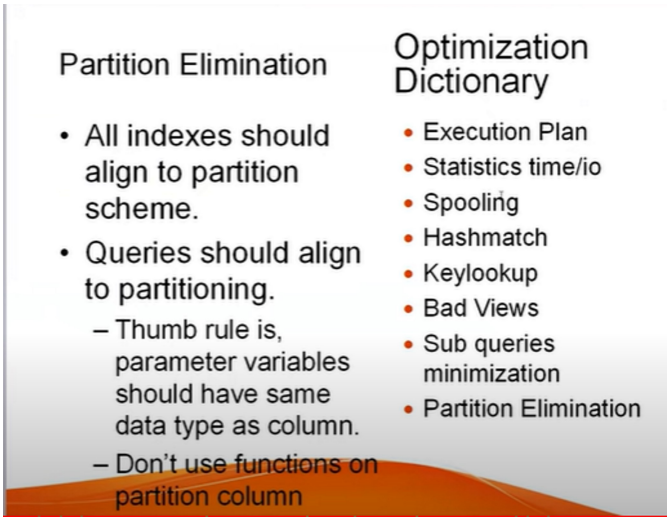
8kb page

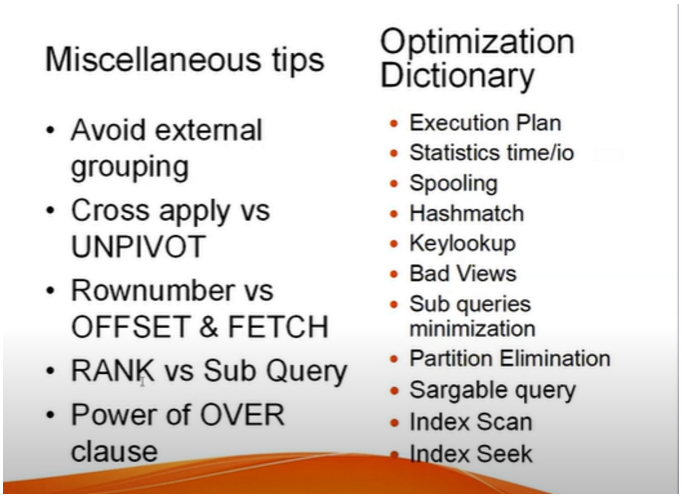
SET STATISTICS io, time on

Logical read - important -

Spooling -
lazy spooling - SQL Server will store the data ,
occurs due to duplicate aggregation , Recursive cte have lazy spooling

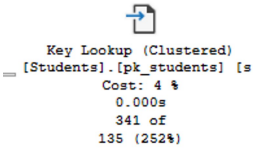






HashMatch -
Unsorted data
Group by - it will do order by first
Solution - use index, dont use function in where clause where indexed column is present

Key Lookup - due to missing column in index
Add the column to index - use include




```
CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>]
ON [dbo].[Students] ([DateOfBirth])
INCLUDE ([FirstName])
```

Data type should be same when we are comparing in sql.

Avoid function in where clause as much as possible

Avoid external group

Partition Elimination - ?
CREATE PARTITION FUNCTION

No of executions

No of rows read

Thick lines - more rows
Narrow lines - low rows

Start with thick lines
Lookup
Spool
Sort
Hash
Nested loops

Index scan -

Lookup
RID lookup (Heap) -
Key lookup (Clustered) -

Spools -
cache in query processor, implemented as hidden table in tempdb,
Table spool (Lazy)
Index spool (Eager)
Table spool (Eager)

Lack of adequate index or unique information

Sort -
Sorting is required for - merger join, order by, merge join, stream agg, window function,
Sort is expensive , use effectively

Hash
Build a hash table
Hash Match (join)
Hash Match (aggregation)

Nested loops -

Scan -
Table scan -
Index scan-
Clustered index scan -

1. Recursive cte
2. Self join
3. Join and then select in apache spark
4. CROSS APPLY , writing SP, FUNCTIONS, Table valued function,

Table valued function

```
Create function fn_getEmployeeByCity(@City varchar(50))
returns table
as
return
(
select * from Employees where  City = @City
)
```

APPLY - Join for table valued function.

“The APPLY operator is similar to the JOIN operator, but the difference is that the right-hand side operator of APPLY can reference columns from the left-hand side”.

CROSS APPLY
Ex. Find the last 3 orders from all customers that live in the UK.

```
select e1.EmployeeID,e2.FirstName from Employees e1
cross apply  fn_getEmployeeByCity(e1.city) e2
where  e1.City = e2.City
```

OUTER APPLY

```
CREATE TABLE Table1 WITH (DISTRIBUTION = HASH(c1), CLUSTERED COLUMNSTORE INDEX ORDER(c1) )
AS SELECT * FROM ExampleTable
OPTION (MAXDOP 1);
```

Check the index of table -

```
select * from sys.indexes
where object_id in (select object_id from sys.objects where name = 'logs_rapidlr')
```

Check Statistics

```
SELECT *
FROM sys.stats
WHERE object_id = OBJECT_ID('fact_loan');
```

```
DBCC SHOW_STATISTICS ( 'dbo.dim_loan'  ,IDX_DIM_LOAN_APPL_ID )
```

Check size of data

```
exec sp_spaceused 'dbo.Applicants'
```

Check SP syntax

```
----
SELECT *
FROM sys.sql_modules
WHERE object_id = (OBJECT_ID(N'dbo.uspLogError'));

------
CREATE PROC [dbo].[CD] @schema_name [varchar](500),
@tablename [varchar](500)
AS BEGIN
DECLARE @OBJECT_ID INT
SELECT
  @OBJECT_ID = OBJECT_iD(@schema_name + '.' + @tablename)
select
  *
from
  sys.objects
WHERE
  OBJECT_iD = @OBJECT_ID
select
  type_desc,
  name CCI_Name
```

```
from
  sys.indexes
where
  OBJECT_ID = @OBJECT_ID
  and type_desc in ('CLUSTERED COLUMNSTORE', 'HEAP');
with cdp as (
  select
    *
  FROM
    sys.pdw_column_distribution_properties
  WHERE
    [object_id] = OBJECT_iD(@schema_name + '.' + @tablename)
)
select
  c.name distribution_column_name
FROM
  cdp
  inner JOIN sys.columns c ON cdp.[object_id] = c.[object_id]
  AND cdp.[column_id] = c.[column_id]
where
  distribution_ordinal = 1
```

Varchar(max) - it supports character strings up to 2 GB (2,147,483,647 bytes) in length.
VARCHAR(MAX) columns cannot be included as a key column of an index.

VARCHAR(MAX) columns do not allow you to restrict the length of the column.

---

1. OFFSET AND FETCH

   This can be used only with order by clause
   OFFSET - to skip the rows of result
   FETCH - specifies the number of rows to return after the OFFSET
   Offset is mandatory while fetch is optional
   FIRST and NEXT are synonyms

   ```
   select * from Person.Person
   order by BusinessEntityID
   OFFSET 5 ROWS
   FETCH NEXT 10 ROWS ONLY
   ```

2. GROUPING SETs

3. EXISTS (subquery) - checks if the subquery returns a single or more rows
   Note that even though the subquery returns a NULL value, the EXISTS operator is still evaluated to TRUE.

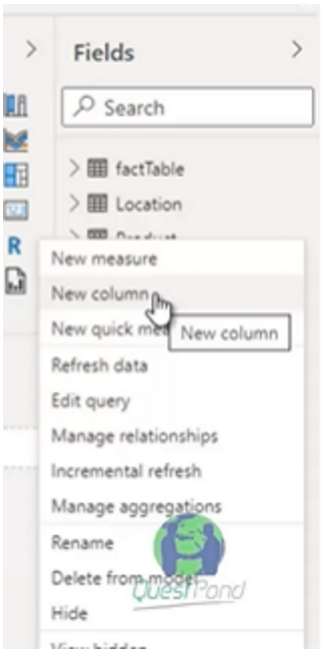
4. Distinct - Select multiple columns

5.

POWERBI

10 March 2024 02:03 PM

1. Calculated column - new column created by calculation on existing columns ,added to data model
   Ex. Concat(first,last)



2. Calculated Measure - Define number, aggregation on data table -> card, it changes based on visualization

3. Difference

Calculated Column	Measure
A new column name added in Data model which is used in formula bar	A new measure is created but not added to data model
Here we can view data in added in new column	Here we cannot view data added in measure
Evaluated for each row in our table	Evaluated when we use it in any visualization and this evaluation happens on entire dataset of table
Columns are also called as static row-level attributes	Measures are also called as dynamic aggregations
Here New Column Occupies Space in PBIX file	Here Measure does not occupy any space but yes it occupies space in cache level
Columns are useful to create relationship between tables where unique field does not exits	Here measures are useful to do only aggregation

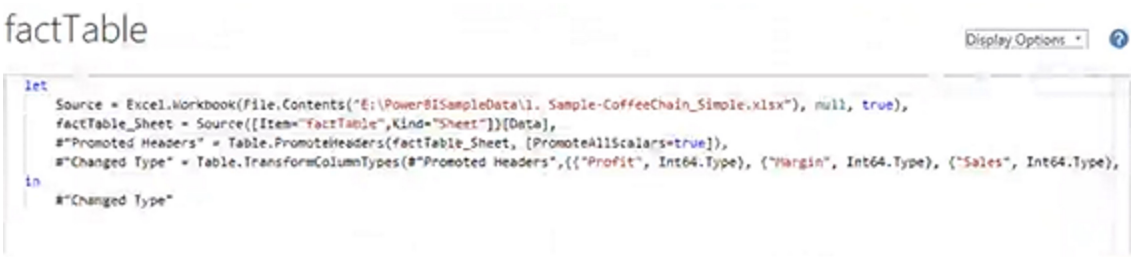
Columns are useful to create relationship between tables where unique field does not exits	Here measures are useful to do only aggregation

4. M code, M query

   Used to clean and structure the data, structure the schema

   3 important things in PBI

- Visualization - in the visualization tab
- Transformation - in power query editor -> advanced editor
- DAX -

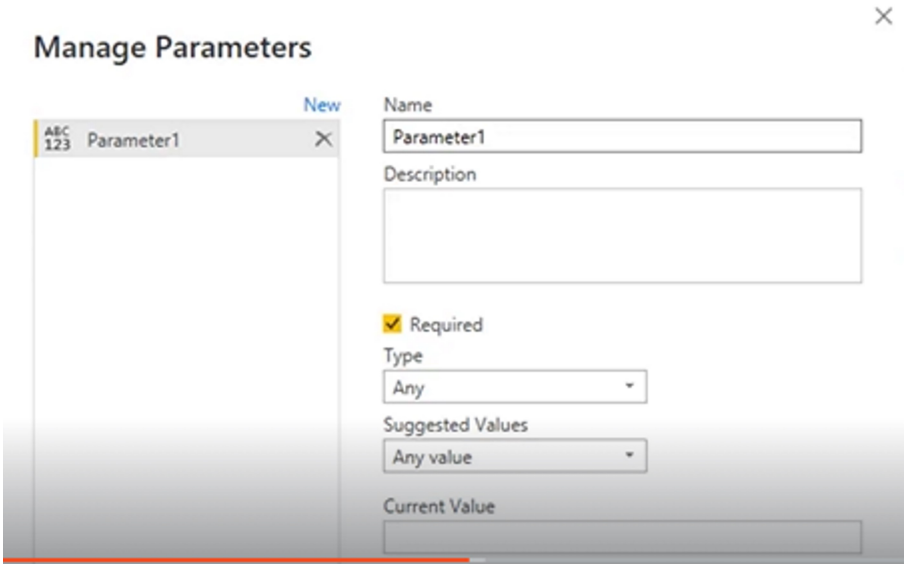


   Logical, Number, Table, Text, Date

   Custom column -

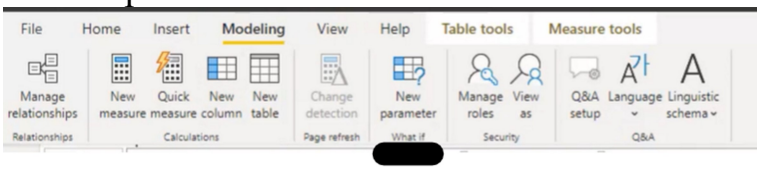
5. Parameter in PowerBI

   Query parameter -> single value or list of values



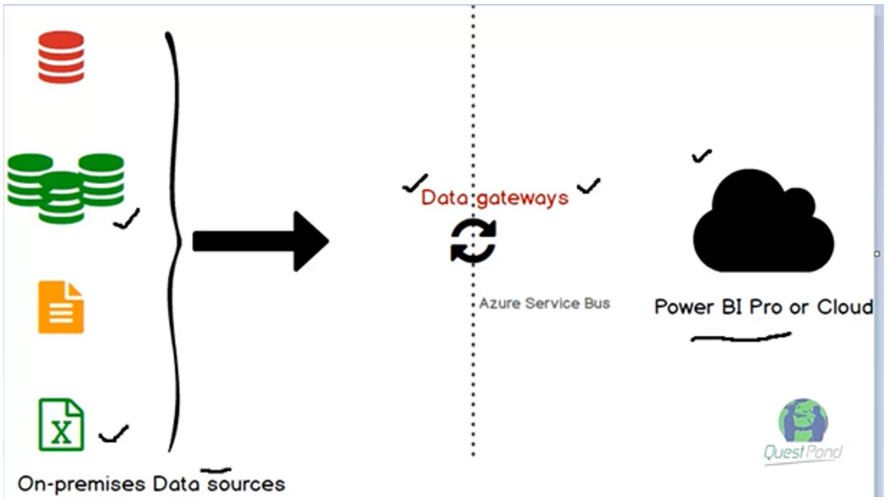
   Use- connecting to SQL server - >
   We can connect to multiple SQL server at same time through parameter

What if parameter ->



To get idea for how much growth you want

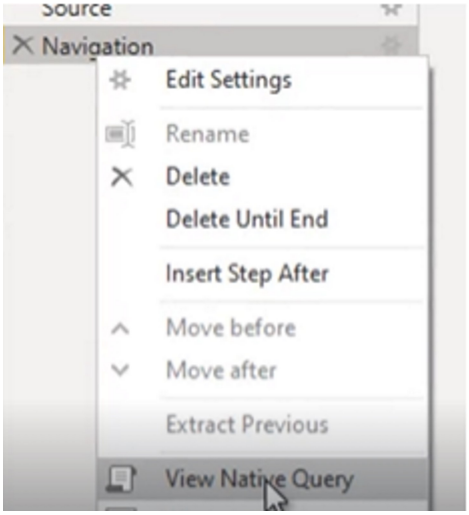
6. Data gateway in powerbi



If PBI files are on PowerBI.com and dataset is on-prem, to sync or refresh data we need PBI gateway.

7. Query Folding in PBI
   Only used in RDBMS

   Change the native query to be fired on RDBMS based on column transformation



When we select the relevant columns , the native query is changed.

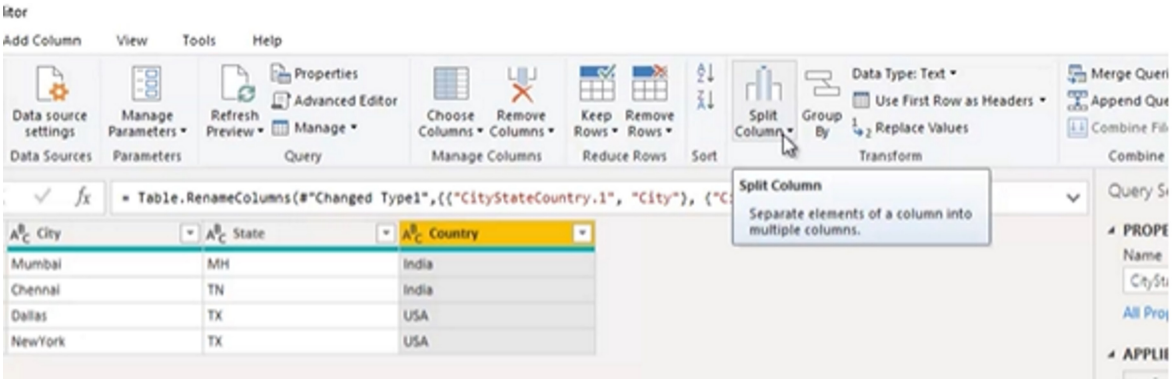
Native Query

```
select [$Table].[BusinessEntityID] as [BusinessEntityID],
    [$Table].[Title] as [Title],
    [$Table].[FirstName] as [FirstName],
    [$Table].[MiddleName] as [MiddleName],
    [$Table].[LastName] as [LastName],
    [$Table].[Suffix] as [Suffix],
    [$Table].[JobTitle] as [JobTitle],
    [$Table].[PhoneNumber] as [PhoneNumber],
    [$Table].[PhoneNumberType] as [PhoneNumberType],
    [$Table].[EmailAddress] as [EmailAddress],
    [$Table].[EmailPromotion] as [EmailPromotion],
    [$Table].[AddressLine1] as [AddressLine1],
    [$Table].[AddressLine2] as [AddressLine2],
    [$Table].[City] as [City],
    [$Table].[StateProvinceName] as [StateProvinceName],
    [$Table].[PostalCode] as [PostalCode],
    [$Table].[CountryRegionName] as [CountryRegionName],
    [$Table].[AdditionalContactInfo] as [AdditionalContactInfo]
from [HumanResources].[vEmployee] as [$Table]
```

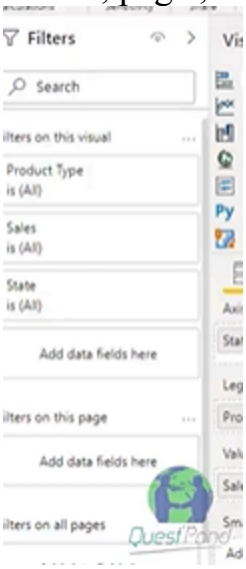
Changed query

```
select [BusinessEntityID],
    [Title],
    [FirstName],
    [MiddleName],
    [LastName]
from [HumanResources].[vEmployee] as [$Table]
```

9. Split Function in PowerBI



10. Filters
    Visual, page , report



11. ALL - Removing the filter
    ALLEXCEPT -

12. Remove rows in dataset
    In power query editor

13. Data grouping -
    In data tab in column options
    Number - Bin or list
    Text - List
    Date -

14. Distinct vs Values -
    Distinct - doesn't return blank values
    Values - return blank values

15.

.NET training - Design Pattern

12 August 2021 06:21 PM

1. What is .NET?
- It's an open source developer platform for building different types of apps.
- Developer platform - language + libraries
- Language - C#, visual basic ,F#

Platform -
.NET Core - runs on windows , Linux, MAC
.NET framework - websites, services , app
Xamarin - a .NET for mobile

C# programs run on .NET, a virtual execution system called the common language runtime (CLR) and a set of class libraries. The CLR is the implementation by Microsoft of the common language infrastructure (CLI), an international standard. The CLI is the basis for creating execution and development environments in which languages and libraries work together seamlessly.

Asynchronous programming

1) .net framework
development platform for c#, C++ and F#

2) .net core - its a cross platform for building website services and websites

3) ASP.net core - creating dynamic web pages - web applications

4) .NET 5 - next version for net core

MVC architecture - model view controller

.NET - used in windows only

IIS -
Node.js - cross platform - JS runtime
Express.js - framework

.NET core 1.0 - cross platform framework - doesn't support windows application

C#
|
Language specific compiler -
|
MSIL - intermediate code
|
Common language runtime (JIT compiler)
|
NATIVE (executable code)

.NET framework BCL - base class library

.NET core
Cross platform BCL

Cross platform RunTIME
.NET CLI tool engine - .NET CLI
ASP.net core & EF core library (App layer) - ORM - object relational mapper -

C# 10.0 - .NET 6.0 - VS 2022
There is only 1 unified .net
Portable class library -

Dll - code which we can use in multiple project - class library project
Exe -

gRPC - dual notification based architecture

ASP.net core - castrel - web server of .net - >
Reverse Proxy
Self hosted -

ASP.net core - application type
Razer - help you to build server side application

1. Blazzer - interactive client side application
2. HTTP api apps
3. Page focused web UI
4. MVC page
5. SignalR - realtime push content instantly - web socket
6. gRPC based application

Solution - it's the group of projects -

ASP .net Razor page
Middleware - all middleware start with use keyword

Asp.net web api - only model and controller no view , it return JSON

openAPI - add swagger support

gRPC call -
Client side - application was generated in client side
UTF-8 - easiest file format to transfer -

Where you are generating html - that will define the application - server or client side

.Use - middleware

Design Pattern -

Humans understand complex system by classification

1. Abstraction - classify the entity. - >Use Interfaces - >
2. Encapsulation -When we are using new methods in class which implements the Interface, use private - Encapsulate it
   Hide the unnecessary thing

3. Association, Aggregation, Composition
4. Polymorphism - ability of an entity to change the class type -> normal customer, special customer,

1. factory pattern - >
   Create 2 things identically , keep it in one place
2. Abstract factory pattern

1. Single Responsibility principle - class should only have only reason to change. Make separate classes for different functionality
2. Open closed principle - open for extension , closed for modification - >extend using interfaces
3. Liskov substitution principle - if class B is subtype of class A , we should be able to replace object of A with B without breaking behaviour of program . Subclass should extend the capability not narrow it down.
4. Interface segmented principle - Interface should be such that client should not implement unnecessary method that they don't need .
5. Dependency inversion principle

---

1. Factory Method -
The Factory Method pattern suggests that you replace direct object construction calls (using the new operator) with calls to a special factory method. Don't worry: the objects are still created via the new operator, but it's being called from within the factory method.

2. Abstract Factory
3. Builder - build classes using different classes
4. Adaptor - change format from 1 object to another.
   You can create an adapter. This is a special object that converts the interface of one object so that another object can understand it.c

Iterator pattern - Ienumerable - you can only loop

Adaptor pattern -
Abstraction - you show only what is necessary

Decouple the envoker from the receiver

Design Rule Engine

What is Dependency Injection ?
Instance is injected by framework rather than caller
IOC - any kind of unnecessary

Parallel processing- threading
Consecutive - async

Only 1 thread will execute

Lock (this){

}

Task should be independent -

In proc - Lock - when you have many threads using same code -
Out proc - when someone else is executing .exe many times - mutex, semaphoreslim ,

Task parallel library -
Thread affinity - both threads try to run on same core -

Task - don't use task

Dependency Injection -
1. By constructor
2. DI container in c# - >

Access Modifiers -

By default access modifier in C# is internal

Modifiers	Same assembly			Different assembly	
	Same class	*Derived class*	*Other class*	*Derived class*	*Other Class*
Private	Yes	No	No	No	No
Protected	Yes	Yes	No	Yes	No
Internal	Yes	Yes	Yes	No	No
Protected Internal	Yes	Yes	Yes	Yes	No
Public	Yes	Yes	Yes	Yes	Yes

Assembly in another class

C# delegate - passing method as parameter

Create an instance of service and inject it
Dont do dot counting

Law of demeter - each code should have limited amount of knowledge
From WET to DRY
Convert polydic to monodic
Change GOD method to multiple methods
Checked and unchecked exceptions
Immutability

Dependency injection - technique which helps you to inject dependent object in class.

Constructor injection -

AddScoped - insert new instance within same request
AddSingleton- same instance - shared data, caching ,
AddTransient - new instance within same request, same instance

C# - stack - variable , data , primitive data type , object pointers
Heap - object are stored

Value type -
Reference type - heap

Boxing - moving from value type to reference type - > obj I = y
Unboxing - moving from reference type to value type - int x = (int)y
Performace implication

Casting -
Int I = 0, double d = I;
Implicit - lower to higher data type
Explicit - higher to lower data type (int)
Data loss -

ArrayList - not strongly typed.

Spring boot

01 August 2021 05:02 PM

1. Day 1

HTML ->

Markup language -> information is represented through tags
information
for look and feel - HTML
store - XML

HTML

```
<HTML>
    <HEAD>
        <TITLE>My First Webpage</TITLE>
    </HEAD>
    <BODY>

        <H1>Hi!! Good Morning</H1>

    </BODY>
</HTML>
```

XML
used as a store
store passenger information in xml

```
<passengers>
<passenger>
    <name>Ram</name>
    <age>22</age>
    <gender>M</gender>
</passenger>
<passenger>
    <name>Mohan</name>
    <age>24</age>
    <gender>M</gender>
</passenger>
</passengers>
```

API -> Application Programming Interface

Java API - set of reusable functions / methods
Collection API
JDBC API

API is a set of reusable functions / methods and other tools.

- enables a developer construct application rapidly

Web page
- Weather forcasting - this service is being provided some external vendor - V1
- horoscope - this service is being provided some external vendor - V2
- educational vidoes - your service

Local application services
--> make available to others
-->convert them into API

Service -> doing something for someone

-> business function
-> provide a solution to a problem
-> reusable
-> Service is the only way to access data

-> We can interact with any application via the services

-> Service is
platform independent
language independent

-> Web Service
-> loosely coupled

Work with Web Service :-
Calculator
int sum(int a, int b)

- make this service available on web

1. Big Web Service - XML Based Webservice - SOAP
2. REST Web Service -
REST -> REpresentational State Transfer

--> architectural style of webservice
based on client server architecure
web browser is one of the clients.

any application can be client-??

client needs to access the resources of server.
how?? in what format??

Server - representation of resources
resource -> image, document, object...
format - XML, JSON, HTML, PLAIN etc
Client-Server negotiation

REST Client and Server exchange information using HTTP (Communication Protocol)

HTTP Request <----------> HTTP Response

http://localhost:8080/greeting/welcome
=>Welcome to REST

http://localhost:8080/greeting/welcome?name=Ram&city=Kashi --> query parameter
REST endpoint

http://localhost:8080/greeting/welcome/Ram --> Path parameter

https://api.openweathermap.org/data/2.5/weather? q=Bangalore,in & appid=501808b3ad6d76bfa4d593840abfde9b

Query String -> will contain query parameters

STS -> Spring Tool Suite

-> on the top eclipse

Spring Boot

- solve the problem of configuration
- can quickly create spring based projects

- not a code generation framework.

