Pre-requisites

09 September 2022 08:34

Lab Setup Requirement	Hardware CPU - Intel Core i3/i5/i7 processor, RAM - at least 8 GB HDD-512 GB / 1 TB, OS - Windows 10 /8.1/11, MS Word/excel, PowerBI desktop (optional)

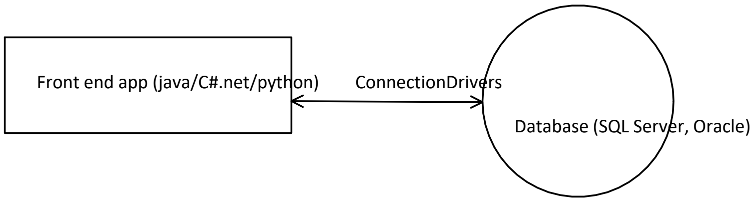
Pre-requisites
Software - SQL Server 2016, 2017 or 2019 Enterprise/Developer edition, Visual Studio 2019/2022/VS code, A Valid Azure Subscription, Azure CLI, Storage explorer, SQL Server Management Studio/Azure Data Studio, Microsoft Azure Subscription, Git tools and GitHub account, Azure PowerShell, PowerShell ISE, Note: Linux environment VMs (Apache hadoop/big data) (Linux OS) Tool: Putty.exe Azure based Linux servers, Vmware player/ virtual box AWS Tools for VS code, GCP Tools for VS code.

Azure Subscription (trial)
-- 12 months of free service + 30 days of 200 USD credit
-- enterprise subscription

Database Fundamental & SQL Server BI 2016

08 September 2022 18:45

Application metadata


Front end app (java/C#.net/python)
ConnectionDrivers
Database (SQL Server, Oracle)

1. User related attributes (which users, port, MSSQL - 1433, encryption, protocol (TCP)
2. SQLConnection
3. ADOConnection (ADO.Net)

Data Dictionary

1. Names of all of the database tables and their schemas (Sales, Customers, Orders, Employees…)
2. Details of all the tables in the database like owners of the tables, the security constraints, when the tables were created etc.
3. Physical information of the tables in the database - where the tables in the db itself have been stored and how
4. Table constraints includes primary key information, foreign key information etc.
5. Information related to database views which are visible

Employee Table - Active Data Dictionary -- self updating
Passive Data Dictionary -- manually updated to match the database