- not a server (web / application)

- provide easy way to integrate web and application servers

- by default it has embedded server - tomcat

- Spring Core
- Spring JDBC
- Spring MVC
- Spring Test
- Spring REST

.....

.....

@SpringBootApplication = @Configuration + @EnableAutoConfiguration + @ComponentScan

http://localhost:8888/sum?num1=100&num2=200

int sum(int num1, int num2){

}

---

2. Day 2

Magazine

id

name

type

price

addMagazine

getAllMagazines

getMagazineById

modifyMagazine

deleteMagazine

==========

@RequestMapping
	path
	method - GET POST PUT DELETE

GET - pass information through parameters - as part of the url

POST - pass information through request body - not a part of url

PostMan ----------> MagazineController ----------> database
table - magazine

```
class DBConnection {

	static Magazine getData(String query){
		//load driver
		// establish connection
		//create statement
		//get resultset
		//return magazine
	}

	static boolean update(String query){
		//load driver
		// establish connection
		//create statement
		//update ....

	}

}
```

==========@ Database ==========
Oracle:-

```
create table magazine (id number primary key, name varchar2(25) not null, type varchar2(15), price number(6,2));

desc magazine

select * from magazine

insert into magazine values(101, 'Vivek Jyoti', 'Monthly',18.00);

commit;
```

MySQL

```
create table magazine (id int primary key, name varchar(25) not null, type varchar(15), price decimal(6,2));
```

==========

in database
	magazine

---

3. Day 3

Spring Data JPA

JPA -> Java Persistence API

- takes care of basic operations - CRUD operations + find operations

- It is a library that adds an extra layer on the top of the JDBC

- It provides Repository
    - CrudRepository - interface
    - JpaRepository - interface

- create your own interface that should extends any Repository interface

- No need to provide implementation for the interface

@Entity
@Table("magazine")
Magazine class -------- magazine table /relation
@Id

id	id
name	name
type	type
price	price

Magazine object ---

```
Magazine magazine1 = new Magazine(100, "Lotpot", "Weekly",20.00);
Magazine magazine2 = new Magazine(101, "The Organizer", "Weekly",20.00);


repository.save(magazine1);
```

Autowiring - > It will provide the required object

```
@Autowired
JpaRepository  repository;
```

MVC

Model View Controller
model will contain data

hibernate - ORM tool

Object Relational Mapping

1. create new table - create
2. use existing table and if table doesn't exist create new table - update
3. create new table , use that table finally delete table - create-drop
4. validate existing table - wiill not create new table - validate

Steps

1. add dependencies - pom.xml
    - jpa , driver

2. add database related information in application.properties file
    - database, dburl, driver, userid, password

3. create Model class (data model class / entity class)

- class with @Entity , @Id, @Column ..... eg. Magazine, Employee, Order

4. create Repository interface
   - extends JpaRepository
     - findById(), save()
5. create Rest API Controller
   - API logic

6. Run it

```
@RequestMapping(name="/magazines", method=RequestMethod.GET)
@RequestMapping(name="/magazines", method=RequestMethod.POST)
@RequestMapping(name="/magazines", method=RequestMethod.PUT)
@RequestMapping(name="/magazines", method=RequestMethod.DELETE)
```

```
@GetMapping("/magazines")
@PostMapping("/magazines")
@PutMapping("/magazines")
@DeleteMapping("/magazines")
```

http://localhost:8888/api/magazines?id=101 -->Query Parameter

PathVariables :-

http://localhost:8888/api/magazines/101

```
@GetMapping("/magazines/{id}")
public Magazine getMagazineById( @PathVariable int id) {
    //code...
}
```

http://localhost:8888/api/magazines/101/weekly

```
@GetMapping("/magazines/{id}/{type}")
public Magazine getMagazineById( @PathVariable int id, @PathVariable String type) {
    //code...
}
```

ResponseEntity
- headers
- body
- status

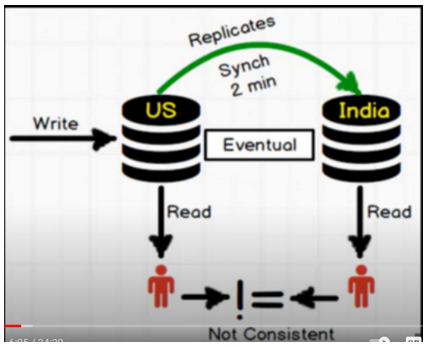
```
public ResponseEntity<Magazine>  getMagazineById(@PathVariable int id) {
    Optional<Magazine> magazineData =  magazineRepository.findById(id);

    if(magazineData.isPresent())
        return  new ResponseEntity(magazineData.get(), HttpStatus.OK);
    else
        return new ResponseEntity(HttpStatus.NOT_FOUND);
}
```

Azure COSMOSDB

13 August 2021 01:18 PM

Cosmos-DB
Planet scale , NO SQL , JSON DB with multi API support

In the planet scale database the primary database is copied into many secondary databases. Whenever any write operation is performed, It will take some time to reflect data into other databases. At this time , if some user is trying to access/read data, it will create problem of consistency.

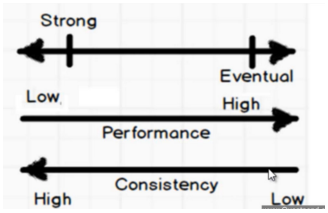


There are 2 options given to user

1. Eventual consistency - low consistency
2. Strong consistency - high consistency - the user will read the old data unless the new data gets reflected into other databases.

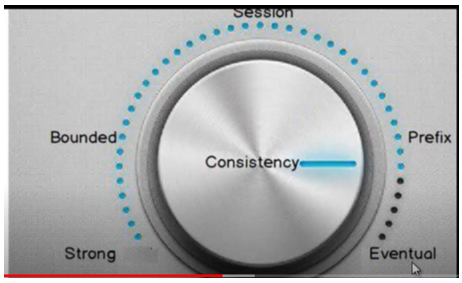
Consistency also create problem of latency

1. Latency - it will take time to reflect data into databases

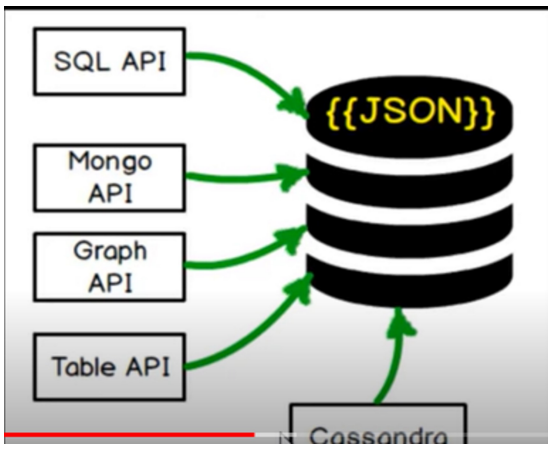


But COSMOS DB offers 3 more solutions
Prefix, session, bounded

1. Bounded staleness - we can see some data old ex. 2 hrs
2. Session - those who are writing will see immediate result
3. Prefix - the data will be read in same sequence



Multi API support - we can use any API in COSMOS DB

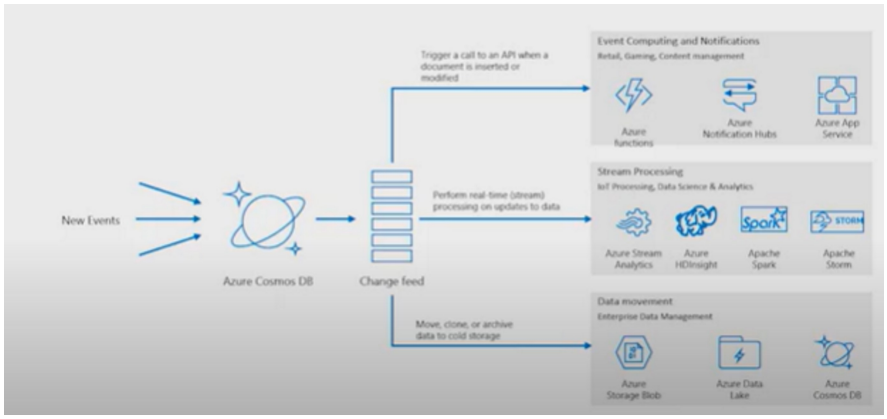


Geo- redundancy - it will enable cosmos DB on other datacentre locations

DB -> collection-> document -> {{JSON}}

For consistency -
Select the option->
Replicate data globally

Change feed - exposing cosmos DB log to end user



Azure Data explorer cache - >

Partitioning -

Logical partition - all the items with same partition key will be in one logical partition. There can be unlimited logical partition. Each partition can store up to 20 GB.

Physical partition - they are internally implemented by cosmos DB. each physical partition can have throughput of 10000 RU/s. storage for each partition is 50 GB.

Each database can have 25 container - Shared and dedicated

Azure Function

25 January 2024 03:27 PM

1. Serverless applications
2. Choice of language
3. Own dependencies
4. Use as API

You can run Azure function on various event and triggers

1. On HTTP Request
2. On Schedule Timer
3. On Document Addition in cosmos db

Small pieces of code in serverless environment -

It Requires storage account ,
consumption plan,

Application insight - database which hold all application logs

Visual studio core -

ASP.NET core

08 July 2023 12:50 PM

Create a MVC project in Visual studio

File structure
Dependency - all the dependencies
Wwwroot - static file of app
Appsettings.json - connection string / secrets
Pipeline - how application should respond to request - > In pipeline we add middleware

Browser - Pipeline [Auth - MVC - Static file]

Model - defines shape of the data
View - Represents the User interface
Controller - Interface between model and view

User - Controller - Action - Result

Routing Basics

https://localhost:12345/{Controller}/{action}/{id}

Controller and action are mandatory & id is optional

IActionResult - generic action class

AddMigration - use in code first approach to create database, tables, etc

IIS ?

New app dev

Download EF core , Sql server nuget package

Create DbContext class and connect with database

Add ApplicationDbContext service as DI

---

1. ASP.net mvc core - cross platform, performance , simplified,

2. What is MVC - architecture pattern which divides project into 3 layers
   Model - business logic
   View - UI
   Controller - binds model and views

3. What is wwwroot - we store static content like html, css, js content
4. Appsettings.json - store as configuration data, name and value,
5. How to read appsettings.json - using Iconfiguration interface belongs to configuration namespace.
   Configuration["environment:production"]

6. Explain dependency injection -
   Its the practice of providing dependent object for a class from outside rather than class creating that objects using new keyword.

   Benefits -
   When we use new object it will be tightly coupled - > DI provides Decoupling
   DbContext -

7. How to add dependency injection - Add in startup.cs file in configureservices method.

   ```
   services.AddScoped<AppContext, AppContext() > ();
   ```

   AddScoped - single object per request irrespective of DI object in method

   AddTransient - new object per every DI object in method

   AddSingleton - global object for all request entire lifecycle

8. What is middleware -
   use to execute preprocessing logic in MVC in Invoke method
   Authentication and authorization -
   - Add the custom class
   - Write the logic in Invoke
   - Pluck into configure method pipeline using app.useMiddleware

9. What is startup.cs
   ConfigureServices - for DI config- services.add
   Configure - for Middleware config - app.useMiddleware

10. Razor page(cshtml) - view Engine - helps to write C# and HTML code together

11. viewModel - wrapper class containing multiple class - data what your view needs
12. Kestrel - free open source cross platform default web server ships with asp.net core
13. Reverse proxy - Request -> IIS -> Kestrel -> MVC and come back
14. Cookies - text files store information in website.
15. Session management - maintain states between http requests.
    Session
    ViewData/bag
    tempData
16.

C# tutorials

23 August 2021 05:29 PM

Using - imports namespace
Namespace - collection of classes

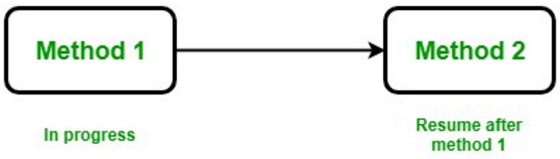
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
   class Program
   {
static void Main(string[] args)
{
Console.WriteLine("Hello, world!");
Console.ReadLine();
}
   }
}
```

---

1. Multithreading in C#

Each process in OS is a lightweight thread

1. Single threading - mainv thread execute all the process one after another -




```
using System;
using System.Threading;

public class Geek {

    // static method one
    public static void method1()
    {

        // It prints numbers from 0 to 10
        for (int I = 0; I <= 10; I++) {

            Console.WriteLine("Method1 is : {0}", I);

            // When the value of I is equal to
            // 5 then this method sleeps for
            // 6 seconds and after 6 seconds
            // it resumes its working
            if (I == 5) {
                Thread.Sleep(6000);
            }
        }
    }

    // static method two
    public static void method2()
    {

        // It prints numbers from 0 to 10
        for (int J = 0; J <= 10; J++) {

            Console.WriteLine("Method2 is : {0}", J);
        }
    }
}

// Driver Class
```

```
public class GFG {

    // Main Method
    static public void Main()
    {

        // Calling static methods
        Geek.method1();
        Geek.method2();
    }
}
```

From <https://www.geeksforgeeks.org/c-sharp-multithreading/>

2. Multithreading - Multi-threading is a process that contains multiple threads within a single process. Here each thread performs different activities. For example, we have a class and this call contains two different methods, now using multithreading each method is executed by a separate thread. So the major advantage of multithreading is it works simultaneously, which means multiple tasks execute at the same time. And also maximizing the utilization of the CPU because multithreading works on time-sharing concept mean each thread takes its own time for execution and does not affect the execution of another thread, this time interval is given by the operating system.


```
// C# program to illustrate the
// concept of multithreading
using System;
using System.Threading;

public class GFG {

    // static method one
    public static void method1()
    {

        // It prints numbers from 0 to 10
        for (int I = 0; I <= 10; I++) {
            Console.WriteLine("Method1 is : {0}", I);

            // When the value of I is equal to 5 then
            // this method sleeps for 6 seconds
            if (I == 5) {
                Thread.Sleep(6000);
            }
        }
    }

    // static method two
    public static void method2()
    {
        // It prints numbers from 0 to 10
        for (int J = 0; J <= 10; J++) {
            Console.WriteLine("Method2 is : {0}", J);
        }
    }

    // Main Method
    static public void Main()
    {

        // Creating and initializing threads
        Thread thr1 = new Thread(method1);
        Thread thr2 = new Thread(method2);
        thr1.Start();
        thr2.Start();
    }
}
```

2. Asynchronous programming in C#

We can run all the methods parallelly by using simple thread programming, but it will block UI and wait to complete all the tasks. To come out of this problem, we have to write too many codes in traditional programming, but if we use the async and await keywords, we will get the solutions in much less code.

For asynchronous operation

1. All task will execute asynchronously

```
Task.run(()=>{
Calculate1()
}));
Task.run(()=>{
Calculate2()
}));
Task.run(()=>{
Calculate3()
}));
```

2. If task 3 is dependent on 1 and 2

```
Var task1 = Task.run(()=>{
Calculate1()
}));
Var task2 = Task.run(()=>{
Calculate2()
}));
Task. waitAll(task1, task2);

Var awaiter 1= task1.GetAwaiter();
Var awaiter2 = task2.GetAwaiter();
Var result1 = awaiter1.GetResult();
Var result2 = awaiter2.GetResult();

Calculate3(result1,result2);
```

---

3. Basic tutorials

1. Variable types

a. Value type - int, float, double , long
b. Reference type - object(type checking at compile time), dynamic(type checking at run time), string
c. Pointer type -

If else - x>5? y=3: y=5
Nullable
Bool? A = we can assign 3 values true, false or null

Multitasking - OS is able to perform multiple task at same time .
Process - responsible for executing application
Thread - lightweight process - which executes the code under application
Every application has minimum 1 thread - main thread

every program is by default single threaded application - sequential execution

Thread t = Thread.CurrentThread; - will return the current executing thread.

Every method in the class will be executed step by step .
Thread.Sleep(5000) - 5 sec

Multithreading in C#
A process can have more than 1 thread. Max utilization in CPU
OS will allocate particular time for each method execution. - time sharing
T1 => 2 2 2
T2=> 2 2 2
T3 =>2 2 2

Thread safety - lock, monitor : - mutex, semaphore slim , semaphore
Thread debug

_____---

Asynchronous programming patter
APM /EAP/TAP - Recent

Asynchronous programming -
From blocking to non-blocking, not blocking UI ->

Await Task - return nothing

Multi-tasking - doing multiple operations simultaneously.

Async - consecutive processing

Concurrency - 2 task on same core - (it gives some time task 1 and some time to task 2) - time sharing
parallelism - parallel execution on 2 different cores -

Concurrency - to make application usable , it shouldn't hang, non-blocking , cant determine what is output
Parallelism - performance

When you make them concurrent , then you have an advantage to make it parallel
But they shouldn't talk each other.

TPL automatically uses multi core , handles the core
Task uses the thread internally
Thread does concurrently

TPL - doesn't use context switching   - use TPL
TPL

_____

Attributes in C#
Used for providing metadata to the class - used in reflection .
When we want to show some warning about class, method to user , then u can use attribute.

Reflection in C#
Late binding - Used by IDE to show the properties of class.

Properties in C#
Properties are an extension of fields and are accessed using the same syntax. They use accessors through which the values of the private fields can be read, written or manipulated.

Delegates in C#
A delegate is an object which refers to a method. For example, if you click on a Button on a form (Windows Form application), the program would call a specific method. In simple words, it is a type that represents references to methods with a particular parameter list and return

type and then calls the method in a program for execution when it is needed.
Used in parallel processing application - > for callbacks
Parallel processing is prerequisite for the delegate. 2 parallel processes are happening and the communication between them happens through delegate. It can make your code safe.

Thread - > parallel processing - > delegates - > broadcast -> events

Multicasting of delegates - > broadcasting
DelegateTest d = new DelegateTest();
d.sender += Receiver;
d.sender += Receiver1;
d.sender += Receiver2;
In broadcasting the subscriber has power to change publisher. It can manipulate the delegate
d.sender = null //

To avoid this issue - > **we can use events** - > events will make the pub sub model. (observable model)
Event is an encapsulation over delegate

Thread safe - when two threads are trying to access same objects.

Events in C#
A publisher is an object that contains the definition of the event and the delegate. The event-delegate association is also defined in this object. A publisher class object invokes the event and it is notified to other objects.

A subscriber is an object that accepts the event and provides an event handler. The delegate in the publisher class invokes the method (event handler) of the subscriber class.

Generics in C#
generics allow you to write a class or method that can work with any data type.

Azure & Data Basics

02 September 2021 05:25 PM

1. **Types of data**

Structured - relational - Azure SQL DB
Semi- structured - no relational - Azure cosmos DB
Unstructured - azure blob storage

2. **Data processing**

Transactional processing -
similar to banking transaction.
{
A transaction is a sequence of operations that are atomic. This means that either all operations in the sequence must be completed successfully, or if something goes wrong, all operations run so far in the sequence must be undone. Bank transfers are a good example; you deduct funds from one account and credit the equivalent funds to another account. If the system fails after deducting the funds, they must be reinstated in the original account (they mustn't be lost). You can then attempt to perform the transfer again. Similarly, you shouldn't be able to credit an account twice with the same funds.
}
Transactional systems are often high-volume, sometimes handling many millions of transactions in a single day. The data being processed has to be accessible very quickly. The work performed by transactional systems is often referred to as Online Transactional Processing (OLTP).
For faster processing during transaction, only those columns which are necessary for bank transfer are taken.
Splitting tables out into separate groups of columns like this is called normalization.
While normalization enables fast throughput for transactions, it can make querying more complex. Queries involving normalized tables will frequently need to join the data held across several tables back together again. This can make it difficult for business users who might need to examine the data.

2. Analytical system -

In contrast to systems designed to support OLTP, an analytical system is designed to support business users who need to query data and gain a big picture view of the information held in a database. They are concerned with capturing data and doing operations on it.
Most analytical data processing systems need to perform similar tasks: data ingestion, data transformation, data querying, and data visualization. The image below illustrates the components in a typical data processing system.

Database must adhere to ACID
ACID (Atomicity, Consistency, Isolation, Durability)

1. Atomicity - each transaction is treated as single unit (Roll back if its incomplete)
2. Consistency - ensures that a transaction can only take the data in the database from one valid state to another.
   The sum of money before and after transaction is same.
3. Isolation -This property ensures that multiple transactions can occur concurrently without leading to the inconsistency of database state
4. Durability - All the changes in database are permanent.

Processing data as it arrives is called ***streaming***. Buffering and processing the data in groups is called **batch processing.**

1. **Streaming**
2. **Batch processing**

You typically use batch processing for performing complex analytics. Stream processing is used for simple response functions, aggregates, or calculations such as rolling averages.

---

2. Explore roles and responsibilities in the world of data Review tasks and tools for database administration

1. Database administrator- >Database Administrators are tasked with managing and organizing databases. A database administrator's primary job is to ensure that data is available, protected from loss, corruption, or theft, and is easily accessible as needed.

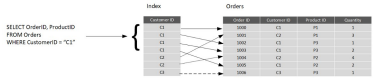
SSMS -SQL Server Management Studio provides a graphical interface, enabling you to query data, perform general database administration tasks, and generate scripts for automating database maintenance and support operations. The example below shows SQL Server Management Studio being used to back up a database.

2. Data engineer -
3. Data analyst

Relational DB
Primary key -
Foreign key - they are primary key for another table

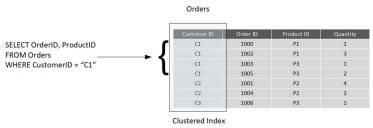
Relational Data structures

1. **Index**- > you specify a column from the table, and the index contains a copy of this data in a sorted order with pointers to the corresponding rows in the table. Whenever we query the database , the process become easy as the indexes are already sorted .
   We can many indexes on a table



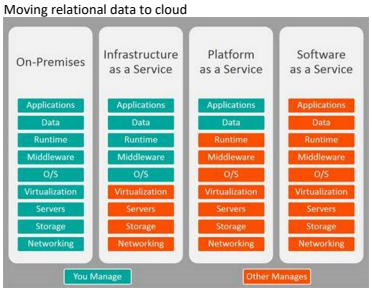
Disadvantages -
- they are not free. Incur additional processing charge
  - An index might consume additional storage space, and each time you insert, update, or delete data in a table, the indexes for that table must be maintained.

2. **Clustered indexes** -A clustered index physically reorganizes a table by the index key. We can have only 1 clustered index

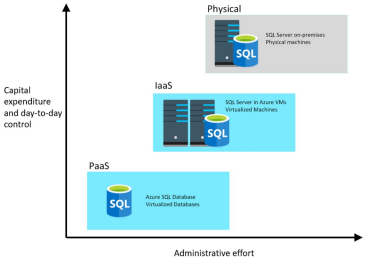


1. **View** - >A view is a virtual table based on the result set of a query. In the simplest case, you can think of a view as a window on specified rows in an underlying table.
   CREATE VIEW P1Orders AS
   SELECT CustomerID, OrderID, Quantity
   FROM Orders
   WHERE ProductID = "P1

Moving relational data to cloud



IAAS - You can create a set of virtual machines, connect them together using a virtual network, and add a range of virtual devices.
PAAS - Azure SQL Database, Azure Database for PostgreSQL, Azure Database for MySQL, and Azure Database for MariaDB.



==========

Non- relational databases

-A key aspect of non-relational databases is that they enable you to store data in a very flexible manner. Non-relational databases don't impose a schema on data. Instead, they focus on the data itself rather than how to structure it. This approach means that you can store information in a natural format, that mirrors the way in which you would consume, query and use it.

- Each entity should have a unique key. The entities in a collection are usually stored in key-value order.
- Queries can then use the index to identify and fetch data based on non-key fields.

Types of non- relational data

1. Semi structured -
   Most used - JSON
   Avro-
   ORC -
   Parquet -

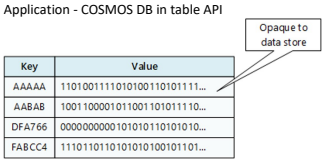
2. Unstructured -

Unstructured data is data that doesn't naturally contain fields. Examples include video, audio, and other media streams. Each item is an amorphous blob(binary large object) of binary data. You can't search for specific elements in this data.

NoSQL -
Not only SQL - DBMS for non-relational data
NOSQL databases generally fall into 4 categories.
Key value , document store , cloud family , graph

1. **Key value** -

The values in key value pair are stored in opaque format
The focus of a key-value store is the ability to read and write data very quickly. Search capabilities are secondary. A key-value store is an excellent choice for data ingestion, when a large volume of data arrives as a continual stream and must be stored immediately.
Application - COSMOS DB in table API



Key	Value
AAAAA	1101001111010100110101111...
AABAB	1001100001011001101011110...
DFA766	0000000000101010110101010...
FABCC4	1110110110101010100101101...

2. **Document store**-

It's similar to key value store but the values are transparent to DB

Key	Document
1001	{ "CustomerID": 99, "OrderItems": [{ "ProductID": 2010, "Quantity": 2, "Cost": 520 }, { "ProductID": 4365, "Quantity": 1, "Cost": 18 }], "OrderDate": "04/01/2017" }
1002	{ "CustomerID": 220, "OrderItems": [{ "ProductID": 1285, "Quantity": 1, "Cost": 120 }], "OrderDate": "05/08/2017" }

Application - Cosmos DB in SQL API

3. **Column family base -**

Application - Cosmos DB in Cassandra

4. **Graph DB**

Application - Cosmos DB in Gremlin API

Concepts of data analysis-
Three main concepts

1. Data ingestion
2. Data processing
3. Data exploration

ETL and Load

Az 900 questions

1) IAAS , PAAS, SAAS

IAAS - Azure storage account (blob, file ,queue , table), Azure VM, Azure DNS server
PAAS - Azure app service, Azure SQL Database , Azure Backup ,
SAAS - Intune , no software update required,

2)
Support plan

Basic - No support ticket
Developer
Standard -
Professional direct
Premier support plan

Azure web app plan

Usage Tiers	FreeTry for free	SharedEnvironment for dev/test	BasicDedicated environment for dev/test	StandardRun production workloads	PremiumEnhanced performance and scale	IsolatedHigh-Performance, Security and Isolation
Web, mobile or API apps	10	100	Unlimited	Unlimited	Unlimited	Unlimited
Disk space	1 GB	1 GB	10 GB	50 GB	250 GB	1 TB
Maximum instances	–	–	Up to 3	Up to 10	Up to 30*	Up to 100
Custom domain	–	Supported	Supported	Supported	Supported	Supported
Auto Scale	–	–	–	Supported	Supported	Supported
Hybrid Connectivity	–	–	Supported	Supported	Supported	Supported
Virtual Network Connectivity	–	–	Supported	Supported	Supported	Supported
Private Endpoints	–	–	Supported	Supported	Supported	Supported
Compute Type	Shared	Shared	Dedicated	Dedicated	Dedicated	Isolated
Pay as you go price	Free	₹903.219/month	₹4,964.709/month	₹6,579.734/month	₹13,159.468/month	₹26,988.872/month

3) Azure Data Redundancy

https://learn.microsoft.com/en-us/azure/storage/common/storage-redundancy
For primary
LRS - 3 copy in same Data centre

ZRS - 3 copy in 3 availability zone(Data centre) in same region

For Secondary
GRS - 3 copy in same Data center in primary in and 3 copy in same Data center in secondary
GZRS - 3 copy in 3 availability zone in primary in and 3 copy in same Data center in secondary

RA -GRS / RA -GZRS - having read only access to secondary

Azure storage account
Blob - for unstructured data
file - mount network drive to computer
queue - for messages
table - for NoSQL

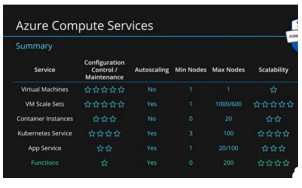
Tiers
Online - Hot , Cool -
Offline - Archive - data cannot be accessed , it must be rehydrated

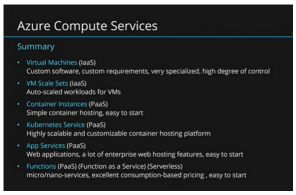
	Hot tier	Cool tier	Cold tier (preview)	Archive tier
Availability	99.9%	99%	99%	99%
Availability (RA-GRS reads)	99.99%	99.9%	99.9%	99.9%
Usage charges	Higher storage costs, but lower access and transaction costs	Lower storage costs, but higher access and transaction costs	Lower storage costs, but higher access and transaction costs	Lowest storage costs, but highest access, and transaction costs
Minimum recommended data retention period	N/A	30 days[1]	90 days[1]	180 days
Latency (Time to first byte)	Milliseconds	Milliseconds	Milliseconds	Hours[2]
Supported redundancy configurations	All	All	All	LRS, GRS, and RA-GRS[3] only

---

Azure compute services

VM - virtualization of physical Machine - IAAS
Virtual machine -
VMSS - many VM sharing load by load balancer - scalable
Container Instances - dont have OS, lightweight , Container group - similar to VM - > for simple deployment
AKS - highly scalable -
App service - simple service -
Azure function - Serverless





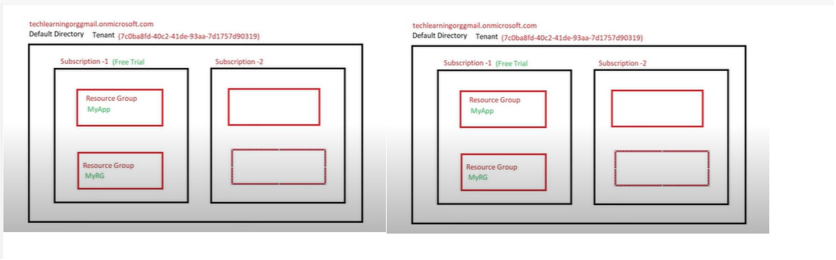
Azure networking services

One Virtual network can be taken in one region
For connection of multiple VN - Vnet pairing & VPN gateway

Microsoft managed desktop -
Azure Devtest labs - for testing many VMs

Azure Active directory -
One stop solution for managing all the users across different resources

Active directory - we can have multiple AAD instances in same active directory
AAD instance (tenant) - subscription - Resource group - resources



Azure management Group -

1. Used for logical grouping of Subscriptions
2. you can define policy at management group level.
3. When we create azure subscription, it will by default create one management group - Root , you can create sub group in management group
4. 10000 management group and up to 6th hierarchy in one tenent

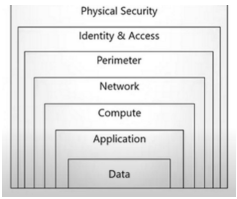
1. Read only lock & Delete lock in Azure Resource level

2. All resources inherit permission from azure resource group

Azure networking
Two resources in same VN can connect to each other

Azure Shell
Any os can be accessed by - Azure CLI, Azure Powershell, Azure Portal

Azure Security, Privacy, Compliance and Trust



Security services -
Azure Firewall - to protect the Virtual network , unrestricted scalability , can work in multiple azure subscriptions
Application rule - access to domain name ,ie - www.google.com
Network rule - access to IP and Port

Azure DDOS - Distributed Denial of service - to stop unwanted traffic

Network security group - to protect subnet and vm
Inbound -
Outbound -

Application security group - when you want create specific rule for a group of VM. You can create rule for a group and configure in NSG

Identity services -
AAD - Single sign on , Create users and groups , can secure both Azure resources and Microsoft 365. provides access tokens
MFA -Password, Microsoft Authenticator App, SMS and Voice call. Only Windows 10 devices.
sign-in risk policy - anonymous IP detection
user risk policy.

Security tools -

Azure security - security recommendation , regulatory recommendation, can monitor both cloud and on premise
Azure Key Vault - for storing secrets , passwords and certificates.
Azure Information Protection - protect documents , emails
Azure Advanced thread protection - for threats

Azure Governance -

Azure Policy - Control user action -
Initiative Definition
Initiative Assignment
If there are any existing resources that aren't compliant with a new policy assignment, they appear under Non compliant resources.

Azure RBAC -

Azure Resource Lock - cannotdelete and readonly

Azure Service Health -
Azure Status
Azure Service Health
Azure Resource Health

Azure Traffic manager -a DNS-based load balancing solution.

Azure Government - a cloud environment specifically built to meet compliance and security requirements for US government. This mission-critical cloud delivers breakthrough innovation to U.S. government customers and their partners. Azure Government applies to government at any level — from state and local governments to federal agencies including Department of Defense agencies.

Azure Activity log -

Azure portals -
Powershell - download and use - cross platform
Azure CLI - download and use - cross platform -
Azure cloud shell - no download - in platform -

Data ingress - we are charged
Data Egress - we are not charged
Data traffic between azure services within same region is always free

You cannot open support cases in the Basic support plan

When Azure trial is finished you can still access
You are not charged for Azure Active Directory user accounts so you can continue to create accounts.
You can access data that is already stored in Azure.
You can access the Azure Portal. You can also reactivate and upgrade the expired subscription in the portal.

With Azure ExpressRoute, all inbound data transfer is free of charge.

Management groups -
Azure Virtual Desktop
Azure files - Network File System (NFS),

defense in depth - uses a series of mechanisms to slow the advancement of an attack that aims to gain unauthorized access to data?
Azure CLI allows you to use the Bash shell to perform administrative tasks.

Azure Advisor - recommendation on best practices, compliances , security, action items

Azure VM - 99.99 - 2 vm and 2 Availability zone

In PAAS we dont install anything

VPN -

Azure Databricks

18 March 2023 06:25 PM

Big Data - Huge unstructured data
Databricks - apache spark technology
Workspace - collaborative place - environment to access your assets
Notebooks, libraries, experiments

Azure databricks service - paid
Community edition - free account

Clustor - computation resources
Notebook - series of commands
Job - mechanism for running requests
]
Library - 3rd party code

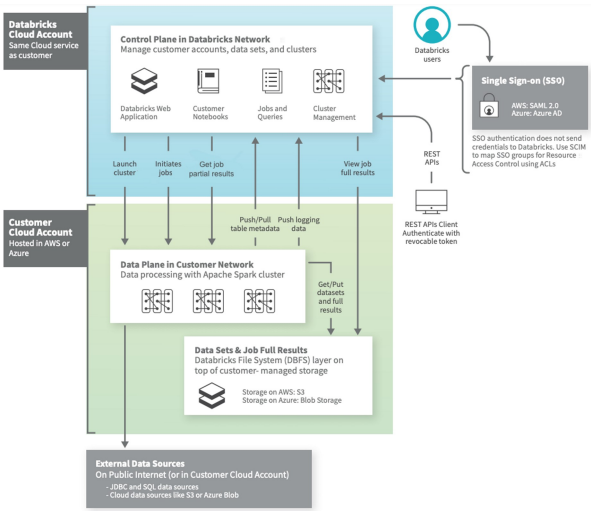
Folders -
Workspace - root folder, contains - Shared and users folder

Control plane - Databricks Account
WEB UI
Cluster Management
Workflows
Notebooks

Data plane - Customer Cloud Account
Cluster VM
Storage

To create comment in table -
COMMENT 'Contains PII'

DBFS - own storage layer of databricks. It's a blob storage in data plane (your own account)



Dbutils - library

Fs
Copy file -dbutils.fs.cp('dbfs:/FileStore/tables/employeeData.csv','dbfs:/FileStore/tables2/employeeData2.csv')

Notebook.exit() - exit with a value

Spark is built on scala language

= easy = medium

Essential Core & Intermediate Spark Operations

TRANSFORMATIONS

General	Math / Statistical	Set Theory / Relational	Data Structure / I/O
• map	• sample	• union	• keyBy
• filter	• randomSplit	• intersection	• zipWithIndex
• flatMap		• subtract	• zipWithUniqueID
• mapPartitions		• distinct	• zipPartitions
• mapPartitionsWithIndex		• cartesian	• coalesce
• groupBy		• zip	• repartition
• sortBy			• repartitionAndSortWithinPartitions
			• pipe

ACTIONS

• reduce	• count	• takeOrdered	• saveAsTextFile
• collect	• takeSample		• saveAsSequenceFile
• aggregate	• max		• saveAsObjectFile
• fold	• min		• saveAsHadoopDataset
• first	• sum		• saveAsHadoopFile
• take	• histogram		• saveAsNewAPIHadoopDataset
• forEach	• mean		• saveAsNewAPIHadoopFile
• top	• variance		
• treeAggregate	• stdev		
• treeReduce	• sampleVariance		
• forEachPartition	• countApprox		
• collectAsMap	• countApproxDistinct		

1. Show() - show the dataframe
2. Select() - select a column
3. Collect() - Action - Returns array of rows to the driver node

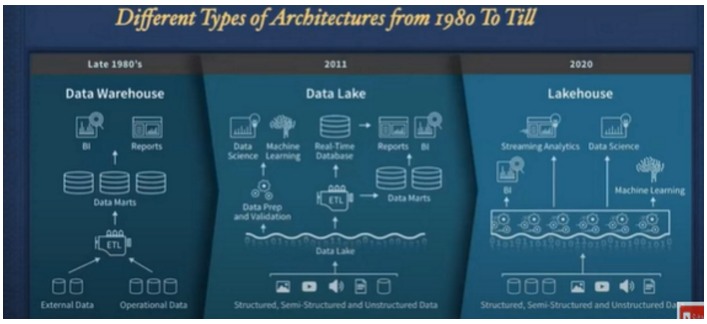
---

Lakehouse

Lakehouse - Delta lake + Unity Catalog

Delta lake - All structured, unstructured and semi structured data will be converted into delta format and all operations will be done on delta tables.

Unity Catalog ensures that you have complete control over who gains access to which data and provides a centralized mechanism for managing all data governance and access controls without needing to replicate your data.



Data Warehouse	Data Lake	Delta Lake
Only Structured Data	Structured/Semi-structured/Unstructured	Structured/Semi-structured/Unstructured/Streaming
Schema-on-Write	Schema-on-Read	Schema-on-Read
Supports ACID Transaction	Minimal support to ACID Transactions	Supports ACID Transaction
Does not corrupt the system	Leaves system in corrupted state	Does not corrupt the system

Each delta table has 1 parquet file and 3 Log files for every transaction -

Parquet file - data is stored in parquet format. If you are deleting data , it wont be deleted immediately. It remains for certain duration and then gets deleted.

Log

1. Json transactional log file - for every transaction one log file will be created.

JSON Log file

```
%fs
head /FileStore/delta/arch_demo/_delta_log/00000000000000000001.json
```

{"commitInfo":{"timestamp":1695109045750,"userId":"3061398064118290","userName":"saurabh.aher@outlook.com","operation":"WRITE","operationParameters":{"mode":"Append","partitionBy":"[]"},"notebook":{"notebookId":"1039915806960769"},"clusterId":"0919-071700-j3lojp9q","readVersion":0,"isolationLevel":"WriteSerializable","isBlindAppend":true,"operationMetrics":{"numFiles":"1","numOutputRows":"1","numOutputBytes":"1521"},"engineInfo":"Databricks-Runtime/12.2.x-scala2.12","txnId":"05e10a73-e754-4d4a-b5e2-ce6f43dc7900"}}
{"add":{"path":"part-00000-90b4b7bc-c7e1-488a-8473-0bee25f48695-c000.snappy.parquet","partitionValues":{},"size":1521,"modificationTime":1695109045000,"dataChange":true,"stats":"{\"numRecords\":1,\"minValues\":{\"emp_id\":100,\"emp_name\":\"Stephen\",\"gender\":\"M\",\"salary\":2000,\"Dept\":\"IT\"},\"maxValues\":{\"emp_id\":100,\"emp_name\":\"Stephen\",\"gender\":\"M\",\"salary\":2000,\"Dept\":\"IT\"},\"nullCount\":{\"emp_id\":0,\"emp_name\":0,\"gender\":0,\"salary\":0,\"Dept\":0}}","tags":{"INSERTION_TIME":"1695109045000000","MIN_INSERTION_TIME":"1695109045000000","MAX_INSERTION_TIME":"1695109045000000","OPTIMIZE_TARGET_SIZE":"268435456"}}}

Command took 1.14 seconds -- by saurabh.aher@outlook.com at 9/19/2023, 1:08:23 PM on DemoCluster

1. Parquet checkpoint file - for every 10 transaction log file, one checkpoint file

Table

	txn	add	remove
8		"MIN_INSERTION_TIME": "1695109668000000", "MAX_INSERTION_TIME": "1695109668000000", "OPTIMIZE_TARGET_SIZE": "268435456"}, "deletionVector": null, "baseRowId": null, "stats": "{\"numRecords\":1,\"minValues\":{\"emp_id\":100,\"emp_name\":\"Stephen\",\"gender\":\"M\",\"salary\":2000,\"Dept\":\"IT\"},\"maxValues\":{\"emp_id\":100,\"emp_name\":\"Stephen\",\"gender\":\"M\",\"salary\":2000,\"Dept\":\"IT\"},\"nullCount\":{\"emp_id\":0,\"emp_name\":0,\"gender\":0,\"salary\":0,\"Dept\":0}}", "stats_parsed": {"numRecords": 1, "minValues": {"emp_id": 100, "emp_name": "Stephen", "gender": "M", "salary": 2000, "Dept": "IT"}, "maxValues": {"emp_id": 100, "emp_name": "Stephen", "gender": "M", "salary": 2000, "Dept": "IT"}, "nullCount": {"emp_id": 0, "emp_name": 0, "gender": 0, "salary": 0, "Dept": 0}}}	
9	null	{"path": "part-00000-cf890312-eed1-4328-bca0-1197a3c5b4a1-c000.snappy.parquet", "partitionValues": {}, "size": 1514, "modificationTime": 1695109341000, "dataChange": false, "tags": {"INSERTION_TIME": "1695109341000000", "MIN_INSERTION_TIME": "1695109341000000", "MAX_INSERTION_TIME": "1695109341000000", "OPTIMIZE_TARGET_SIZE": "268435456"}, "deletionVector": null, "baseRowId": null, "stats": "{\"numRecords\":1,\"minValues\":{\"emp_id\":300,\"emp_name\":\"Vada\",\"gender\":\"M\",\"salary\":22000,\"Dept\":\"COMP\"},\"maxValues\":{\"emp_id\":300,\"emp_name\":\"Vada\",\"gender\":\"M\",\"salary\":22000,\"Dept\":\"COMP\"},\"nullCount\":{\"emp_id\":0,\"emp_name\":0,\"gender\":0,\"salary\":0,\"Dept\":0}}", "stats_parsed": {"numRecords": 1, "minValues": {"emp_id": 300, "emp_name": "Vada", "gender": "M", "salary": 22000, "Dept": "COMP"}, "maxValues": {"emp_id": 300, "emp_name": "Vada", "gender": "M", "salary": 22000, "Dept": "COMP"}, "nullCount": {"emp_id": 0, "emp_name": 0, "gender": 0, "salary": 0, "Dept": 0}}}	null
10	null	null	{"path": "part-00000-ddbb5eea-f436-440 "dataChange": false, "extendedFileMetadata
11	null	null	{"path": "part-00000-e0e7fd83-5d19-487 "dataChange": false, "extendedFileMetadata

2. Cyclic redundant check - a checksum added to prevent corruption if a parquet file is corrupted in flight

{"txnId":"05e10a73-e754-4d4a-b5e2-ce6f43dc7900","tableSizeBytes":1521,"numFiles":1,"numMetadata":1,"numProtocol":1,"setTransactions":[],"metadata":{"id":"4f4d8577-a884-44bf-bd92-bb5536969f91","format":{"provider":"parquet","options":{}},"schemaString":"{\"type\":\"struct\",\"fields\":[{\"name\":\"emp_id\",\"type\":\"integer\",\"nullable\":true,\"metadata\":{}},{\"name\":\"emp_name\",\"type\":\"string\",\"nullable\":true,\"metadata\":{}},{\"name\":\"gender\",\"type\":\"string\",\"nullable\":true,\"metadata\":{}},{\"name\":\"salary\",\"type\":\"integer\",\"nullable\":true,\"metadata\":{}},{\"name\":\"Dept\",\"type\":\"string\",\"nullable\":true,\"metadata\":{}}]}","partitionColumns":[],"configuration":{"description":"table for demp purpose"},"createdTime":1695108333084},"protocol":{"minReaderVersion":1,"minWriterVersion":2},"histogramOpt":{"sortedBinBoundaries":[0,8192,16384,32768,65536,131072,262144,524288,1048576,2097152,4194304,8388608,12582912,16777216,20971520,25165824,29360128,33554432,37748736,41943040,50331648,58720256,67108864,75497472,83886080,92274688,100663296,109051904,117440512,125829120,130023424,134217728,138412032,142606336,146800640,150994944,167772160,184549376,201326592,218103808,234881024,251658240,268435456,285212672,301989888,318767104,335544320,352321536,369098752,385875968,402653184,419430400,436207616,452984832,469762048,486539264,503316480,520093696,536870912,553648128,570425344,587202560,603979776,671088640,738197504,805306368,872415232,939524096,1006632960,1073741824,1140850688,1207959552,1275068416,1342177280,1409286144,1476395008,1610612736,1744830464,1879048192,2013265920,2147483648,2415919104,2684354560,2952790016,3221225472,3489660928,3758096384,4026531840,4294967296,8589934592,17179869184,34359738368,68719476736,137438953472,274877906944],"fileCounts":[1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],"totalBytes":[1521,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]},"allFiles":[{"path":"part-00000-90b4b7bc-c7e1-488a-8473-0bee25f48695-c000.snappy.parquet","partitionValues":{},"size":1521,"modificationTime":1695109045000,"dataChange":false,"stats":"{\"numRecords\":1,\"minValues\":{\"emp_id\":100,\"emp_name\":\"Stephen\",\"gender\":\"M\",\"salary\":2000,\"Dept\":\"IT\"},\"maxValues\":{\"emp_id\":100,\"emp_name\":\"Stephen\",\"gender\":\"M\",\"salary\":2000,\"Dept\":\"IT\"},\"nullCount\":{\"emp_id\":0,\"emp_name\":0,\"gender\":0,\"salary\":0,\"Dept\":0}}","tags":{"INSERTION_TIME":"1695109045000000","MIN_INSERTION_TIME":"1695109045000000","MAX_INSERTION_TIME":"1695109045000000","OPTIMIZE_TARGET_SIZE":"268435456"}}]}

Command took 1.09 seconds -- by saurabh.aher@outlook.com at 9/19/2023, 1:16:13 PM on DemoCluster

Cluster Management	
Command	**Description**
cluster create	Creates a new cluster
cluster list	Lists all existing clusters
cluster start <cluster_id>	Starts a stopped cluster
cluster stop <cluster_id>	Stops a running cluster
cluster delete <cluster_id>	Deletes a cluster
Notebook Management	
Command	**Description**
notebook create	Creates a new notebook
notebook list	Lists all existing notebooks
notebook delete <notebook_id>	Deletes a notebook
notebook import <path>	Imports a notebook from a file
notebook export <notebook_id> <path>	Exports a notebook to a file
Data Management	
Command	**Description**
dbutils.fs.ls <path>	Lists files in a directory
dbutils.fs.cp <src> <dest>	Copies a file
dbutils.fs.rm <path>	Deletes a file or directory
dbutils.fs.mv <src> <dest>	Moves a file
dbutils.fs.mkdirs <path>	Creates a directory
dbutils.fs.head <path>	Displays the first few lines of a file
dbutils.fs.help()	Displays help for the dbutils.fs module
DataFrame Operations	
Command	**Description**
df.show()	Displays the first 20 rows of a DataFrame
df.printSchema()	Prints the schema of a DataFrame
df.select(<cols>)	Selects specified columns from a DataFrame
df.filter(<condition>)	Filters a DataFrame based on a condition

Commands

Command	Description	Example
dbutils.fs.cp	Copy	dbutils.fs.cp("/FileStore/old_file.txt", "/tmp/new/new_file.txt")
Dbutils.fs.head	Return specified bytes	dbutils.fs.head("/tmp/my_file.txt", 25)
Dbutils.fs.ls	List the directory files	dbutils.fs.ls("/tmp")
dbutils.fs.mkdirs	Create the directory structure if not exists	dbutils.fs.mkdirs("/tmp/parent/child/grandchild")
dbutils.fs.mount	Mount a source	dbutils.fs.mount(source = "wasbs://<container-name>@<storage-account-name>.blob.core.windows.net", mount_point = "/mnt/<mount-name>", extra_configs = {"<conf-key>":dbutils.secrets.get(scope = "<scope-name>", key = "<key-name>")})
dbutils.fs.mounts()	Display all mounted sources	dbutils.fs.mounts()

Benefit of mounting - to simplify data access patterns, to interact with cloud object storage using familiar file paths relative to the Databricks file system.

Partitioning Strategy

- File should be in splitable format -

Compression Options in Hadoop (2/2)

Format	Codec (Defined in io.compression.codecs)	File Extn.	Splittable	Java/ Native
zlib/ DEFLATE (default)	org.apache.hadoop.io.compress.DefaultCodec	.deflate	N	Y/ Y
gzip	org.apache.hadoop.io.compress.GzipCodec	.gz	N	Y/ Y
bzip2	org.apache.hadoop.io.compress.BZip2Codec	.bz2	Y	Y/ Y
LZO (download separately)	com.hadoop.compression.lzo.LzoCodec	.lzo	N	N/ Y
LZ4	org.apache.hadoop.io.compress.Lz4Codec	.lz4	N	N/ Y
Snappy	org.apache.hadoop.io.compress.SnappyCodec	.snappy	N	N/ Y

NOTES:

- **Splittability** – Bzip2 is "splittable", can be decompressed in parallel by multiple MapReduce tasks. Other algorithms require all blocks together for decompression with a single MapReduce task.
- **LZO** – Removed from Hadoop because the LZO libraries are licensed under the GNU GPL. LZO format is still supported and the codec can be downloaded separately and enabled manually.
- **Native bzip2 codec** – added by Yahoo! as part of this work in Hadoop 0.23

Spark.default.parallelism - total cpu core count - 8

Spark.sql.files.maxPartitionBytes - 128 mb (can be changed)

Spark.sql.files.openCostInBytes - 4 mb (overhead when merging 2 files)

bytesPerCore =(sum of size of all files + count of files * openCostInBytes) /default.parallelism

MaxSplitByte (Each partition size) = Min (maxPartitionBytes ,bytesPerCore)

3 files - 100, 500, 700 ,
4 core machine - 4 executor

Bytepercore = (1300 + 3 *4) /4 = 328
MaxSplitByte = Min (328, 128) = 128

Perform InputSplit

100 - > 100
500 - > 128 ,128, 128, 116
700 - > 128 , 128, 128, 128, 128 ,44

RecordReader
Partition packing happens if one partition can be made of 2 small files
Create 11 physical partitions

Partition Packing scenerio -

1. We have 5 files of 63 mb.

If we try to merge 2 files , then new partition size -> 63 +63 +4 = 130 > 128
Hence no files will be combined,
Total partition created will be 5.

1. We have 5 files of 62 mb.

If we try to merge 2 files , then new partition size -> 62 +62 +4 = 128
4 files will be combined to create 2 partition. 1 file will be in separate partition.
Total partition created will be 3.

DAG -
Job - stages - task

1 Task - 1 partition - 1 slot - 1 core

Repartition vs coalesce

Repartition	Coalesce
Use to increase or decrease no of partitions	use to decrease the number of partitions
Must require data shuffling	May not require data shuffling
Creates even sized partitions	Creates uneven partitions

Cache vs persists

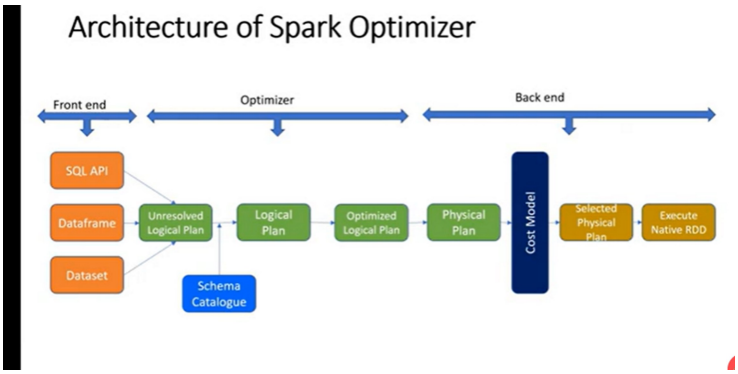
when we trigger action , then in memory processing starts and Data is prepared. If we execute similar operation multiple times, then the same data will be prepared many times. Hence we can store the require data in memory or disc for faster processing

1. Cache - store data in memory across worker nodes
2. Persist - store data in either in memory or disc

Catalyst Optimizer

Scala program to find most efficient plan to run spark program
Its available only for Dataframe or Dataset not for RDD

https://medium.com/@Shkha_24/catalyst-optimizer-the-power-of-spark-sql-cad8af46097f



Unresolved - after performing syntax and semantics checks -
Logical plan - adding schema validation to data

Explain plan - dfJoin.explain()

Catalyst Optimizer - is the man behind optimizing the SPARK SQL Queries. Let's see what it can do and how.

The Catalyst Optimizer is a component of Apache Spark's SQL engine, known as Spark SQL. It is a powerful tool used for optimizing query performance in Spark. The primary goal of the Catalyst Optimizer is to transform and optimize the user's SQL or DataFrame operations into an efficient physical execution plan.

The Catalyst Optimizer applies a series of optimization techniques to improve query execution time. Some of these optimizations include but are not limited to:

Predicate Pushdown: This optimization pushes down filters and predicates closer to the data source, reducing the amount of unnecessary data that needs to be processed.
Column Pruning: It eliminates unnecessary columns from being read or loaded during query execution, reducing I/O and memory usage.
Constant Folding: This optimization identifies and evaluates constant expressions during query analysis, reducing computational overhead during execution.
Projection Pushdown: It projects only the required columns of a table or dataset, reducing the amount of data that needs to be processed, thus improving query performance.
Join Reordering: The Catalyst Optimizer reorders join operations based on statistical information about the data, which can lead to more efficient join strategies.
Cost-Based Optimization: It leverages statistics and cost models to estimate the cost of different query plans and selects the most efficient plan based on these estimates.
By applying these and other optimization techniques, the Catalyst Optimizer aims to generate an optimized physical plan that executes the query efficiently, thereby improving the overall performance of Spark SQL queries.

Here is an overview of how the Catalyst Optimizer works:

1. Parsing: The first step is parsing the SQL query or DataFrame operations. The query is parsed to create an abstract syntax tree (AST), representing the logical structure of the query.

2. Analysis: In this step, the Catalyst Optimizer performs semantic analysis on the AST. It resolves table and column names, checks for syntax errors, and applies type checking to ensure the query is valid. Additionally, it collects metadata about the tables and columns involved in the query.

3. Logical Optimization: Once the query is analyzed, the Catalyst Optimizer applies a set of logical optimizations. These optimizations modify the logical plan to simplify or rewrite the query's operations, without changing the overall behavior or result.

4. Logical Plan: After the logical optimizations, the Catalyst Optimizer produces a refined logical plan. The logical plan is a tree-like representation of the query's operations and their dependencies. It captures the high-level operations like filters, joins, aggregations, and projections.

5. Physical Planning: At this stage, the Catalyst Optimizer generates various alternative physical plans based on the logical plan. It explores different execution strategies and physical operators suitable for the specific data sources involved in the query.

6. Cost-Based Optimization: The Catalyst Optimizer leverages statistics and cost models to estimate the cost of each physical plan. It considers factors like data distribution, network latency, disk I/O, CPU usage, and memory consumption. Using these cost estimates, it selects the physical plan with the lowest estimated cost.

7. Code Generation: Once the physical plan is selected, the Catalyst Optimizer generates efficient Java bytecode or optimized Spark SQL code for executing the query. This code generation process can further improve the performance by leveraging the optimizations provided by the underlying execution engine.

8. Execution: Finally, Spark SQL executes the optimized physical plan to compute the result of the query. During the execution phase, Spark leverages various techniques like in-memory caching, partitioning, and parallel processing to efficiently process the data and produce the desired output.

By following this comprehensive process, the Catalyst Optimizer in Spark SQL optimizes and executes queries with improved performance and efficiency.

Broadcast Variable -

Separate copy on each variable -
For joining a fact and small dimension table ,
using broadcast variable we can make a copy of dimension table to all worker node,
this will improve performance.
Broadcast variable is cached in each node.
It avoids data shuffle
It reduces network I/O
joinDF = transactionDF.join(broadcast(dimDF), transactionDF.id == dimDF.product_id)

Accumulator - single copy on driver node
Shared variable - which can be updated by everyone -
Executor cannot read the data ,it can only update

Adaptive Query Execution -

tunes the execution plan for a given query, based on the runtime statistics of the data.

Spark.set.config(spark.sql.adaptive.enabled, True)

Data skew -

Optimizing query in stages

1. Dynamically coalescing shuffle partitions - it will change partitions dynamically.
2. Dynamically switching join strategies - choose best between shuffle join and broadcast join
3. Dynamically optimizing skew joins - choose best partition strategy

   - spark.cofig.set("spark.sql.adaptive.enabled", true)

RDD

1. Foreach ,aggregate , foreach vs Map, fold,
2.

Pyspark

1. Read

spark.read.format("parquet").load("/FileStore/data/auto_mpg.parquet")

1. Write without header - data will be partitioned

```
auto_df.write.mode('overwrite').csv("/FileStore/data/output.csv")
```

Writing with header - no data partitioning - saving file to only 1 partition

```
auto_df.coalesce(1).write.csv("/FileStore/data/header.csv", header="true")
```

3. Create hive catalog table

```
auto_df.write.mode("overwrite").saveAsTable("autompg")
df = spark.table("autompg")
```

4. SQL query on hive cataloh

```
df = sqlContext.sql("select carname, mpg, horsepower from autompg where horsepower > 100 and mpg > 25"
)
```

PartitionBy

Load data from azure blob to azure databricks

UDF - black box for spark engine, not recommended for databricks optimization, it envolves serialization and deserialization

```
def convertCase(string):
    retString =""
    if string is not None:
        string = str(string)
        retString = string[0:1].upper()+string[1: len(string)] +""
    return retString

# Converting function to UDF
convertUDF = udf(lambda z: convertCase(z), StringType())
```

Map - stringlist.map(x => x.length) - map a specific function to all values of list.
Reduce - reduce((x,y) => x + y) - reduce the table to particular value

Databricks workflows -
Collection of tasks -

Integrate ADLS Gen 2 to ADB

1. Using service principle
2. AAD credential using credential passthrough
3. Using ADLS access key directly
4. Create mount point using ADLS access key

Handling nulls

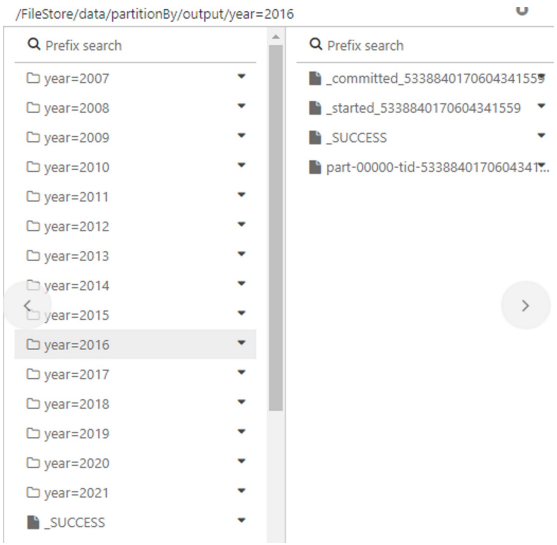
Df.isnull(), df.isnotNull(),
na.drop() - drop all rows or columns containing null values
Na.fill() - fill the null values
df.na.fill("").show()
df.na.fill({"state": "NA", "gender": ""}).show()

Array Functions -

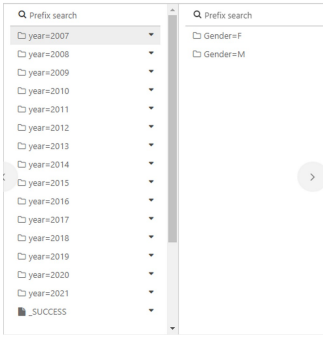
PartitionBy - write the data into disc partition by specific keys
Df.write.partitionBy(key).csv(path)

1. **Partition by one key column** -

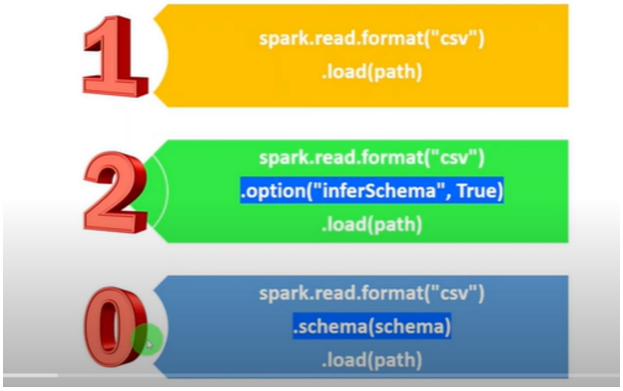
df.write.option("header",True).partitionBy("year").mode("overwrite").csv("/FileStore/data/partitionBy/output")



2. df.write.option("header",True).partitionBy("year","Gender").mode("overwrite").csv("/FileStore/data/partitionBy/output_multiple")



3. How many spark job gets created ?



Joining in Spark

1. Broadcast Join - by default when joining with small dim tables - less than 1 gb
spark.sql.autoBroadcastJoinThreshold

2. Sort Merge Join - default for many joining operations

   Shuffle + sort + Merge
   Shuffling and sorting are costly operation

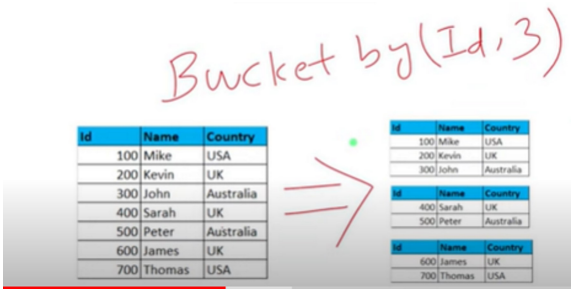
3. Shuffle Hash Join

[Top 10 query performance tuning tips for Databrick... - Databricks - 43218]

BucketBy -

1. Shuffle and sort once - store it in storage -

2. PartitionBy - create one folder for each partition value.
   Its suitable for column with low cardinality
   BucketBy - create defined buckets with data -
   Its useful for high cardinality columns



Join Hints -

Select /* + BROADCASTJOIN(t1) */ * from t1 inner join t2
on t1.key = t2.key

Skew

Non uniform distribution of values in column -

1. Repartition, Coalesce

2. Specify Join hints -
   Select /* + BROADCASTJOIN(t1) */ * from t1 inner join t2
   on t1.key = t2.key

3. Salt the skewed column - Introduce randomness by concatenating with random column

How to identify the skew -

Check the summary metrics for task

Summary Metrics for 8000 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	0 ms	1.5 min	3.8 min	8.5 min	1.7 h
Scheduler Delay	0 ms	4 ms	4 ms	8 ms	1.8 h
Task Deserialization Time	0 ms	17 ms	18 ms	25 ms	6.1 min
GC Time	0 ms	8 s	21 s	50 s	1.2 h
Result Serialization Time	0 ms	0 ms	0 ms	0 ms	3 s
Getting Result Time	0 ms	0 ms	0 ms	0 ms	0 ms
Peak Execution Memory	0.0 B	0.0 B	0.0 B	0.0 B	0.0 B
Input Size / Records	0.0 B / 0	0.0 B / 0	0.0 B / 0	0.0 B / 0	5.0 GB / 4507
Output Size / Records	0.0 B / 0	152.5 MB / 1445781	353.8 MB / 3943952	675.0 MB / 8661301	4.4 GB / 67617267
Shuffle Read Blocked Time	0 ms	1 ms	3 ms	7 ms	18 s
Shuffle Read Size / Records	0.0 B / 0	185.0 MB / 1444759	427.4 MB / 3943915	808.4 MB / 8658728	5.3 GB / 67617267
Shuffle Remote Reads	0.0 B	183.4 MB	423.4 MB	800.9 MB	5.2 GB
Shuffle spill (memory)	0.0 B	0.0 B	0.0 B	3.6 GB	34.0 GB
Shuffle spill (disk)	0.0 B	0.0 B	0.0 B	590.5 MB	4.0 GB

1. If some task is taking too long to complete execution.

Spill

Act of moving rdd from RAM to disk and later back to RAM.
Occurs when the given partition is simply too large to fit in RAM
Out of Memory errors

Check spark.sql.files.maxPartitionBytes

You need to add more memory -

Shuffle

Wide transformation -
Join, distinct , groupBy , orderBy, count()

1. Cost Based optimizer (spark 3.0) helps to give least shuffle plan
2. Reduce network I/O by using fewer and larger nodes
3. Reduce the amount of data being shuffled
   Select only required data

Size

Small size problem

Ideal partition size - 128 mb to 1 GB

Compact small files -
Optimize - create 1 GB file - spark.databricks.delta.optimize.maxFileSize
Auto Optimize - always optimize
Vaccum - remove previous files
Z - ordering - order the optimize file

Spark UI

Ganglia Metric

Delta

1. Create delta table - Python, sql, dataframewrite
2. Delta table instance -
   soft link of actual table - both instance and table are in sync.
   Use - to perform DML operation using pyspark language if you don't wish to use SQL.

   Delta = DeltaTable.forPath(spark, "/FileStore/tables/DeltaUsingPyspark")

   Create using - Path and Table Name

3. Slowly changing dimensions -
   SCD1 - overwrite ,
   SCD2 - previous inactive and new record active - duplicity
   SCD3 - new column in updated value
4. Delta table time travel
   select * from employee_demo version as of 19;
   select * from employee_demo timestamp as of "2023-10-05T06:32:36.000+0000"
5. Delta table Restore
   It will restore the table to previous version but history is not overwritten , restore history will be added.
   Delta.restoreToVersion(34)
   Delta.restoreToTimestamp("2023-10-06T10:31:38.000+0000")

6. Delta table Optimize
   Combine the parquet file to reduce the no. Of files. - create 1 GB files
   --optimize employee_demo

7. Delta table Z - ordering - optimize with data ordering - data sorting first - better performance
   optimize employee_demo zorder emp_id
   Delta stores the statistics for each file.

8. Delta table vaccum - cleanup data periodically
   Delete the files which are invalidated before 7 days - 168 hr
   vacuum employee_demo retain 72 hours dry run

9. Delta table Schema evolution - to handle continuous change of dataframe schema
   df.write.option("mergeSchema", "true").format("delta").mode("append").saveAsTable("employee_demo")

10. Delta table formatting -

Repartition, coalesce , Optimize, z-order ,vaccum, schemaEvolution

---

Spark Optimization -

1. Hierarchy -

   Local disk - faster the disk , faster the performance

2. Transformation - Narrow(no shuffling) and wide (shufling)
   Action - made of transformations -
   Job -> stages -> Task- > interact with hardware - (1 Task - 1 Partition - 1 Core - 1 Slot)

3. Understand spark UI
4. Understand hardware -

5. CPU utilization graph - databricks metrics

6. Minimize data scans (Lazy Load)
   Hive partitions
   Bucketing
   Delta Z ordering

7. Cast explicitly the column types -

8. Add filter to data first -

Hive partition is not spark partition

spark partition -
input, shuffle, output

Shuffle partition default - 200

Partition count = stage input data / target size

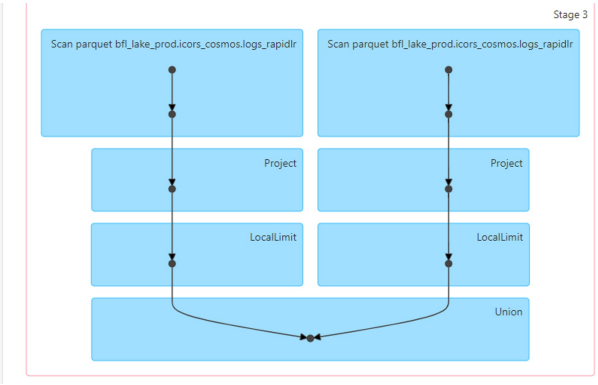
DAG -
WholeStageCodeGen - it uses java code gen , more memory efficient and performant, we should see this box everywhere.

ColumnarToRow - spark is row based whereas parquet is columnar

Project - withColumn, Select,

LocalSort - df.sortWithinPartition("itemId") - No exchange
GlobalSort - df.orderBy("itemId") - Exchange involved

Union - Narrow



Memory management in spark

Ease of access = Register -> Cache - > Memory -> Disk

Two uses of memory

1. Execution - shuffle, join, sort, agg , short lived
2. Storage - Cache , long lived

Unified memory management - one memory for storage and execution- > evict storage, spill to disc

Delta Cache - disk persist

Spark cache - In memory persis

Questions -

1. Client and cluster mode - depend on location of driver
   Client - driver in client,
2. Cluster manager - YARN , Mesos, YARN

3. Spark session and spark context -
   Spark session - unified entry point for spark context, sql context , streaming context

4. How to achieve parallelism in spark ? Multiple nodes
5. RDD , Dataframe and dataset -
6. Lazy evaluation ? Spark will execute actions.
7. DAG ? -
8. Spark architecture ?

WORKFLOWS IN DATABRICKS

1. JDBC connection defination - secrets

```
driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver"

database_host = "sa-sql-db.database.windows.net"
database_port = "1433"
database_name = "azureSQL"
table = "students"
user = "saurabh"
password = "Test@1234"

url = f"jdbc:sqlserver://{database_host}:{database_port};database={database_name};user={user};password={password}"
```

2. Read data from azure sql db in dataframe

```
database = "BFL_EDW"
table = "table_name"
user = "generic_id"
```

```
password = "generic_id_pass"

df1 = spark.read\
.format("jdbc")\
.option("url", f"jdbc:sqlserver://bflazdw-dev.database.windows.net:1433;database=BFL_EDW;")\
.option("dbtable", table)\
.option("user", user)\
.option("password", password)\
.option("fetchsize","1000000")\
.option("driver", "com.microsoft.sqlserver.jdbc.SQLServerDriver")\
.load()
```

3. Create a mount point to ADLS

```
dbutils.fs.mount(
   source ="wasbs://adfdemo@storagesaurabh2023.blob.core.windows.net",
   mount_point = "/mnt/blob_test",
   extra_configs =
{"fs.azure.account.key.storagesaurabh2023.blob.core.windows.net":"y+wV6xrnYnkTRUl/LBMWPptES38/BrxK6TliDmzyW2FJzdoTPvTG35d7aZXup7ZG
GjuoJtW7xvGA+ASt1Af1SA=="}
)

dbutils.fs.ls("/mnt/blob_test")
```

4. Write to ADLS

```
df1.write.format('delta').mode("overwrite").saveAsTable("bfl_std_lake.schema_name.table_name")

df1.write.format('delta').mode("overwrite").save("path")
```

1. Create JOB -

With workflows job we can execute multiple tasks (notebook) ->
Second task can be dependent on first task
One task can have multiple task

You can use all purpose or job cluster for schedule job

All purpose cluster - >
Small
Standard e4v4 -each worker -> 4 core, 32GB gb
1-2 Workers
32-64 GB Memory
4-8 Cores
12.2 LTS (includes Apache Spark 3.3.2, Scala 2.12)
2–3 DBU/h

Common analysis

Standard e8as_v4 each of 8 core, 64 gb
Total 32 core, 256 gb

4 -10 DBU

JOB Cluster-
Standard e8as_v4

1 - 8 workers each of (8core -64 gb)

Total - > 1 -8 core, 64 -512 GB
11.3 LTS (Scala 2.12, Spark 3.3.0)
12.2 LTS (includes Apache Spark 3.3.2, Scala 2.12)

EXECUTE notebook via ADF

It will require user access token

Input - widget-> text, dropdown, combo
Output - > dbutils.notebook.exit(<value>)

ADF - @activity().output.runOutput

sparkContext - one sparkcontext per JVM - used for creating RDD, accumulator, broadcast variable,
sparkSession - driver process

UC
EDW and DL pipeline connect
Optimization through cluster management

Revision
Metastore

UC to EDW

```
database = "BFL_EDW"
#Database Name

table = "SCHMA_NAME.TABLE_NAME"
#EDW Table name where data will be stored

user = "Generic User Name"
password = "Generic User Password"
#Details of Managed Identity for JDBC connection

tempDirPath = "abfss://tempdir@bfldlstandardisedzone.dfs.core.windows.net/tempDirEDW/"
# ADLS Container path for data staging

data = spark.sql("SELECT * FROM Unity_Table_to_be_written")
# Table/Data to be written/tranferred to EDW

data.write.format("com.databricks.spark.sqldw")\
.option("url",f"jdbc:sqlserver://bflazdw-dev.database.windows.net:1433;database=BFL_EDW;encrypt=true;")\
.option("useAzureMSI","true")\
.option("dbtable",table)\
.option("user",user)\
.option("password",password)\
.option("batchsize","100000")\
.option("tempDir", tempDirPath)\
.option("tempCompression","SNAPPY")\
.option(queryTimeout, "7200")\
.mode('append')\
.save()

# Please not the following:
# This method automatically creates a table if the table is not present in EDW with the corresponding data types for EDW:
#  string --> nvarchar
#  decimal(18,0)     --> decimal(18,0)
#  date --> date
#  bigint --> bigint
#  double --> float
#  float --> real
# It is recommended that you convert all the floating data types into decimal before writing into EDW to avoid precision loss.
#
#  1. There are different modes of writing data into table. mode('append') signifies data will be inserted into an existing table.
#     'overwrite' option overwrites the existing data with the new data in the dataframe.
#
#  2. option("tempDir", tempDirPath) is used to make a temporary staging area for the data before transfer via Polybase.
#     This option ensures the metadata is kept intact and increases the preformance of writes.
#     The path present in the variable tempDirPath is a shared container across all users in UC. This path will be auto cleared by Datalake team from time to time.
#     As this path has public access, recommended to use personal containers while writing PII data in EDW instead of the same.
#
#  3. option("tempCompression","SNAPPY") is the compression algo used to encode the data. We can also use GZIP compression although, SNAPPY performs
faster.
#
#  4. option("maxStrLength","1000") is to be added when to the code when the size of the code exceeds 256 charachters(default).
#     1000 to be replaced with the required length value for each string column.
#
#  5. option(queryTimeout, "7200") sets the duration post which the query should be automatically timed-out.
#
#  6. option("batchsize","100000") increases the batch size from 100(default) to 100000 improving the write speed into EDW.
#
#  7. Isolation Levels also play a vital role in how fast the data is written into EDW although, can result into several issues if used incorrectly.
#          There are 4 isolation modes:
#           1. READ_UNCOMMITTED -> Data from the table can be read even when a transaction is not committed & is the default mode.
#           2. READ_COMMITTED -> Data can be only ready post commit of any transaction.
#      3. REPEATABLE_READ -> Locks the row/transaction state of the table to the first read/query.
```

```
#      4. SERIALIZABLE -> Locks the entire table till the transaction is completed.
#     Please use the option("isolationLevel","IsolationType") with caution.
```

Dbdemos - for demos

Delta Sharing -

WithColumn vs Select - use select

Different ways of select -
Df.column
Df["column"]
Col("column")
"column"

Azure databricks secret scopes

Collection of secret in databricks

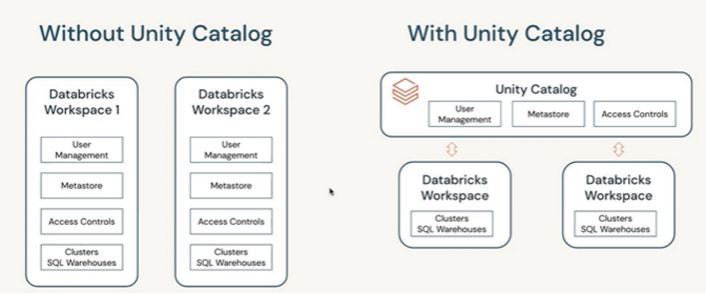
1. Key vault backed scope - created by secrets/createScope
2. Databricks backed scope - created by databricks CLI

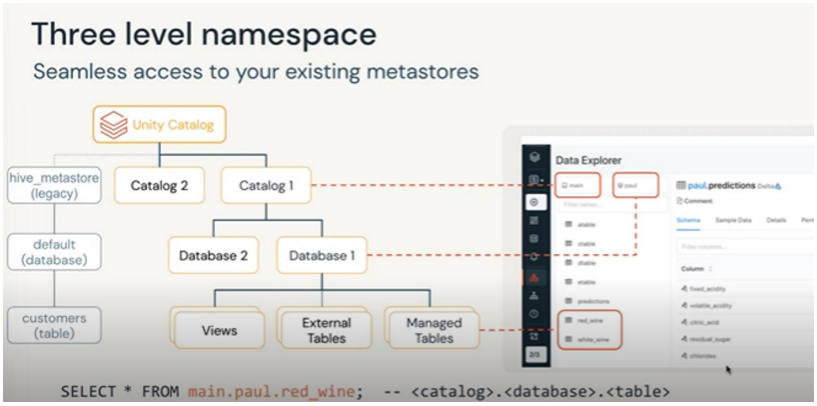
Unity Catalog

Unified governance model across all data assets

Previous to UC - everything to workspace level

With UC - account

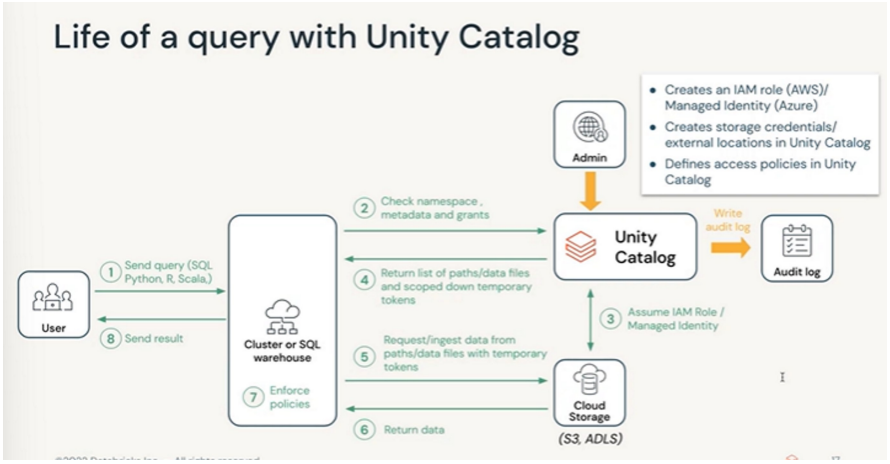




Function in SQL
Grant Permission in SQL
RLS in PBI
SET ROW FILTER in SQL
SET MASK

External tables will not be dropped any time

Query lifecycle -



Azure Storage account
Azure databricks
Azure access connector - create and assign a role as storage blob data contributor in storage account

Go to https://accounts.azuredatabricks.net/ - > data - > create metastore

Create metastore and assign workspace to it.

Lineage - the visualization for table transformation

Delta sharing - share data with non databricks
Connect with powerbi

1. Create RDD
2. Create Execution plan
   Split into stages - for efficiently
3. Schedule task (data +computation)

Stages - shuffling involved

Shuffle - pull based

Comment - for comment for table

Array _function - for handling nested json

Create function - for creating UDF - Create function

```
CREATE or replace  Function textUpper (text STRING )
RETURNS STRING
RETURN concat(upper(text), '111')
```

Autoloader - structured streaming

COPY INTO - ingest raw file into bronze layer -
load data from a file location into a Delta table. Exactly once processing
Guarantee than file is read only once.
Copy into transactions
From "filepath"
Fileformat = parquet

UNION, MINUS, INTERSECT

Object -
Table -
external- in external location ,
 managed - in hive warehouse
View -
Standard - view query is saved
temporary - view is stored for the particular session only
Global (temporary view with sharing across other notebooks.)
Function-

Reading from cloud data
Create table employee
Using JDBC
When to use USING ?

Auto loader -

CICD pipeline

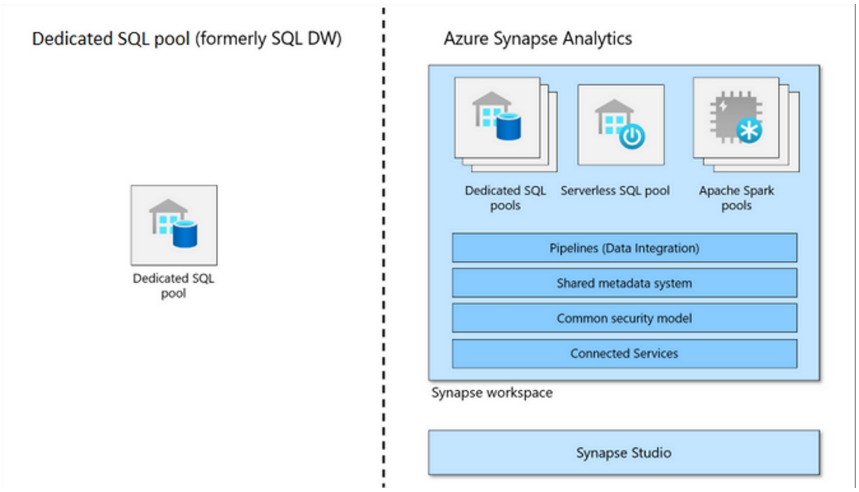
JOB Scheduling

Handling json in pyspark

Azure Synapse

27 February 2024 11:51 AM

Azure Synapse - One solution for Data warehousing in azure _> SQL + ADF +ADB



Create-
It requires ADLS Gen2 Account as storage
Managed resource group - synapse create another resource which will be stored in this resource group.

[Performance tuning guidance for Azure Synapse Analytics serverless SQL pool - Azure Synapse Analytics | Microsoft Learn]

Synapse workspace - collaboration place where you can perform data engineering , science analytics.
SQL Pool and Spark Pool - cluster runtime

Linked Service - Connection string to connect with external resources

Synapse SQL - ability to execute SQL queries
SQL Pool - >
dedicated
Serverless - default (pay for only that time when you are running the query.) (amount of data queried)
Built In

Spark Pool ->

Pipelines - ETL pipelines

Dataset - pointer for data

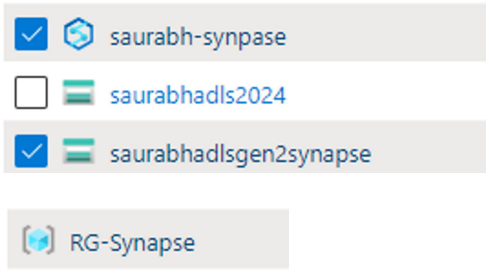
you can write query on unstructured file also
openrow
Select * from openrowset(
BULK,
FORMAT,

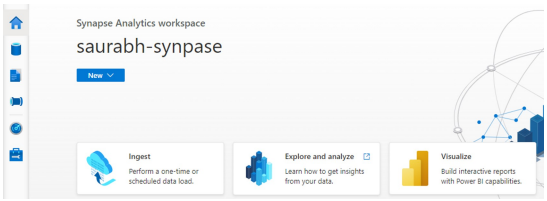
Header_row = true

)

P2. Create Synapse workspace-

1. It requires 2 resource group - 1 for default and another for managing the synapse resources.
2. It requires one ADLS Gen2 Account





Similar to ADF

Serverless SQL Pool Name - Built In

Linked Services - >

saurabh-synpase-WorkspaceDefaultSqlServer	Azure Synapse Analytics
saurabh-synpase-WorkspaceDefaultStorage	Azure Data Lake Storage Gen2

CETAS

```
CREATE EXTERNAL TABLE population_by_year_state
WITH (
   LOCATION = 'aggregated_data/',
   DATA_SOURCE = population_ds,
   FILE_FORMAT = census_file_format
)
AS
SELECT decennialTime, stateName, SUM(population) AS population
FROM
   OPENROWSET(BULK
'https://azureopendatastorage.dfs.core.windows.net/censusdatacontainer/release/us_population_county/year=*/*.parquet',
   FORMAT='PARQUET') AS [r]
GROUP BY decennialTime, stateName
GO
```

```
-- you can query the newly created external table
SELECT * FROM population_by_year_state
```

```

```

gf

```
CREATE EXTERNAL DATA SOURCE MyAzureInvoices
 WITH ( LOCATION = 'https://<storage_account>.dfs.core.windows.net/<container>/<path>') ;
```

CTAS

```

```

```
SELECT *
FROM OPENROWSET(
BULK 'http://<storage account>.dfs.core.windows.net/container/folder/*.parquet',
FORMAT = 'PARQUET'
) AS [file]


Select * from openrowset(
BULK,
FORMAT,
Header_row = true

)
```

OPENROWSET - access file from storage account

Dedicated SQL pool - it has cost -
DWU - unit for pricing
When you create DSP - it creates database and other resources also also

SPARK POOL
Used for running notebooks

Adding Role in synapse RBAC

Dedicated SQL pool

Whenever you submit query to control node, it will be divided in parallel by MPP engine to all compute nodes.
Each distribution will be computed by a compute node
DMS - Data movement services

Serverless-
Whenever you submit query to compute node, it will be distributed into smaller task by distributed query engine
With serverless you can actually query data in ADLS, Delta, Cosmos DB

We can create Logical DW - layer - it will only store metadata -
there wont be any tables, triggers, materialized view, DDL, DML statements
It can store - external - > sources, schema, views, tables,

Data virtualization
External data source - connection string to external data.

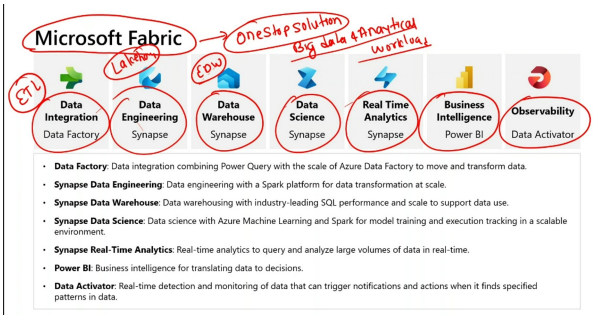
Distribution

```
Create table dbo.table
(
Col1 int NOT NULL,
Col2 varchar NOT NULL
)
With
(
DISTRIBUTION = HASH (col1).
CLUSTURED COLUMNSTORE INDEX
)
```

HASH -

Fabric Training

01 December 2023 09:33 AM



Onelake - single copy of data.

Fabric - must be enabled at tenant level by fabric admin, power platform, global admin,

Lakehouse - database built on delta table - consist of tables and files,

Managed table - location is not specified. - it will be created under default metastore.Dropping table deletes the files as well.

External table - table given with location. Dropping table doesnt deletes files.

Streaming data - readstream -

Spark tutorial

10 February 2024 05:44 PM

Spark is a general purpose, In memory, compute engine

Compute Engine

Hadoop provides 3 things -
Storage - HDFS
mapReduce - Compute
YARN - Resource manager

Spark is an alternative to mapReduce.

It requires storage and manager
Storage - local, HDFS, S3, blob
Resource manager - YARN, Mesos, Kubernates

What is in memory ?

mapReduce - take data from disk and saves the output to disk
Many mapReduce job , Increase i/o operation

Spark - all processing is done in memory

Spark is 10 to 100 times faster than mapReduce.

General Purpose

In Hadoop we have to use
Pig for cleaning
Hive for querying
Sqoop for data ingestion
Mapreduce is bound only with map and Reduce

In spark we can do everything in one

The basic unit which holds the data in spark is called as RDD.
RDD - Resilient Distributed Dataset.
- Its a big list divided into many machines

Directed Acyclic Graph -
There are 2 type of operations-

Transformation - Lazy - it will create a DAG
(why lazy - it will find the best suitable execution plan)

Actions - Eager

RDD is
Resilient (fault tolerance) - its resilient to failures , if we loose RDD, we can recover(create by using graph) it back. It provides fault tolerance through DAG.

Distributed - RDD is distributed in memory , Files are distributed in file storage.

Immutability - we cannot change content of RDD once created, for fault tolerance

RDD is filled with data - RDD is materialized

Practical ->

Word count in spark
Cloudera vm -

HDFS processing -
Spark-shell - scala
Pyspark

SparkContext - sc - entry point to run your code on cluster

Val rdd1 = Sc.textFile("loadPath)

Rdd2 = Rdd1.flatmap(x => x.split(" "))

Flatmap - works on each line, whenever we split, we get array

Rdd3 = Rdd2.map(x => (x,1))

Rdd4 = Rdd3.reduceByKey((x,y) = > x+y)

Localhost: 4040

saveAsTextFile("dataFile")

File : // to store file in local storage

DateDiff - for difference in month, days ,week

CountByValue - action , not value
Whatever action done after this will be done on your local machine.

When we want
no of output = no of input use map

no of output > no of input use flatmap

no of output < no of input use reduce

Partitions
mapValues - deal with only values

sc.defaultParallelism - 8 as default parallelism

To check the number of partition - Rdd.getNumPartitions

While loading from DBFS - no of partition = no of executors
Creating rdd in code using parallelize - no of partition = sc.defaultParallelism

sc.defaultMinPartitions - determines the minimum no. Of partition rdd has when we load from file, if partition is less than 2 , it will make 2 partition

Repartition vs coleasce

Repartition - to increase or decrease number of partition , wide transformation, shuffling involved
It creates equal size partition

Coalesce - it can only decrease the number of partition,
but it won't give error we try to give higher value .
It will try to minimize the shuffling while reduction. It will try to combine partition on same machine.

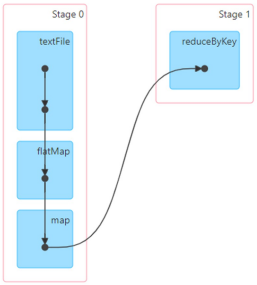
Narrow and wide transformation
Narrow - no shuffling - Map, flatmap, filter
Wide - shuffling - reduceByKey

Shuffling - move data from one machine to another

Stages in spark - whenever there is shuffle, a new stage gets created , whenever we call wide transformation, new stage will be created

If you use 3 wide transformation - 4 stages will get created.
Output of stage 1 will be sent to disk and stage 2 reads it back from disk. Involves disk I/O

Minimize stages - use wide transformation at end only.



- groupBykey vs reduceBykey

reduceByKey - first it will do local aggregation on single machine and then sent for shuffling.
Its like a combiner acting at backend.
Less shuffling involved.

GroupByKey - no local aggregation , more shuffling , not performant, dont use in production
One key will be only on one machine - it can lead to out of memory error

No of job - no of actions
No of stages - no of wide trasformation + 1
No of task - no of partition

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	64.0 ms	64.0 ms	78.0 ms	78.0 ms	78.0 ms
GC Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Shuffle Read Size / Records	1 KiB / 138	1 KiB / 138	1.1 KiB / 165	1.1 KiB / 165	1.1 KiB / 165
Shuffle Read Fetch Wait Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Shuffle Remote Reads	0.0 B	0.0 B	0.0 B	0.0 B	0.0 B

Pair RDD - Keys can repeat

Map - can only have unique keys

Output of spark will always be a folder.

Collect - can lead to driver crashing

sortByKey - part of it is eager , it will be shown as a job.
Count -
Transformation -
Narrow - map, filter,
Wide - reduceByKey, groupBykey

Action - collect , saveAstext, Reduce, count

Stage Skipping
Action - Whenever you call action - all transformation will be executed. But sometime spark optimize it internally, and hence skip the stages. It takes output from the stage.

If we want to reuse RDD instead of created again and again

Cache - Rdd.cache() - green color , but it will still show the stage

RDD which is not cached is reevaluated again each time when RDD is cached.

Cache - it will cache in Memory
Persist - persist in disk or cache -
If you use persist without argument, it is also in memory
Persist() , Persist(StorageLevel.) - same as cache

Persist(StorageLevel.MEMORY_ONLY)

MEMORY_ONLY - non serialized format , like cache ,takes more storage
DISK_ONLY - stored at disk in serialized format
MEMORY_AND_DISK - first stored in memory and if space is not there then gets stored at disk
OFF_HEAP - blocks are cached in off Heap memory , no garbage collection required

Block eviction -
LRU - least recently used

Personal

21 March 2022 10:38 AM

How to ask for referrals?

1) Don’t start with small talk and get straight to the point. Don't start a conversation with a "Hi" and then wait for the other person to react. Send one text with all the details.

2) Establish a proper context. Don't just ask, "Can you refer me?" and leave it at that. Include your work experience, educational credentials, and why you want the job and are a good fit for it.

3) Keep the message informative. Send your message with the proper job id and URL of the opening (from company career page, LinkedIn, etc). Save them the job. Make it easy for them.

4) Make a template for yourself. It would be easier for you to apply to several roles if you use an appropriate referral message template that covers the above items.

Example :

Greetings,
Thank you for connecting with me, hope this message finds you well. I’m XYZ, from ABC College. I saw XYZ job vacancy at your organisation and believe that my expertise with ABC and XYZ qualifies me for the position. Is it possible for you to refer me for the same?

I’ve attached my resume (and portfolio) below for your reference and I’ll be happy to communicate further or get on a call if you can spare some time for the same. I'll be looking forward to hearing from you.

Thank you
Sincerely,
ABC

5) You can also ask for advice before asking for referral. "Hey, I thought position X can be a good fit for me. Do you think I am ready to apply for this position, or am I missing key skills? Is there anything you can suggest to help me be a better candidate for this position?

6) It is usually easier to obtain a response from someone who is 2-5 posts above your interested position than it is to get a response from someone who is 20+ levels above your position. Be mindful.

7) If you don't hear back after 4-5 business days, follow up, but don't spam. Even if you don't get the job, keep them informed and build connections with whom you can reconnect later in your career.

Checkout my last posts for Who to ask for referrals and how to network! You can also contact me for further help. Do follow and stay tuned!

lenovo thinkbook 15 - 60000
HP pavilian 15 - 60000
lenovo ideapad flex 15
mi notebook pro - 61000 - 16gb ddr 4 -512 ssd - no expandable
lenovo ideapad slim 5
acer nitro 5 - 4gb graphic card , i5 11, 8gb ddr4, 512gb ssd, - 60000 - best

acer aspire 7 -
Hp victus
leanovo ideaokad slim 5 - i5 11

HP pavilian gaming - 8gb, 512gb, ryzen 5600
acer aspire 7 -ryzen 5500u, 57000
HP victus - ryzen 5600h, amd -
lenovo ideapad gaming 3 - i5 11, 65990-
acer nitro 5 -

Video editing -
Performance -

---

designed and developed a data flow model for a fortune 500-client which involved automating the ingestion and transformation of huge amount of data on daily basis using Microsoft Azure platform tools catering to more than 50 internal projects.
▪ Use of Local Development and Testing methodology-
• Using test cases and mockup data creation
• CI/CD pipelines
• Sphinx Documentation
• Functional Programming
• Data Quality Management
▪ Implementation and administration of Git-based code repositories and branching/versioning standards
▪ Using Azure DevOps to plan sprints in a smarter way and keep track of all developments, also used to test the product in an automated way.

▪ Salesforce to Redshift Ingestion - Migration from Informatica to Native AWS
-> Tech Stack – Salesforce, Informatica, S3, Lambda, Glue, AppFlow, Redshift, SNS
- Crafted generic scalable Native AWS solution for Salesforce to Redshift ingestion
- It helped to move ingestion pipelines from third party tool Informatica and saved cost for heavy license fee
- This generic framework helped other business units for smooth ingestion of newly onboarded Salesforce object into Redshift Datalake

▪ Incremental Ingestion pipeline – Employee Benefits Data
-> Tech Stack – Shell Scripting, AWS CLI, S3, EMR, Glue, Redshift, SNS, QuickSight,PySpark
- Build generic & optimized ingestion pipeline for highly critical & confidential Employee Benefits Data
- Pipeline is designed in a way to handle GB's of daily & weekly data together for different use cases like Audit, Payroll, Reimbursement, Education Reimbursement etc
- Took complete ownership and worked closely with business teams to understand the requirements & deliver enriching dashboards

▪ Pipeline Optimization & Enhancement
-> Tech Stack – Shell Scripting, AWS CLI, S3, EMR, Glue, Redshift, SNS, QuickSight, PySpark, Lambda
- Enhanced & optimized multiple pipelines, built for different business units like Peoplesoft, Audit, AEM, Immigration, Accurate, MyDocs, Background Verification etc
- Reduced execution time by 50% and improved alerting system for different edge cases
- Leadership principals like Customer Obsession, Earn Trust and Think Big, helped me to keep improving existing systems

▪ Automated Alerting System for Job Monitoring
-> Tech Stack – Python, AWS CLI, QuickSight
- Created automated alerting system for Redshift load metrics and Job monitoring
- It saved 1.5 Hours/day of manual efforts by each team member to monitor & prepare Daily Job Status▪ Salesforce to Redshift Ingestion - Migration from Informatica to Native AWS -> Tech Stack – Salesforce, Informatica, S3, Lambda, Glue, AppFlow, Redshift, SNS - Crafted generic scalable Native AWS solution for Salesforce to Redshift ingestion - It helped to move ingestion pipelines from third party tool Informatica and saved cost for heavy license fee - This generic framework helped other business units for smooth ingestion of newly onboarded Salesforce object into Redshift Datalake ▪ Incremental Ingestion pipeline – Employee Benefits Data -> Tech Stack – Shell Scripting, AWS CLI, S3, EMR, Glue, Redshift, SNS, QuickSight,PySpark - Build generic & optimized ingestion pipeline for highly critical & confidential Employee Benefits Data - Pipeline is designed in a way to handle GB's of daily & weekly data together

for different use cases like Audit, Payroll, Reimbursement, Education Reimbursement etc - Took complete ownership and worked closely with business teams to understand the requirements & deliver enriching dashboards ▪ Pipeline Optimization & Enhancement -> Tech Stack – Shell Scripting, AWS CLI, S3, EMR, Glue, Redshift, SNS, QuickSight, PySpark, Lambda - Enhanced & optimized multiple pipelines, built for different business units like Peoplesoft, Audit, AEM, Immigration, Accurate, MyDocs, Background Verification etc - Reduced execution time by 50% and improved alerting system for different edge cases - Leadership principals like Customer Obsession, Earn Trust and Think Big, helped me to keep improving existing systems ▪ Automated Alerting System for Job Monitoring -> Tech Stack – Python, AWS CLI, QuickSight - Created automated alerting system for Redshift load metrics and Job monitoring - It saved 1.5 Hours/day of manual efforts by each team member to monitor & prepare Daily Job Status McKinsey & Company logo

▪ Data Ingestion & Sync Process
-> Tech Stack – Python, Hive, ElasticSearch, Scala Play Framework, SBT, EMR, Lambda, DynamoDB, Azkaban, Jenkins
- Crafted data-sync logic by prioritizing datasets (High/Medium/Low tag) based upon criticality to meet SLO
- Built premption logic to prioritize highly critical datasets when multiple low priority sync processes are running
- Designed Rest API in data ingestion for retention of GA data in order to optimize cluster space
- Added exception handling scenarios in data sync logic to fix multiple bugs
- Fix for missing PG data from Kafka for UMP panel - Created a new pipeline to ingest missing data from HDFS to ElasticSearch in case of cluster failure

▪ Near Real Time Data Pipeline – POC
-> Tech Stack – Java, Spark, Kafka, Datastax Cassandra, Datastax studio, Zookeeper, Maven
- Crafted a Cassandra based real time ingestion pipeline for marketplace data in order to help DWH team to reduce request load from production MySQL. The Objective was to shift business users from production, to overcome data leaks & security issues
- Interacted with different business users to know about their use cases, ingestion tables, PII data and built data models accordingly for faster insertion/updation of data
- Setup web interface Datastax Studio for users to query real time data from Cassandra using LDAP authentication

▪ Hive Query Parser
-> Tech Stack – Django, Python, NGINX
- Query Validator and Optimization Engine - Created a Django web application to parse and validate user's hive queries. In case of a bad query (missing partition columns/unbalanced joins), it also provides suggestions to improve the query - PII detector – Built a Django web application to detect all running hive queries which are fetching PII data.

Alert monitoring for Kaizala Alerts
Tech Stack - Microsoft Power Automate, COSMOS DB
1. Developed Alert notification flows using Microsoft Powerautomate to alert for business critical data in COSMOS DB container.
2. Integrated Microsoft Kaizala API to receive Real time Alerts in case of conditional failures.

Power BI business Dashboards for Real time business Data
Tech Stack - POWER BI, Microsoft SQL Server, COSMOS DB
1. Developed Power BI dashboards for Streaming data in COSMOS Container with Refresh Frequency of 4 hr.Used COSMOS DB SQL API to generate optimized Query.
2. DevelopedBusiness FTD and MTD Dashboards for Scheduled data in Microsoft SQL Server with Refresh Frequency of 2 hr.

Streaming Data Pipeline from COSMOS DB to Azure Data Explorer
Tech Stack - COSMOS DB change Feed, Azure Data Explorer SDK, Dot Net Core, C#
1. Developed COSMOS DB Real time data pipelines using Changefeed SDK to move data from COSMOS to ADX.
2. Used ChangeFeed SDK and Kusto SDK in Dot Net Core Environment.
3. Modeled NoSQL JSON data into tabular format in ADX using C# classes.

Real time Lead Acquisition Process using COSMOS ChangeFeed
Tech Stack - COSMOS DB change Feed, Azure App Service, Azure DevOps, Azure Repos, Dot Net Core, C#,
1. Worked on Azure Cosmos NoSQL DB changeFeed to develop Realtime processes for customer Acquision via Dot net SDK for more than 0.5M Data.

2. The Process captures the lead events from various sources like WEB, Mobile, POS to process it in Realtime Basis using microbatching pull model of Changefeed. The Data is filtered, deduped and eligible leads will be pushed to various Notification and dialer HTTP API. The Log data is stored in COSMOS DB and Microsoft SQL server.
3. Published the Process Webjobs in azure app services production environment.
4. Developed test cases and mockup data creation for Data Quality Management.
5. Implementation and administration of Git-based code repositories and branching/versioning standards.

Worked on Azure Cosmos NoSQL DB changeFeed to develop Realtime processes for customer Acquision via Dot net SDK for more than 0.3M Data.
Developed Dot Net Programs for data Subscription and calling various APIs.
Published the Webjobs in azure app services production environment.
Created Pipelines to process 5M+ Data from COSMOS DB to Azure ADX for KPI monitoring.
Developed PowerBI reports for MTD and FTD for RealTime Processes.
Experienced in Handling NoSQL data in C# code.

Worked on Azure Cosmos NoSQL DB changeFeed to develop Realtime processes for customer Acquision via Dot net SDK for more than 0.5M Data.
Developed and published Dot Net Programs for data Subscription and calling various APIs.
Created Realtime data Pipelines to process 5M+ Data from COSMOS DB to Azure Data Explorer for KPI monitoring.
Developed MTD and FTD PowerBI Dashboards using source as Microsoft SQL Server, NoSQL Databases for business KPIs.
Implemented Alert notification flows using Microsoft Powerautomate to alert for business critical data in COSMOS DB container.
Developed test cases and mockup data creation for Data Quality Management.
Implementation and administration of Git-based code repositories and branching/versioning standards.

Worked on Azure Cosmos NoSQL DB ChangeFeed to develop Realtime processes for customer Acquisition via Dot net SDK for more than 0.5M Data.
Developed and published Dot Net Programs for data Subscription and calling various APIs.
Created Realtime data Pipelines to process 5M+ Data from COSMOS DB to Azure Data Explorer for KPI monitoring.
Developed MTD and FTD Power-BI Dashboards using source as Microsoft SQL Server, NoSQL Databases for business KPIs.
Implemented Alert notification flows using Microsoft Power Automate to alert for business critical data in COSMOS DB container.
Developed test cases and mockup data creation for Data Quality Management.
Implementation and administration of Git-based code repositories and branching/versioning standards.

Contact Information

Name: John Doe

Email: johndoe@gmail.com

Phone: 555-555-1212

Professional Summary

Experienced data engineer with a strong background in programming and database management. Skilled in Python, SQL, and data visualization tools such as Tableau. Proven track record of building and maintaining efficient data pipelines for large-scale data processing.

Skills

Programming: Python, SQL, Java
Databases: MySQL, PostgreSQL, MongoDB
Data visualization: Tableau, Matplotlib
Big data technologies: Hadoop, Spark
Cloud platforms: AWS, Google Cloud

Work Experience

Data Engineer
ABC Company
Jan 2021 - Present

Designed and implemented a data pipeline to process and analyze terabytes of data on a daily basis
Developed machine learning models to forecast demand and optimize inventory management
Created dashboards in Tableau to visualize and communicate key metrics to stakeholders
Data Engineer
XYZ Company
Sep 2019 - Dec 2020

Collaborated with data scientists to design and implement data pipelines for a large-scale recommendation system
Maintained and optimized existing data pipelines, resulting in a 20% increase in performance
Developed scripts in Python to automate data cleaning and preparation tasks
Education

Bachelor of Science in Computer Science
University of California, Berkeley
Sep 2015 - May 2019

Projects

Data Pipeline for E-Commerce Website

Developed a data pipeline to extract, transform, and load data from various sources into a data warehouse. Used Python, SQL, and AWS technologies.

Machine Learning for Fraud Detection

Implemented a machine learning model to detect fraudulent transactions in real-time. Used Python, scikit-learn, and SQL.

Certifications

Data Engineering Certificate, Coursera
AWS Certified Solutions Architect - Associate

Python
ADF
ADB
PowerBI
MS SQL Server
Azure Data lake
COSMOS DB

Skills
C# - not complete - 8/10
.NET - Incomplete - API call / MVC/ App development / API development
COSMOS DB - incomplete knowledge , no demand in market - only SDK
APP service - No knowledge
Azure - no hands on
SQL - no
JAVA - little

POWERBI
PYTHON
ADB
ADF

Skills	Level	Env. Available	Required
C#	8/10	Course	
.NET	5/10	Course	
DSA	4/10	Course	
Azure	2/10	Need to create account	
Cosmos DB	5/10	Prod account	
ADB	4/10	Has UAT access	Required
ADF	4/10	Has UAT access	NO
Python	2/10	Yes	
SQL/ Synapse	2/10	Yes	
PowerBI	4/10	Has access	
KQL			

What you want to do ?
Azure / SQL/ Python(PySpark) / PowerBI / ADF/SSMS/SSIS

Cloud - Azure
Language - c# / Python
RDBMS - SQL server/ Synapse
NOSQL DB - Cosmos dB
ETL - ADF
Compute - ADB
BI - PowerBI

Domain	**Tools**	**What to learn**	**Status**
Cloud	Azure	DP- 203, AZ -104	DP 203
Language	C# .NET core	API, app dev, ASP.net core ,MVC , (build a strong web app) , API	NA
	Python		trendytech
RDBMS	SQL server/ synapse	Questpond sql server/ Synapse tutorial	SQL optimization
NOSQL	COSMOS DB	Connect with Ef core, optimization system	
Compute	ADB	Raja data engineering	Spark course
	PySpark	Online Video	
ETL	ADF	WAFA studies build etl pipeline	ADF tutorial
BI	PowerBI	Develop dashboards	
Devops	Azure Devops /networking	Questpond	

RapidLR Processor - Cosmos DB changefeed, App service, C#, .NET, GIT,

POWERBI - Error dashboard , SQL , DAX,

ADX - KQL queries, Kusto SDK

ADF - 1 pipeline

SQL Server -no exp

To move - Python / PySpark - ADB

Questions

1. Strong project explaination skills - its important
2. Datawarehousing concepts - Dimensions type,
3. ADF activities - polybase vs bulk, Dataflow, foreach inside foreach, activities in detail,
4. Databricks -
5. Hints in spark.
6. Huge table load to data lake in databricks. - partitioning while reading
7. Catalyst optimizer working
8. Predicate pushdown, partition pruning, projection - Pyspark: Partition Pruning, Predicate, and Projection Pushdown | by Justin Davis | Medium

---

1. Recursive cte
2. Self join
3. Join and then select in apache spark
4. CROSS APPLY , writing SP, FUNCTIONS, Table valued function,
5. Recursive CTE
6. DateDiff - for difference in month, days ,week

---

1. What is AQE , explain
2. What is Hash distribution
3. Broadcasting variable how to change value ?

ETL Pipeline Development using Azure Data Factory
•Developed automated Pipelines using ADF to Extract, Transform and load 50M+ data daily from various sources namely Azure SQL DB, Blob storage, Azure Synapse, Azure Data explorer, Azure COSMOS DB.
•Worked on creating dependent activities, Linked Services ,triggers in Azure Data factory.
•Transformed Semi-structured JSON and un-structured blob data to Structured format for data analysis.
Data Lakehouse integration using Azure Databricks
•Implemented ETL pipeline from Data lake to Azure SQL Datawarehouse using Databricks notebook. •Secured the secret keys using key vault and scheduled the job with 4 hr frequency.
•Good understanding of Spark Architecture including spark core, DataFrame, Pyspark, Driver Node, Worker Node, Stages, Executors and Tasks, Deployment modes, DAG and Python. Enterprise Data warehouse Optimization and Governance
•Optimized complex T-SQL queries using joins, stored procedures, triggers, user defined functions in SQL.
•Achieved Performance Tuning and Query Optimization of 98.3% by maintaining clustered and non clustured indexes, statistics, Error logging, Exceptional Handling.
•Good understanding of Relational Database Systems, Normalization, partitioning, data modeling.
Business Intelligence Unit Dashboard development
•Tabulated monthly to date (MTD) and FTD dashboards using Power BI and SQL warehouse, utilizing import query mode for scheduled data. Developed Real time dashboards using Direct Query mode.
•Utilized Power Query, measures, DAX Expressions, measures, Pie charts, Scatter plots for visualizations.
•Implemented Row level security and Email alert for monitoring purpose.
Real-time Lead Acquisition and Engagement Process •Acquired 2M+ customers daily by developing a streaming NoSQL Data product using Azure COSMOS DB change feed and c# .Net Core SDK.
•Implemented and administered Git-based code repositories and branching/versioning standards.

•Developed test cases and mockup data creation for data quality management.
•Integrated Microsoft Kaizala API with Power Automate to receive real-time failure alerts.
•Insured continuous up-time of projects with the help of an alert mechanism.

--_____

Behavioural questions

I will do better when no one bothering me .
Measures sociability -> team playing- > open to collaboration
-> Strongly disagree, disagree

I always take risk
I find routine boring
->

Be the best -> always give best answer
Communication skills
Honesty and integrity
Flexibility
Persistant
Loyal
Problem solving
Quick learner
Self motivated
Work effectively under pressure

_____

SALARY COMPONENTS	PROPOSED SALARY	
	Per Month	Per Annum
BASIC*	77,500	930,000
HOUSE RENT ALLOWANCE	38,750	465,000
CONVEYANCE ALLOWANCE		
FDA		
EDUCATIONAL ALLOWANCE	200	2,400
MEDICAL ALLOWANCE		
SPECIAL ALLOWANCE	32,092	385,104
LEAVE TRAVEL ALLOWANCE	6,458	77,500
GROSS SALARY	**155,000**	**1,860,004**
PF	9,300	111,600
GRATUITY	3,726	44,712
ESI	-	-
SAVINGS & RETIREMENT BENEFITS	**13,026**	**156,312**
PERFORMANCE LINKED BONUS (ONTARGET)	**15,500**	**186,000**
CTC	**183,526**	**2,202,316**

SALARY COMPONENTS	PROPOSED SALARY	
	Per Month	Per Annum
BASIC*	77,500	930,000
HOUSE RENT ALLOWANCE	38,750	465,000
CONVEYANCE ALLOWANCE		
FDA		
EDUCATIONAL ALLOWANCE	200	2,400
MEDICAL ALLOWANCE		
SPECIAL ALLOWANCE	32,092	385,104
LEAVE TRAVEL ALLOWANCE	6,458	77,500
GROSS SALARY	**155,000**	**1,860,004**
PF	9,300	111,600
GRATUITY	3,726	44,712
ESI	-	-
SAVINGS & RETIREMENT BENEFITS	**13,026**	**156,312**
PERFORMANCE LINKED BONUS (ONTARGET)	15,500	186,000
CTC	**183,526**	**2,202,316**

Employee Name	Saurabh Aher
Date of Joining	1-Apr-24
Position	Data Engineer

Salary Breakup (in INR)		
Particulars	**Per Month**	**Per Annum**
Basic Salary (50% of Base Pay)	79,167	950,000
HRA (50% of Basic Salary)	39,583	475,000
Special Allowance	37,783	453,400
Employer's Contribution to Provident Fund	1,800	21,600
Base Pay (Fixed pay)	**158,333**	**1,900,000**
Performance Bonus (Target)	-	190,000
Estimated Total Direct Compensation (TDC)		**2,090,000**
Other Benefits		
Gratuity (4.81% of Basic Salary)	3,808	45,695
Estimated Cost to Company		**2,135,695**

Important Note:

Performance Bonus
Bonus plan reflects our pay for performance compensation philosophy, as determined by the Company in its sole discretion, which will be subject to both your individual performance during the year and the performance of the Company. Amount is calculated off of annual earnings. New hires must have a start date on or before October 1 to participate in the bonus plan for that year.

Other Disclosures
Information regarding employment terms and compensation will be listed in the employment agreement document and signature will be required. This document is to provide an estimated calculation on total direct compensation only and does not take place of the employment agreement.

Additional Benefits
Group Medical Insurance Coverage of Sum Insured INR 5 lacs for Employee, Spouse, 2 children and parents.
Group Personal Accident Insurance coverage for the employee.
Paid leaves & Public holidays as per company policy.
Professional Tax & Income Tax will be subject to the applicable laws.
Employee PF contribution shall be part of employee's salary and shall be deductible therefrom.
Gratuity will be paid as per the statutes of the Payment of Gratuity Act.

Python

06 February 2023 02:10 PM

Databricks - Automated cluster management , Coding notebooks - Abstraction
Azure Databricks - Databricks hosted on azure.

Benefits of Azure Databricks -
Optimized spark engine , Machine Learning , ML Flow ,
Choice of language - .Net
Collaborative notebooks-
Delta Lake -
Integration with azure services -
Interactive workspaces
Enterprise grade security

DbUtils - helps us to perform powerful task include efficient object storage, chaining notebooks and working with secrets.

Integrating azure data bricks with azure blob storage .

Python tutorial

Variable - assigned when we use it

x= "name" / or x = 'name' : no difference in single or double quotes
x= 123 ; The variable type can be changed when we assign to different data.
x= str(123) ; we can cast also to specific data type
type(123) ; get the data type of variable

Text Type: str
Numeric Types: int, float, complex
Sequence Types: list, tuple, range
Mapping Type: dict
Set Types: set, frozenset
Boolean Type: bool
Binary Types: bytes, bytearray, memoryview
None Type: NoneType

String -

1. "Hello" is same as 'Hello'

2. """Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua."""
Multiline data using triple quotes

3. strings in Python are arrays of bytes representing unicode characters.

4. We can also loop through a string -> for x in "banana": print(x)
5. len(a) - length of the string
6. In and not in - check to use the strings
7. b[2:5] - string slicing ; end character is not included : indexes start from 0 ; -1 is last index
8. a.upper() - uppercase ,a.lower()
9. a.strip() - remove whitespaces
10. a.replace() - replace all the occurances with new string
11. a.split()
12. Enter int into the string
    age = 36
    txt = "My name is John, and I am {}"
    print(txt.format(age))
13. Escape character -
    "We are the so-called \"Vikings\" from the north."
14. String concat;
    Using + operator

Lists
thislist = ["apple", "banana", "cherry"]
print(thislist)

1. List items are ordered, changeable, and allow duplicate values, Any data type.

2. Can be created using constructors
   thislist = list(("apple", "banana", "cherry"))

   List - is a collection which is ordered and changeable. Allows duplicate members.
   Tuple - is a collection which is ordered and unchangeable. Allows duplicate members.
   Set - is a collection which is unordered, unchangeable*, and unindexed. No duplicate members.
   Dictionary - is a collection which is ordered** and changeable. No duplicate members.

3. Replace in list
   thislist = ["apple", "banana", "cherry", "orange", "kiwi", "mango"]
   thislist[1:3] = ["blackcurrant", "watermelon"]
   print(thislist)
   ['apple', 'blackcurrant', 'watermelon', 'orange', 'kiwi', 'mango']

4. Insert in List
   thislist = ["apple", "banana", "cherry"]
   thislist.insert(2, "watermelon")
   print(thislist)

5. Append in List
   thislist = ["apple", "banana", "cherry"]
   thislist.append("orange")
   print(thislist)

6. Add any data type (List ,set ,Tuple ,Dictionary) to list
   thislist = ["apple", "banana", "cherry"]
   thistuple = ("kiwi", "orange")
   thislist.extend(thistuple)
   print(thislist)

   Also simple add will also work
   list3 = list1 + list2

7. Remove first occurance of any specified element in list
   thislist = ["apple", "banana", "cherry"]
   thislist.remove("banana")
   print(thislist)

8. Remove specified index from the list, If not specified index it will remove the last index element
   thislist = ["apple", "banana", "cherry"]
   thislist.pop(1)
   print(thislist)

9. Delete an element or delete complete list
   del thislist[0] - delete specified element
   del thislist - delete complete list

10. Clears the list
    thislist = ["apple", "banana", "cherry"]
    thislist.clear()
    print(thislist)

11. Looping

a) For loop
   thislist = ["apple", "banana", "cherry"]
   for x in thislist:
     print(x)

b) Using indexes
   for i in range(len(thislist)):
     print(thislist[i])

c) List comprehension
   thislist = ["apple", "banana", "cherry"]
   [print(x) for x in thislist]

12. List comprehension
    newlist = [expression for item in iterable if condition == True]
    newlist = [x for x in fruits if x != "apple"]

13. Sort list
    thislist.sort() - sort alphabetically , if want reverse use - thislist.sort(reverse = True)
    Custom sort

    def myfunc(n):
      return abs(n - 50)

thislist = [100, 50, 65, 82, 23]
thislist.sort(key = myfunc)
print(thislist)

Case Insensitive sort - thislist.sort(key = str.lower)

14. Copy a list
    mylist = thislist.copy()
    mylist = list(thislist)

15. Count a specific element in list

    fruits = ['apple', 'banana', 'cherry']

    x = fruits.count("cherry")
    fruits.reverse()
    x = fruits.index("cherry") - return a first index of the element

Tuples :
Tuple items are ordered, unchangeable, and allow duplicate values.

1. thistuple = ("apple", "banana", "cherry", "apple", "cherry")
2. thistuple = ("apple",) # one item tuple, comma is important
3. Update tuple -
   Change the tuple into list and make changes - again convert to tuple
4. del thistuple - delete the tuple

Sets:

A set is a collection which is unordered, unchangeable*, and unindexed. -(you can remove and add new items)
They cannot be referred by indexes. They don't allow duplicates

thisset = {"apple", "banana", "cherry"}
1. Set.add() - add new items
2. Set.update(set) - add any iterable in set
3. thisset.remove("banana") - remove a specific item
4. thisset.pop() - remove a random element
5. thisset.clear() - clears the set
6. del thisset - deletes the set
7. thisset.discard("banana") - discard works same as remove but raise no error if element is not present
8. The values True and 1 are considered the same value in sets, and are treated as duplicates:

Dictionary

A dictionary is a collection which is ordered*, changeable and do not allow duplicates.

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

1. thisdict["brand"] - accessing the value from the key // thisdict.get("model")
2. thisdict = dict(name = "John", age = 36, country = "Norway") - declaring using constructor
3. x = thisdict.keys() - list of keys
4. thisdict.values() - list of values
5. car["color"] = "red" - add new key value to dict / also for updating
6. x = thisdict.items() - The items() method will return each item in a dictionary, as tuples in a list.
7. thisdict.update({"year": 2020}) - update and add the dict
8. thisdict.pop("model") - remove specific item from dictionary / del thisdict["model"]
9. del thisdict - delete dictionary
10. thisdict.clear() - empties the dictionary
11. for x, y in thisdict.items():
      print(x, y)
    Printing both keys and values
12. Copy a dict - mydict = thisdict.copy()
    mydict = dict(thisdict)
13. print(myfamily["child2"]["name"]) -access in nested list

14. Sorting by value first and then by keys
    sorted_dict = dict(sorted(my_dict.items(), key=lambda item: (item[1], item[0])))

    For reverse sorting -
    sorted_dict = dict(sorted(my_dict.items(), key=lambda item: (-item[1], item[0])))

Functions :

1. Function call -
   def my_function(fname, lname):
   print(fname + " " + lname)
2. Pass any number of arguments - it will take tuple of argument
   def my_function(*kids):
   print("The youngest child is " + kids[2])
3. Default value pass
   def my_function(country = "Norway"):
   print("I am from " + country)

4. Lambda function (No name functions)
   x = lambda a, b : a * b
   print(x(5, 6))

5. Map Function :returns a map object(which is an iterator) of the results after applying the given function to each item of a given iterable (list, tuple etc.)

   numbers = (1, 2, 3, 4)
   result = map(lambda x: x + x, numbers)
   print(list(result))

   numbers1 = [1, 2, 3]
   numbers2 = [4, 5, 6]

   result = map(lambda x, y: x + y, numbers1, numbers2)
   print(list(result))

6. print(i, end = " ") - to print on same line

Class

```
class MyClass:

    x = 5
```

Object

```
p1 = MyClass()
print(p1.x)
```

__init__() - its a constructor in python -

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

p1 = Person("John", 36)

print(p1.name)
print(p1.age)
```

__str__() Function - return the string representation of object

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def __str__(self):
    return f"{self.name}({self.age})"

p1 = Person("John", 36)

print(p1)
```

Self parameter
The self parameter is a reference to the current instance of the class,
and is used to access variables that belongs to the class.
It does not have to be named self , you can call it whatever you like,
but it has to be the first parameter of any function in the class:

Child class

```
class Student(Person):
  pass
```

Pandas library

1. Series
   import pandas as pd
   a = [1, 7, 2]
   myvar = pd.Series(a)
   print(myvar)

Series is like a column, a DataFrame is the whole table.

2. Dataframe

```
data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

df= pd.DataFrame(data)

print(myvar)
```

   a. df.loc[0] - return a single row as series
   b. df.loc[[0, 1]] - list 2 rows as dataframe
   c. df = pd.read_csv('data.csv') - reading csv
   d. pd.read_json('data.json') - reading json
   e. df.head() - returns the 5 five elements / df.tail() - return the last 5
   f. df.dropna() - remove empty cells

Use help to explore any function

Main Data Types	Data Types	Pyspark Data Type
Numeric	Byte	ByteType()
	Short	ShortType()
	Integer	IntegerType()
	Long Type	Long TypeType()
	Float	FloatType()
	Double	DoubleType()
	Decimal	DecimalType()
String	String	StringType()
Binary	Binary	BinaryType()
Boolean	Boolean	BooleanType()
Datetime	Timestamp	TimestampType()
	Date	DateType()
Complex	Array	ArrayType()
	Map	MapType()
	Struct	StructType()
	StructField	StructFieldType()

Spark - sparksession objects

1. Spark.createDataFrame(data, schema)
2. Spark.read.format('csv').load()
3. spark.read.csv - reading csv files

Python OOP

1. Class

```
class person:
  Teeth = 32   #Class Variable - unique for all instance of class
  def __init__(self, name):  #Constructor
    self.name= name  # Instance Variable - different for different instance
    self.age = 0

  def show(self):
    print(self.name,self.age)
```

```
 def __str__(self):
     return f"Name: {self.name} | Age: {self.age} | {self.Teeth}"
 def __del__(self):
     print("Object Deleted")

p = person("vaibhav")
print(p)
p.Teeth= 35
print(p.Teeth)
print(person.Teeth)
del p


class Employee(Person):  # Inheritance
 def __init__(self,name, company):
   super().__init__(name)  # Super Keyword
   self.company = company

E = Employee("Vikas", "Airbnb")
print(E)
```

2. Inheritance

Types of Inheritance
Super() - used instead of class Name

Questions -

1. List vs tuple - List is mutable, tuple immutable

2. First Class Objects
   In Python, functions are first class objects which means that functions in Python can be used or passed as arguments.
   Properties of first class functions:
   A function is an instance of the Object type.
   You can store the function in a variable.
   You can pass the function as a parameter to another function.
   You can return the function from a function.
   You can store them in data structures such as hash tables, lists, …

3. Decorator -
   Function that takes function , changes it and returns a function
   It is used to modify the behaviour of a function or class.

4. Python memory management

   Stack -> only reference
   Private heap memory -> actual object
   Garbage collector - > automatic

5. __init__.py - its a python package , constructor of package
   __init__() - constructor in python

6. Module vs Package -
   Module - files
   Package - folder

7. Range vs xrange
   Xrange - return a generator object

8. Ternary operator
   n if n >10 else 10

9. Pickling - accept python object and transform into string representation and dump into file. - serialization
   Unpickling - retrieving data from stored string into python object. - deserialization

   Import

Alphanumeric character check

isalnum()

Try except

```
try:
  print(x)
except NameError:
  print("Variable x is not defined")
except:
  print("Something else went wrong")
```

Regex - >

Datetime -

```
d1 = dt.datetime.strptime(t1, "%a %d %b %Y %H:%M:%S %z")
d2 = dt.datetime.strptime(t2, "%a %d %b %Y %H:%M:%S %z")
return str(int(abs(d1- d2).total_seconds()))
```

```
A = [int(x) for x in input().split()]
```

```
set(map(int, input().split()))
```

package in python , pickle , dump,

POWERBI Training

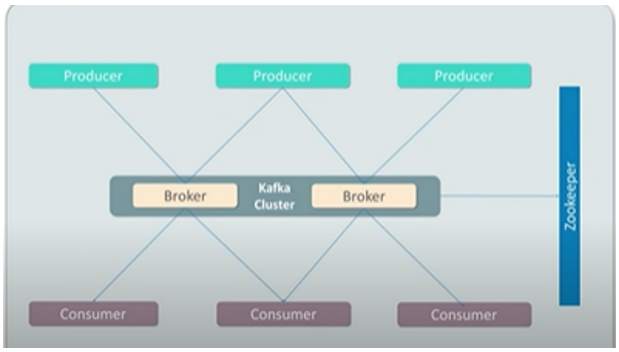
01 December 2021 11:19 AM

Power Query
Power Pivot
Power View
Power BI Pro

Apache kafka

22 June 2022 12:10 PM

1. Kafka clustor -



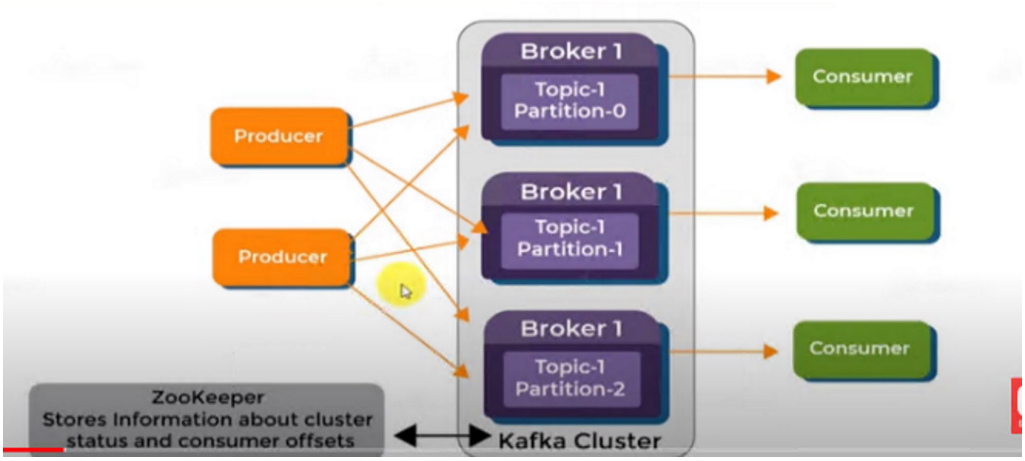
2. Storage in Log - Retention

Begin - >

Multiple frontend and backend servers and they wish to communicate.
Rise with huge data pipelines.
Kafka helps us in this scenerio. It's a messaging service.

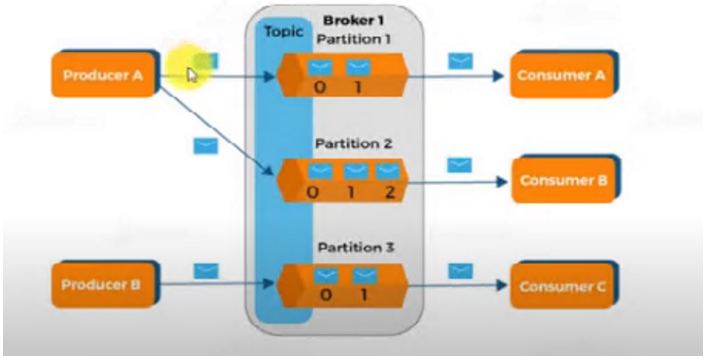
Producers (frontend server produce data) and sent to kafka cluster. Kafka cluster stores the data in the streaming form and then consumer consumes the data.
Kafka - Pub sub system , High throughput , Asynchronous communication,
Reduces the data pipeline communication
Scalability - highly scalable distributed system with no downtime - > 1 cluster - multiple brokers - > multiple node
Fault tolerance - replica of nodes
Reliability
-Distributed - distributed commit log.
Performance - High throughput
Zero Downtime and Zero data loss -
Extensible -

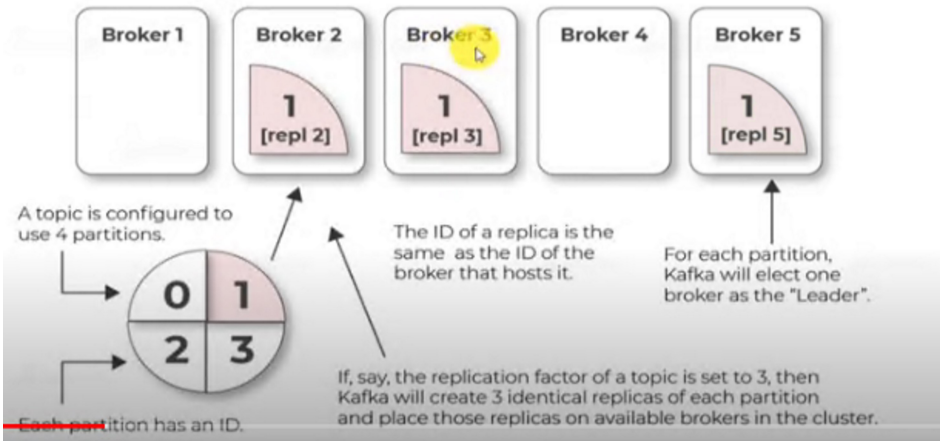
Brokers - Server that manage conversation between 2 system
Message - byte array - JSON , string, avro
Topic - all messages are maintained in 1 store called topic , Ex. Sales topic, Marketing topic
Cluster - Set of brokers is called as cluster
Producers - frontend entity that generate data
Consumers - processes that consume data by subscribing to 1 or more topic in cluster
Partitions - Every brokers holds few partitions - topics are divided into partitions -



Kafka cluster architecture

Offset - each msg is tagged by broker with an offset - > 0,1 , 2 - and stored in the disk . Messages are written and stored in append only pattern





Consumer group - many consumer consuming same topic. Group assures that 1 partition will be consumed by only 1 consumer.
Kafka zookeeper -
manages any distributed application
Naming service - we can identify which broker is present and which cluster is working.

Kafka workflow -
Consumer subscribe to a particular topic.

Kafka cluster - multiple replica of topics

1. Single node - single broker cluster -
2. Single node -multiple broker cluster
3. Multi node multi broker cluster

KAFKA -
Messaging System

Pub/ Sub Model

Publisher - > Broker -> Subscriber

There can be one or more brokers in Kafka cluster

Cluster (Server) -> Broker -> Topic -> Topic Partitions (in different VMs)
Topic act as DB table- Classification

Offset - when message arrives in partition, it will be stored with a number, its called as offset.

Consumer - reading message from partition - >
Consumer group - unit of consumers for reading message from multiple partitions partitions.
Any consumer can talk to any partitions.

Zookeeper - track status of all kafka components

Big Data

02 November 2022 02:50 PM

1. Big data - volume , variety, velocity, veracity

Storage - Process - Scale

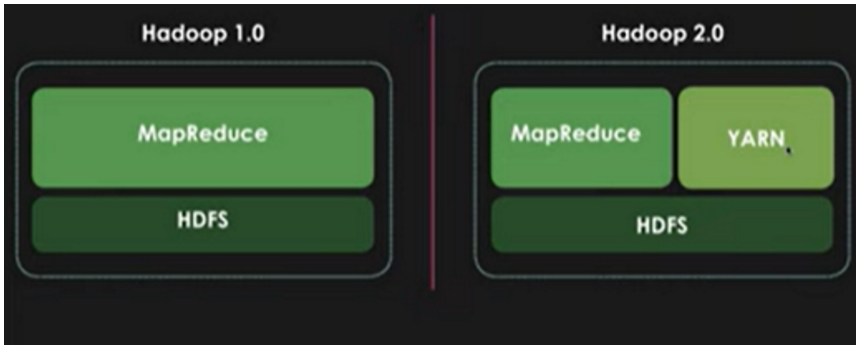
Monolithic - single powerful server - *vertical scaling*
Distributed - many small nodes coming together to create cluster - *Horizontal scaling*

Resources -
RAM - memory | Hard Disc - storage | CPU - compute

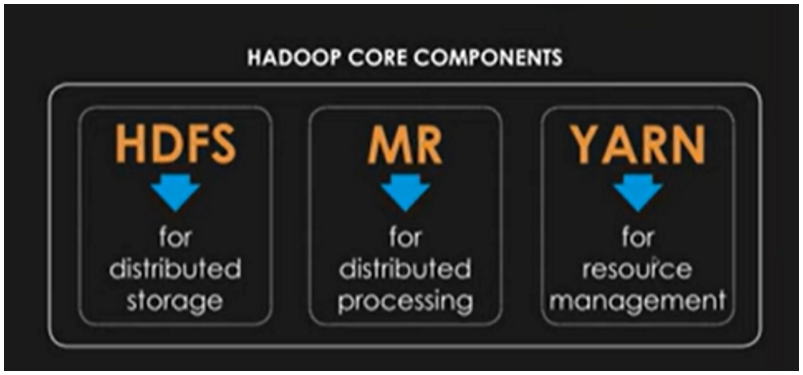
Hadoop is the framework to solve big data problems

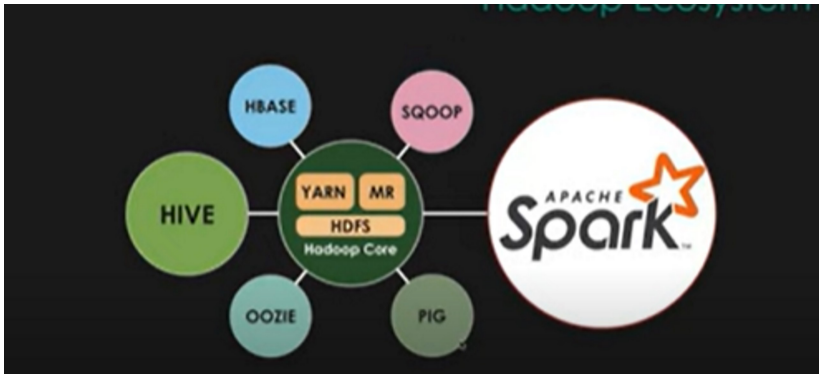
GFS - google file system paper published by google for storage
Mapreduce - google paper for compute

HDFS - yahoo implemented the GFS and changed to HDFS
MapReduce - yahoo implemented the MapReduce



YARN - Yet Another Resource Negotiator - Resource manager in a distributed system

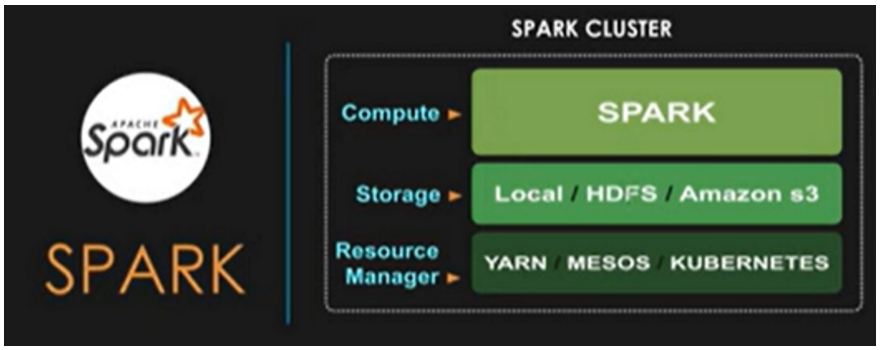




MapReduce - used in Java
HIVE - SQL wrapper on MapReduce - basically HQL
SQOOP - Transfer the data from traditional DB to HDFS - data migration - MapReduce Job where only mappers work
OOZIE - a workflow scheduler to manage Hadoop job
HBASE - column oriented NoSQL DB - run on top of HDFS

SPARK - general purpose in memory compute engine

Spark need any storage and any resource manager -



Spark is written in scala -

HDFS -
Block size - 128 mb - Data is stored in the blocks
Master Node - Name Node - stores the metadata where the data is stored.
Slave Node- Data Node - stores the actual data

Replication factor - basic 3 - stored on different locations

Replication factor -
Heart Beat -
_____-
MapReduce - has 2 phases
Map and Reduce -

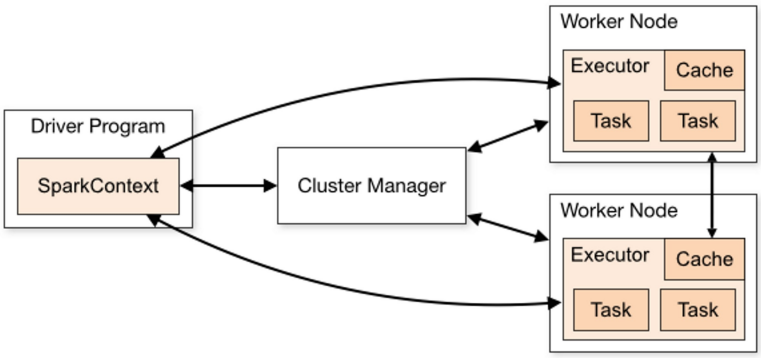
(K,V) -> MAP -> (K,V) -> REDUCE-> (K,V)
Traditional programming works when data is kept on single machine.
MapReduce is a compute which works on data on multiple machine.

The code will also be distributed in different blocks where the data is kept. Means the code will go to data and process the data there itself.. This is called as data locality.

Map - The code in each block . Mappers run parallelly on the blocks.
The output of each block will be aggregated and the aggregated output is formed by reducer code .
Reduce - aggregation code.

Record Reader- convert the raw data into key value pair

Apache Spark ---



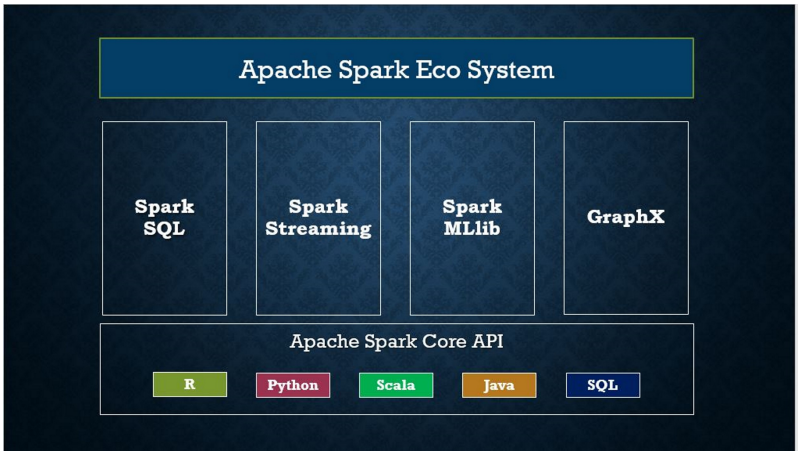
Apache Spark works in a master-slave architecture where the master is called “Driver” and slaves are called “Workers”. When you run a Spark application, Spark Driver creates a context that is an entry point to your application, and all operations (transformations and actions) are executed on worker nodes, and the resources are managed by Cluster Manager.

Cluster manager

As of writing this Apache Spark Tutorial, Spark supports below cluster managers:

- Standalone – a simple cluster manager included with Spark that makes it easy to set up a cluster.
- Apache Mesos – Mesons is a Cluster manager that can also run Hadoop MapReduce and Spark applications.
- Hadoop YARN – the resource manager in Hadoop 2. This is mostly used, cluster manager.
- Kubernetes – an open-source system for automating deployment, scaling, and management of containerized applications.
- local – which is not really a cluster manager but still I wanted to mention as we use “local” for master() in order to run Spark on your laptop/computer.

Spark web ui and spark history server



RDD - Resilient distributed data set - fundamental data structure in spark
RDD, DataFrame and Dataset - 3 data structure

DS Q

26 August 2023 11:01 AM

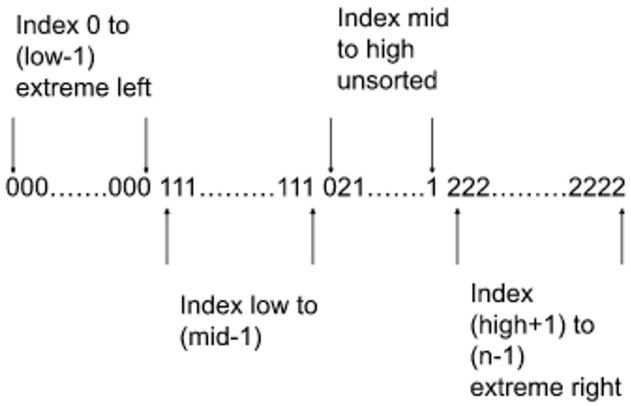
1. Pascal triangle

2. Next permutation
   [1,2,3], [1,3,2], [2, 1, 3], [2, 3, 1], [3,1,2], [3,2,1] - find the next permutation

   [1,5,4,3,2] - >[2 1 3 4 5]
   Find the breakpoint from right where t-1 > t -> 1
   If the breakpoint index is -1 then reverse array and return
   Else Replace the breakpoint with smallest element from right array which is greater than breakpoint
   -> replacing 1 & 2 -> 2 5 4 3 1
   Reverse the right array -> 2 1 3 4 5

3. Max subarray sum - Kadane algorithm
   nums = [-2,1,-3,4,-1,2,1,-5,4] - > 6
   Declare maxsum and currsum with value num[0]
   For the elements from 1 to len(num)
   currentSum = max(num, currentSum + num)
   maxSum = max(maxSum, currentSum)

4. Sort an array of 0s, 1s and 2s - (Dutch National flag algorithm)
   low=0
   mid=0
   high=len(nums)-1




```
while(mid <=high):
            if nums[mid]==0:
                nums[mid], nums[low] = nums[low], nums[mid]
                mid+=1
                low+=1
            elif nums[mid]==1:
                mid+=1
```

```
            elif nums[mid]==2:
                nums[mid], nums[high] = nums[high], nums[mid]
                high-=1
```

5. Stock Buy And Sell - We can maintain a minimum from the start of the array and compare it with every element of the array, if it is greater than the minimum then take the difference and maintain it in maxProfit,

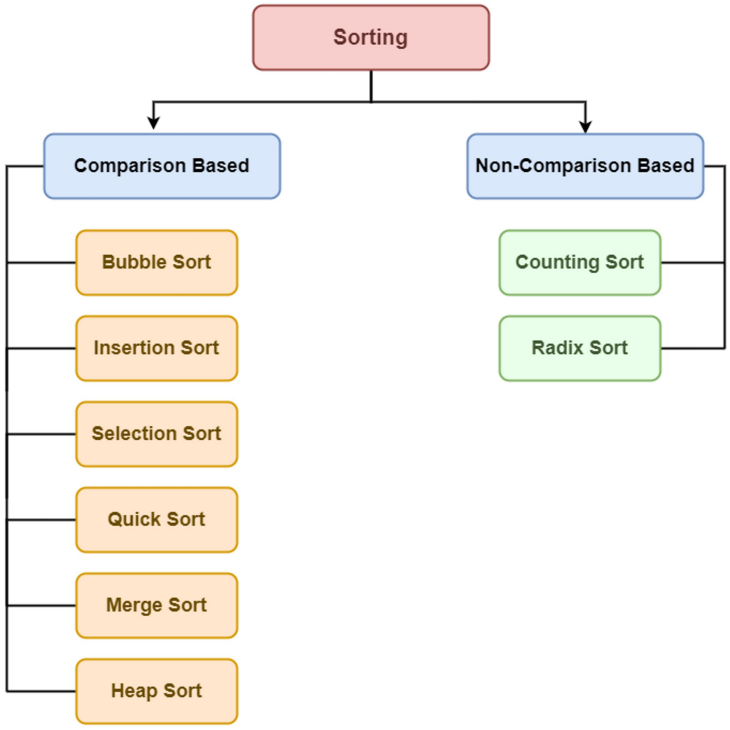
```
min = 0
   max = 0
   maxProfit = 0
   for i in range(1, len(prices)):
     if prices[i]< prices[min]:
       min = i
     if (prices[i]- prices[min]) >= maxProfit:
       max = i
       maxProfit = prices[i] -prices[min]
   return maxProfit
```

6. Rotate matrix by 90 degree -
   Take transpose and reverse the arrray

7. Merge Overlapping subintervals :
   Sort the list first and then use logic to compare the last and first element, if its less, then take max of both last element, else add the array.
   Sample code -

```
intervals.sort(key=lambda x: x[0])
    ml =[]
    ml.append(intervals[0])
    a= 0

    for i in range(1, len(intervals)):
      if(intervals[i][0] <= ml[a][1]):
        ml[a][1] =max(intervals[i][1], ml[a][1])

      elif intervals[i][0] > ml[a][1]:
        ml.append(intervals[i])
        a+=1
    return ml
```

Sorting



Name	Technique	Best Case	Average Case	Worst Case	Memory	Stable	Method Used
Quick Sort		n log n	n log n	n^{2}	log n	No	Partitioning
Merge Sort	dividing an array into smaller subarrays, sorting each subarray, and then merging the sorted subarrays back together to form the final sorted array.	n log n	n log n	n log n	n	Yes	Merging
Heap Sort		n log n	n log n	n log n	1	No	Selection
Insertion Sort	Values from the unsorted part are picked and placed at the correct position in the sorted part.	n	n^{2}	n^{2}	1	Yes	Insertion
Tim Sort		n	n log n	n log n	n	Yes	Insertion & Merging
Selection Sort	Find minimum and place at beginning	n^{2}	n^{2}	n^{2}	1	No	Selection
Shell Sort		n log n	n^{4/3}	n^{3/2}	1	No	Insertion
Bubble Sort	swapping the adjacent elements if they are in the wrong order, not used as its having high time complexity	n	n^{2}	n^{2}	1	Yes	Exchanging
Tree Sort		n log n	n log n	n log n	n	Yes	Insertion
Cycle Sort		n^{2}	n^{2}	n^{2}	1	No	Selection
Strand Sort		n	n^{2}	n^{2}	n	Yes	Selection
Cocktail Shaker Sort		n	n^{2}	n^{2}	1	Yes	Exchanging
Comb Sort		n log n	n^{2}	n^{2}	1	No	Exchanging
Gnome Sort		n	n^{2}	n^{2}	1	Yes	Exchanging

Odd–even Sort		n	n^{2}	n^{2}	1	Yes	Exchanging

8. Merge sorted arrays
   Compare each element and sort the array same like merge sort

9. find the duplicate in integer array [1….n]
   scanning the array from left to right, we set the item to its negative value.

```
// Visited
public static int findDuplicate_mark(int[] nums) {
    int len = nums.length;
    for (int num : nums) {
        int idx = Math.abs(num);
        if (nums[idx] < 0) {
            return idx;
        }
        nums[idx] = -nums[idx];
    }

    return len;
}
```

10. Search in 2D sorted array - >
    Use binary search

11. Pow(x,n)

12. Majority element - moore's voting algorithm

    Initialize count and val, if count>

---

1. Sliding window - In a sliding window, the two pointers usually move in the same direction will never overtake each other. This ensures that each value is only visited at most twice and the time complexity is still O(n).
   Use Hashmap or dictionary

   Q1. Contains Duplicate II -> calculate frequency using dictionary
   Q2. Min size subarray sum -> calculate the window sum and minimize window as per sum
   Q3. Fruits in basket -> Calculate the longest array with 2 distinct element
   Take a dictionary, add elements and frequency , if the size more than 2 , take

snapshot and reduce the elements until the window becomes valid again with 2 elements.

Q4. Best time to buy and sell the stock - We can maintain a minimum from the start of the array and compare it with every element of the array, if it is greater than the minimum then take the difference and maintain it in maxProfit,

Q5. Longest Substring Without Repeating Characters -> Find the longest string with unique character, use list as set and maintain unique values in last, as soon as you encounter repeating element, remove the characters till that element is not present in list.

Q6. Maximum Number of Vowels in a Substring of Given Length - take fixed size window and calculate total vowels, remove last element and add new element, count the vowels every time

Q7. Replacement Q pending

---

1. Two sum - use hashmap or dictionary to store elements
2.

Spark Optimization

22 October 2023 07:29 PM

Internal Working of Apache Spark

Lets say you have a 20 node spark cluster

Each node is of size - 16 cpu cores / 64 gb RAM

Let's say each node has 3 executors,
with each executor of size - 5 cpu cores / 21 GB RAM

=> 1. What's the total capacity of cluster?

We will have 20 * 3 = 60 executors
Total CPU capacity: 60 * 5 = 300 cpu Cores
Total Memory capacity: 60 * 21 = 1260 GB RAM

=> 2. How many parallel tasks can run on this cluster?

We have 300 CPU cores, we can run 300 parallel tasks on this cluster.

=> 3. Let's say you requested for 4 executors then how many parallel tasks can run?

so the capacity we got is 20 cpu cores
84 GB RAM
so a total of 20 parallel tasks can run.

=> 4. Let's say we read a csv file of 10.1 GB stored in datalake and have to do some filtering of data, how many tasks will run?

if we create a dataframe out of 10.1 GB file we will get 81 partitions in our dataframe. (will cover in my next post on how many partitions are created)

so we have 81 partitions each of size 128 mb, the last partition will be a bit smaller.

so our job will have 81 total tasks.
but we have 20 cpu cores

lets say each task takes around 10 second to process 128 mb data.
so first 20 tasks run in parallel,
once these 20 tasks are done the other 20 tasks are executed and so on...

so totally 5 cycles, if we think the most ideal scenario.

10 sec + 10 sec + 10 sec + 10 sec + 8 sec

first 4 cycles is to process 80 tasks all of 128 mb,
last 8 sec is to process just one task of around 100 mb, so it takes little lesser but 19 cpu cores were free during this time.

=> 5. is there a possibility of, out of memory error in the above scenario?

Each executor has 5 cpu cores and 21 gb ram.
This 21 gb RAM is divided in various parts -
300 mb reserved memory,
40% user memory to store user defined variables/data. example hashmap
60% spark memory - this is divided 50:50 between storage memory and execution memory.

so basically we are looking at execution memory and it will be around 28% roughly of the total memory allotted.

so consider around 6 GB of 21 GB memory is meant for execution memory.
per cpu core we have (6 GB / 5 cores) = 1.2 GB execution memory.
That means our task can roughly handle around 1.2 GB of data.
however, we are handling 128 mb so we are well under this range.

Resource level optimization

10 node cluster - 10 worker nodes , each has 16 core, 64 Ram
(1 core alloted for background processor , 1 gb for operating system)

Executor - container of resources , it holds RAM and cpu core , JVM

1 node can hold more than 1 executor -

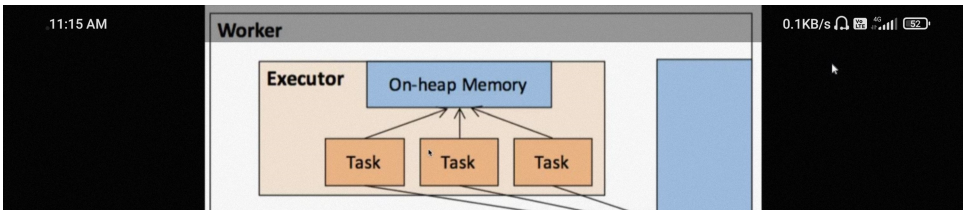
How to create executor from node - two options

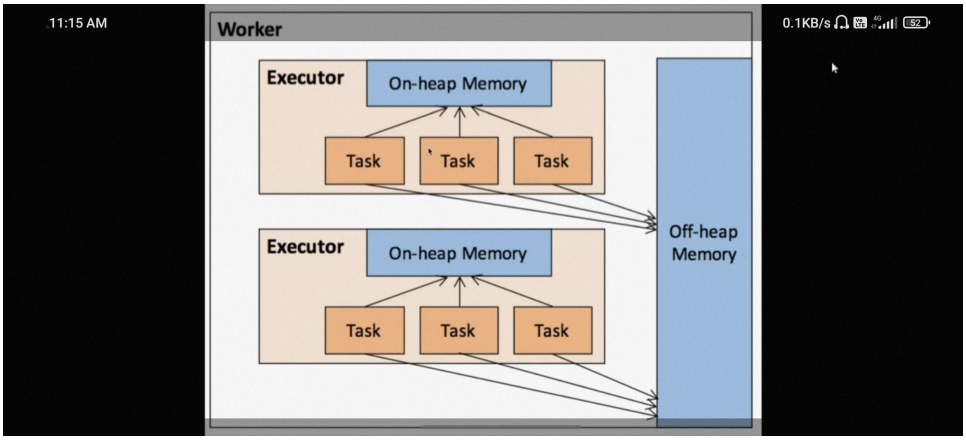
1. Thin executor - minimum core -
   16 executor - 1 core, 4 gb ram

   Drawback
   No multithreading possible in executor.
   Lot of copies will be required for broadcasting -

2. Fat executor - maximum core -
   1 executor - 16 core, 64 gb ram

   Drawback
   If the executor holds more than 5 cpu core then hdfs throughput suffers
   If the executor holds very huge memory , then garbage collection takes lot of time





10 node cluster - 10 worker nodes , each has 16 core, 64 Ram
(1 core alloted for background processor , 1 gb for operating system)

15 core, 63 gb ram

We can have 3 Executor each 5 core, 21 gb
off heap memory - Out of 21 gb some will be gone as off heap memory overhead - this is not part of executor
Off heap -max (384 mb, 7% of executor memory) - (384 mb, 1.5 gb) = 1.5 gb
For optimization memory

19 gb (on heap memory)

3 executor ~ = 5 core, 19 gb

10 node cluster - 30 executor (5 core, 19 gb (on heap memory))

1 executor out of 30 will be given for YARN application manager

29 executor left

1 core - 1 task can run

Vcpu - 2 * cpu =

5 worker node
8 cpu core, 32 gb
8 core , 24 usable

Vcpu = 2 * cpu

16 core, 24 usable
Out of 16, 12 can be used for executors , rest are for other purpose

5 worker node each 12 vcpu core, 24 usable

YARN and Ambari setup

Scp - secure copy - scp sourcepath destinationpath

One driver

Df.as[caseClass]

Azure Data Factory

23 August 2023 02:11 PM

Integration Runtime

1. Azure IR - fully managed ,serverless compute, elastic scaling
   Can be used for Data flow, copy and transform activity
   It is auto generated when you create ADF with location - auto resolve
   Need for creating new IR - If you want to have IR in specific region or you want to group IR
   IR is required for creating Linked service

2. Self Hosted IR -
   Can be used for copy and transform in cloud store, private network or in on premise
   It should be installed in on premise machine or in VM in private network
   From cloud(public) to on prem (private) -
   Node - 1 machine /VM - 4 node per self hosted IR is allowed
   For setup - create Self hosted IR in azure, download the software in VM and.
   Shared Self hosted IR - You can share IR for multiple Data factory , For this you have to add Linked Self hosted IR in ADF which you want link.

IR type	Public Network Support	Private Link Support
Azure	Data Flow Data movement Activity dispatch	Data Flow Data movement Activity dispatch
Self-hosted	Data movement Activity dispatch	Data movement Activity dispatch
Azure-SSIS	SSIS package execution	SSIS package execution

Parameterize Linked Service, Dataset in ADF

If you have 10 SQL server and you want to use it in ADF, the general way is to create 10 linked service.
But you can parameterize linked service.
You can parametrize - trigger, pipeline, dataset, linkedservice,

System Variable -

@pipeline().DataFactory Name of the data or Synapse workspace the pipeline run is running in
@pipeline().Pipeline Name of the pipeline
@pipeline().RunId ID of the specific pipeline run

Connectors -

https://learn.microsoft.com/en-us/azure/data-factory/connector-overview

File Format

Binary format
Delimited Text
Json format
Avro format
Parquet format
ORC format

Triggers

Schedule Triggers - schedule at any interval
Tumbling window Trigger - wants data between specific time slices , cyclic intervals, it can pull past and future data as well
windowStartTime and windowEndTime -
A schedule trigger can only trigger future dated loads. But tumbling window triggers can be configured to initiate past and future dated loads.
Schedule pipelines and triggers have a many-to-many relationship. Whereas a tumbling window trigger has a one-to-one relationship with a pipeline and can only reference a single pipeline.
Tumbling window trigger has a self-dependency property which means the trigger shouldn't proceed to the next window until the preceding window is

successfully completed.

Event Based Trigger - response to blob events

Copy Activity

DIU - compute power in unit
Degree of copy parallelism - max connections for read /write
Fault tolerance - in case of incompatible data
Enable staging - source --> staging ---> sink (temporary middle storage)
Wildcard path - specify files ".txt"
Max concurrent connection - The upper limit of concurrent connections established to the data store during the activity run.
Specify a value only when you want to limit concurrent connections.

Monitor

Delete - for deleting the files

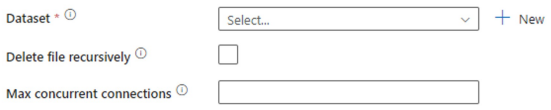
Done with demo
TO DO-> Copy Data from Blob csv to azure sql db

Costing of ADF

1. Orchestration and execution - activity run , trigger execution, debug run
2. Data movement - copy - per DIU hour
3. Pipeline activity - Pipeline activities execute on integration runtime. Pipeline activities include Lookup, Get Metadata, Delete, and schema operations during authoring (test connection, browse folder list and table list, get schema, and preview data).
4. External pipeline - ADB, Hdinsight ,

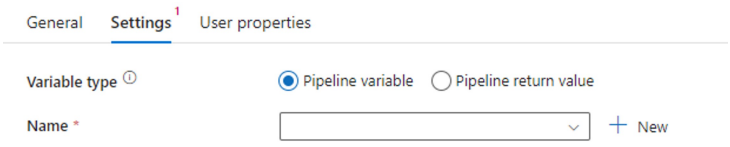
Data flow execution and debug
DataSet read/write/ monitoring ,
Inactive pipeline - no run or trigger in month

Delete Activity
To delete any data
It will generate the log file



Variables

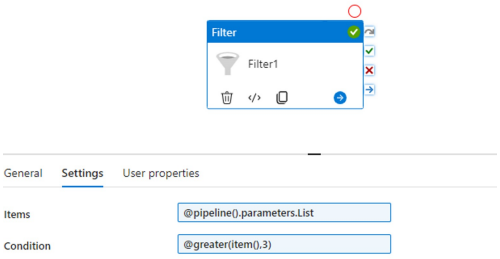
Parameter - you can pass from triggers or from master pipeline
Variable - can be set inside a pipeline - or using set variable
User properties - user defined properties for pipeline data - you can see it in monitor row, max 5 user properties



Execute Pipeline Activity -
Allows activity to invoke another pipeline

Filter activity -

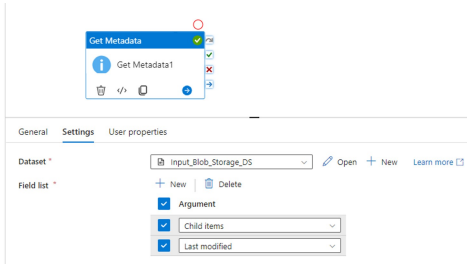
Give a list of items and filter condition , it will filter the output,



ForEach Activity - use same pipeline for many use cases
Ex. Copy same file to multiple destination folders.
@item () forEach iterator

GetMetaData Activity -

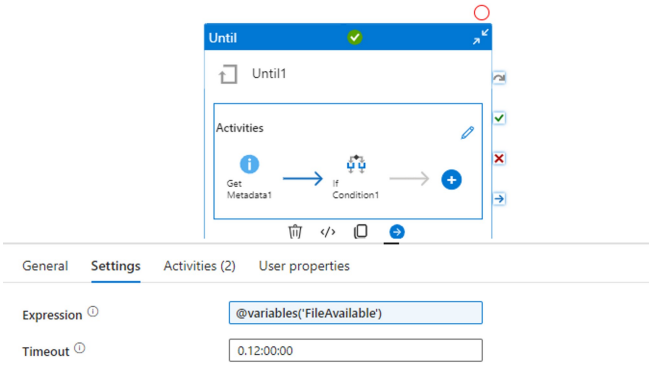
Metadata of files , folders , doesn't go recursively
Values - Item Name, ItemType, LastModified, Exists,



IF activity

Wait activity - give specified wait

Until - Do while loop - Activity will continue till the expression become true



WEB activity - Asynchronous - you can make rest API calls using WEB activity

WEBHOOK activity - synchronous API call. It will wait for the callback

SWITCH activity - switch between activities , what to do first

Validation Activity - it will work when the attached dataset reference exists
It will stop only when it finds the file or timeout occurs , GetmetaData Can be also use for validation and with exists value
Lookup Activity - retrieve a dataset, stored procedure , execution

Transform Data

DATA FLOW

- data transformation logic in graphical approaches
For data transformation we generally use Spark , U-SQL .
Same can be done in Data flow. It will be executed in ADB cluster.
You can run Data flow in pipeline using Data flow activity
Mapping Data flow - SQL operations using GUI
Wrangling Data flow - Power Query online editor

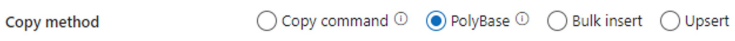
What is polybase ?
Data loading in synapse

1. Single gated load - load to control node first (bcp(bulk copy), sqlBulkCopy) - useful for small data movement.
2. Parallel load (Polybase, Copy) - MPP - directly to compute load - very efficient for large data

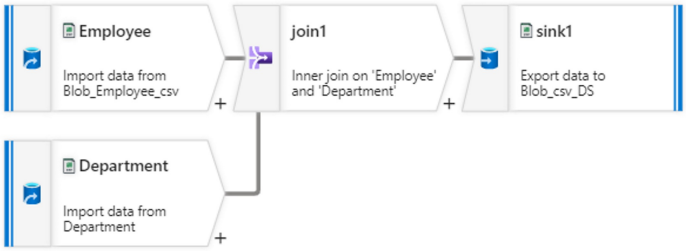
   Polybase -
   Its used when we want to move data to azure synapse
   It will load data to azure blob first and then create a table in synapse.
   Then the data will be transferred from 1 synapse table to another.
   CTAS - for importing data to synapse
   CETAS - for exporting data from synapse to other

   Copy - COPY INTO without creating table ,no need strict control permission , expect table to be existing
   Copy is superior to polybase

   if you want to load data from a data store that is already in Blob storage, you can use the COPY statement in Azure Synapse Analytics to load data directly into Azure Synapse Analytics. This is known as direct data movement. No staging blob is required



Mapping Data flow in ADF



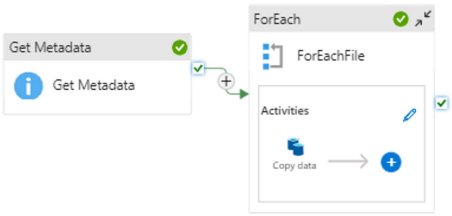
Real time scenerios

---

1. Copy Files dynamically from one folder to another

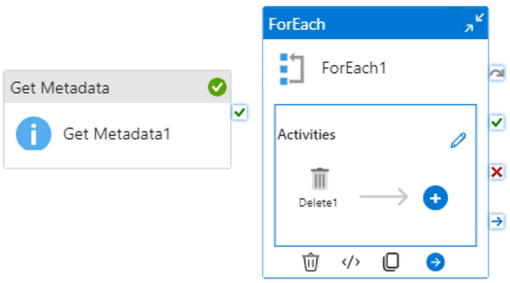
Solution - Get the File names using GetMetaData - loop through each file name using foreach and copy to sink
Use Item().name in for loop

```
"childItems": [
    {
        "name": "csvdemo.csv",
        "type": "File"
    },
    {
        "name": "dataInput.txt",
        "type": "File"
    }
```

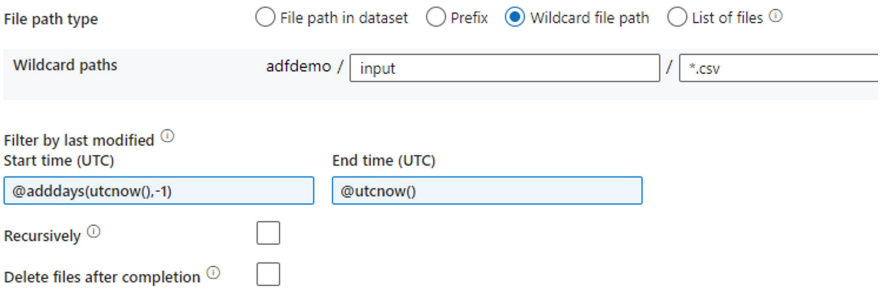



2. Delete old files from storage account using ADF

   Solution - Get lastModifiedDate of older files using GetMetaData and then perform delete using foreach



3. Incrementally copy and load the files using only copy activity



4. Copy file when file becomes available

   Solution - Use Validation activity to check if the dataset is present

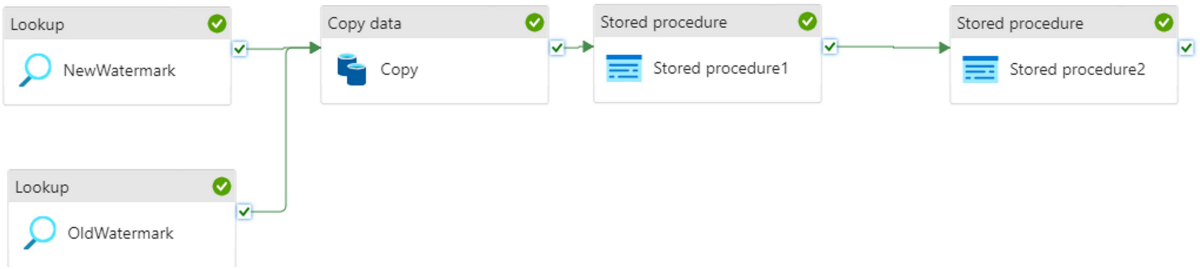
5. Get Email Notification in ADF for pipeline success and failure
   > Use logic app

6. Incrementally copy data from 1 SQL table to another SQL

Source	Destination	Tool	How ?	
Blob	Blob	ADF	Provide File path	
Blob	SQL	ADF		
Blob	NoSQL	ADF		

SQL	Blob	ADF	EDW to DL	
SQL	SQL	ADF	Incremental load - using staging	
SQL	NoSQL	ADF	EDW to cosmos lookup	
NoSQL	Blob	ADF		
NoSQL	SQL	ADF		
NoSQL	NoSQL	ADF		

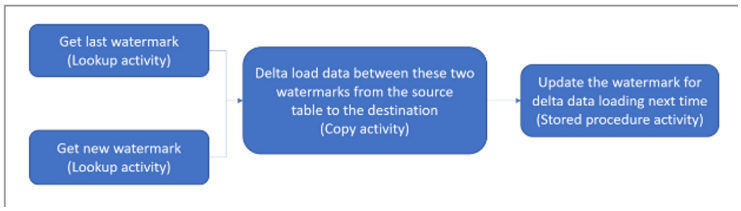
1. SQL to SQL
2. Blob to SQL



7. Blob to SQL incremental - Daily load

---

8. SQL to blob incremental load daily



1. Create linked service
2. Create dataset

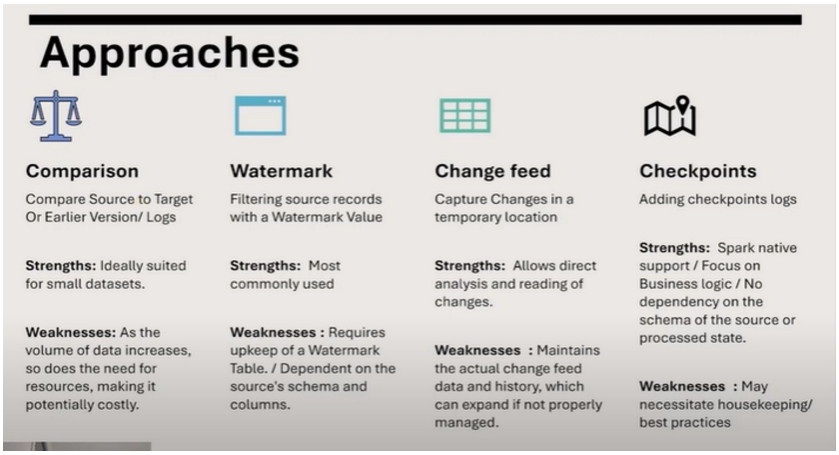
See quotes - two single quote for one single quote

@concat('Select * from dbo.totalsale where region=',"'Asia'")
@concat('Select * from dbo.totalsale where region='',pipeline().parameters.Region,'''')
@concat('Select * from dbo.totalsale where region='',pipeline().parameters.Region,'''','and Country='',pipeline().parameters.Country,'''')

select watermarkValue as OldwatermarkValue from [dbo].[watermarktable] where TableName = '@{pipeline().parameters.TableName}'

Use '@{}' for putting value in query

select * from @{pipeline().parameters.TableName} where
LastModifytime >
'@{activity('OldWaterMark').output.firstRow.OldWaterMarkValue}'
and LastModifytime <=
'@{activity('NewWaterMark').output.firstRow.NewWaterMarkValue}'



logs_@{addminutes(utcnow(),330)}

@{formatDateTime(addminutes(utcnow(),330),'yyyy/MM/dd/HH/')}

```
from pyspark.sql.types import *

spark.conf.set(
   "fs.azure.account.key.storagesaurabh2023.dfs.core.windows.net",
   "y+wV6xrnYnkTRUl/LBMWPptES38/BrxK6TliDmzyW2FJzdoTPvTG35d7aZXup7ZGGjuoJtW7xvGA+ASt1Af1SA=="
)

dbutils.fs.ls("abfss://incremental-load@storagesaurabh2023.dfs.core.windows.net/2024/")

schema = StructType([StructField("PersonID", IntegerType(), True),
   StructField("Name",  StringType(), True)
   ])

df = spark.read.format("parquet").option("header", "true").schema(schema).option("recursiveFileLookup","true").load("abfss://incremental-load@storagesaurabh2023.dfs.core.windows.net/2024/")
display(df)
```

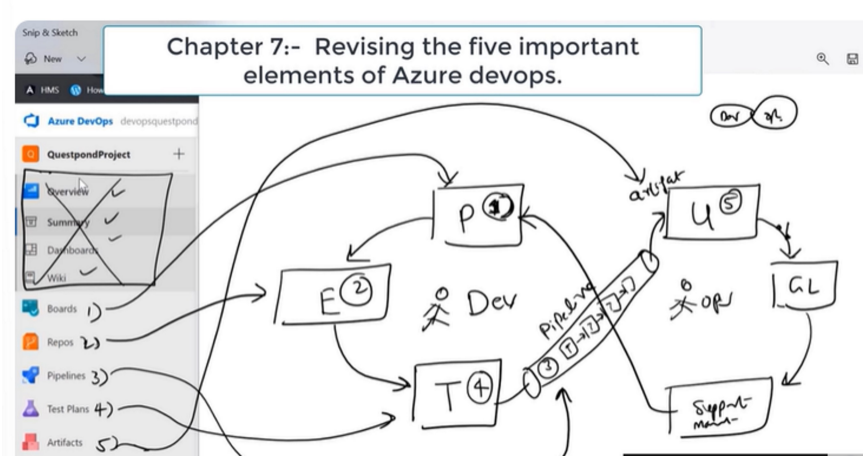
SQL to blob incremental - new files

SQL to SQL incremental - by merge

Azure devops

07 October 2023 05:51 PM

1. DevOps is a philosophy which tells - Dev and operations should work together.
2. Organization - project -



Pipeline - Test , build , approval, Publish

Pipeline execution requires machine - agent
Agent - machine which execute tasks
2 Agents - Azure pipelines & default -
create our own agent

YAML - data serialization format for storing config data -

Docker

12 February 2023 11:38 AM

1. Containerization -
   Isolated ecosystem.
   ContainerD - base docker company

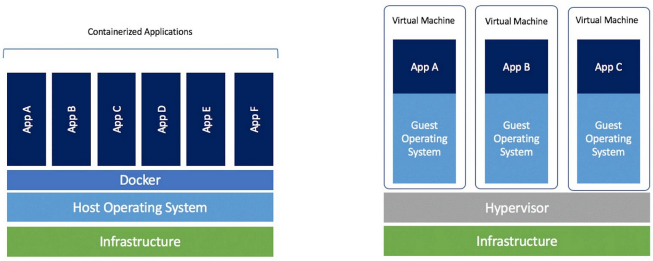
   Orchastration -
   Container - small microprocessors

   History :-

   Problems -
   Initially 1 applications on 1 server : -
   Vmware - introduced Virtual machine - multiple app on 1 VM. But it requires specific OS.
   Applications requires separate packages, dependencies - not work on other machine

   Isolation - similar to dual boot - requires dedicated hardwares
   Containers -
   Hypervisor - manages the virtual machine
   Container engine - manages the containers - docker is a tool which provides container engine

   If you want to run container on windows - > Docker desktop -
   Linux - docker engine



   Containers -
   Everything requires for your app, you put it in the container and share with other.
   Hypervisor - manages the virtual machines and is used to create multiple machines on the host operating system.
   Containers -

   Docker Daemon - It is the heart of the Docker architecture, which does the crucial work of building, running, and distributing the containers. It also manages the Docker images and the containers.

   Docker image - The file that contains the source code, operating system files to run the application along with its other dependencies inside the container is called Docker image. A containerized application is the running instance of an image. Docker images are immutable.

   Dockerfile - set of instructions to create dockerimage

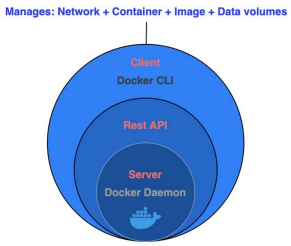
   Docker Registries -
   This is where Docker images are stored. Docker hub is the official online public repository of Docker where you can find all the images of popular applications. Docker hub also allows us to create our own images for our application.

   Docker run - run the image to create the container
   Docker images - shows the images
   docker container ls - shows the running containers
   docker inspect [image_name] -shows info about the image

   Three parts of docker -

   Docker Runtime - Allows us to start and stop the containers -
      Runc - low level runtime - work with OS and start and stop the containers
      Containerd - manages runc , interacting with network ,

Docker Engine - Interact with docker - > Docker Daemon



Docker orchestrations - Allows us to manage the containers

Docker image - instruction to make app
Dockerfile - docker image - docker container

-it -> interactive environment - go inside the container

Docker image - contains OS files and all dependencies for the app

Docker pull image - pull the image , not run it

Docker exec
Docker stop id - stop the container

Docker run -d -p 8080:80 nginx
To run the container app on your local machine , specify the - p port number

Images are built in layers - collection of layers -> every layer has # value
When the images have some layers in common, it won't download that image

How to create dockerfile -

FROM "base image"
CMD ["echo","HelloWorld"]
Exit - exit out of the container

Azure Event Hub

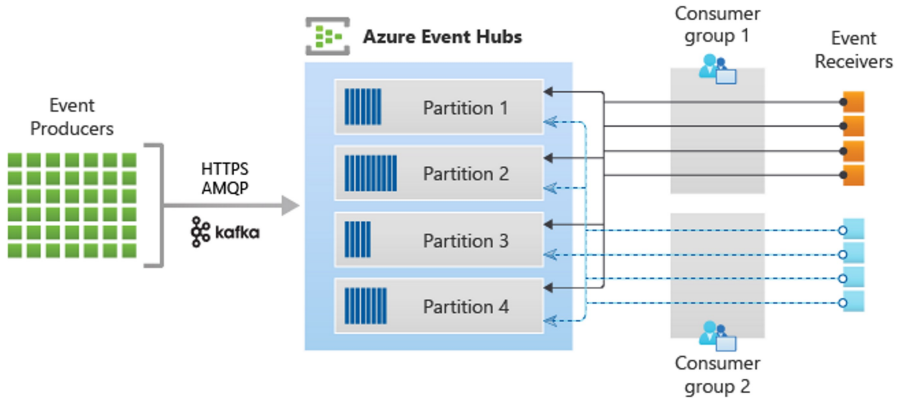
23 February 2023 12:54 PM

Partition - 1 to 32 partitions
Messages will be sent to different partitions.

Namespace -
Logical container for collection of eventhub

It's a PAAS service
Support for Real time and batch , event capture,

1. Namespace - management container for all eventHub.
2. Event producer - entity that sends data to event hub.
3. Partition - Each consumer only reads a specific subset, or partition, of the message stream.
4. Consumer groups - A view (state, position, or offset) of an entire event hub. Consumer groups enable consuming applications to each have a separate view of the event stream. They read the stream independently at their own pace and with their own offsets.
5. Throughput unit - Pre-purchased units of capacity that control the throughput capacity of Event Hubs.
6. Event receivers - Any entity that reads event data from an event hub.



Kafka Concept	Event Hubs Concept
Cluster	Namespace
Topic	An event hub
Partition	Partition
Consumer Group	Consumer Group
Offset	Offset

6. Azure Event Hubs for Apache Kafka supports both **idempotent producers and idempotent consumers.**
7. **at-least once delivery.** This approach ensures that events will always be delivered. It also means that events can be received more than once, even repeatedly.

AVRO file format

Csv format - > data type are not same, columns not available
Relational db - > data defination vary for each database
JSON - No data schema inforced - Repeated many type
Avro - its defined as a schema , It has schema and payload -

Schema registry -

Event publish max limit - 1 mb . Event Hubs ensures that all events sharing a partition key value are stored together and delivered in order of arrival.
Event retention -1 hr to 90 days - event can be stored in ADLS Gen 2 using event capture.

SAS tokens - Shared Access Signatures ,

Consumer groups - The publish/subscribe mechanism of Event Hubs is enabled through consumer groups. A consumer group is a view (state, position, or offset) of an entire event hub.

EventHubConsumerClient - supports reading from single partition also
EventProcessorClient - used for reading from all partitions -> recommended for production
PartitionReceiver - read from single event hub
PluggableCheckpointStoreEventProcessor<TPartition>
EventProcessor<TPartition>

Producer - event hub producer has 2 clients -
EventHubProducerClient - send batches - > send all data to one batch.
EventHubBufferedProducerClient

Guidance for buffered producer handler implementation

One important consideration for buffered publishing is to understand the impact of the code in the SendEventBatchSucceededAsync and SendEventBatchFailedAsync handlers. Because the producer will await the execution of these handlers after a batch is published, the time that it takes the handler to complete its work will influence how quickly another batch can be prepared and published. For maximum throughput, it is advised that handlers be kept as lightweight as possible and avoid long-running operations.

It is also important that you guard against exceptions in your handler code; it is strongly recommended to wrap your entire handler in a try/catch block and ensure that you do not re-throw exceptions. Any exceptions thrown from your handler will be caught by the buffered producer, logged, and then ignored. This ensures that the producer is resilient to handler errors but makes it difficult for applications to detect unhandled exceptions in their handler code.

EventHubConsumerClient - EventHubConsumerClient is not associated with any specific partition and the same instance can be used for reading from multiple partitions.By default, reading starts from the beginning of partitions.

It is possible to request reading from the end of each partition instead by setting startReadingAtEarliestEvent to false

The ReadEventsFromPartitionAsync method of the EventHubConsumerClient allows events to be read from a

specific partition and is suitable for production scenarios. -

SQL Queries

22 September 2023 09:33 AM

Incremental Load

```
CREATE PROCEDURE temp.usp_write_watermark @LastModifiedtime datetime, @TableName varchar(50)
AS

BEGIN

UPDATE temp.watermarktable_sa
SET [WatermarkValue] = @LastModifiedtime
WHERE [TableName] = @TableName

END


create procedure temp.stg_to_main_sa
as
BEGIN
MERGE temp.data_sink_table_sa t
USING temp.stg_data_sink_table_sa s
on t.PersonID = s.PersonID
WHEN MATCHED THEN
      UPDATE SET
         t.Name = s.Name,
         t.LastModifytime = s.LastModifytime
WHEN NOT MATCHED by TARGET THEN
      INSERT (PersonID, Name, LastModifytime)
      VALUES (s.PersonID, s.Name, s.LastModifytime);

END


update  temp.data_source_table_sa set Name='AAAA' where PersonId='1'

select * from temp.data_source_table_sa(nolock)

select * from temp.watermarktable_sa(nolock)

select * from temp.data_sink_table_sa(nolock)

select * from temp.stg_data_sink_table_sa (nolock)
insert into temp.stg_data_sink_table_sa select * from temp.data_source_table_sa where PersonID='2'

truncate table temp.watermarktable_sa

select max(LastModifytime) from temp.data_source_table_sa(nolock)
```

```
select top 1 WatermarkValue from temp.watermarktable_sa(nolock) where TableName='data_source_table'

select * from temp.data_source_table_sa(nolock) where  LastModifytime>

INSERT INTO temp.watermarktable_sa
VALUES ('data_source_table','1/1/2010 12:00:00 AM')

exec temp.usp_write_watermark '1/1/2010 12:00:00 AM', 'data_source_table'
```
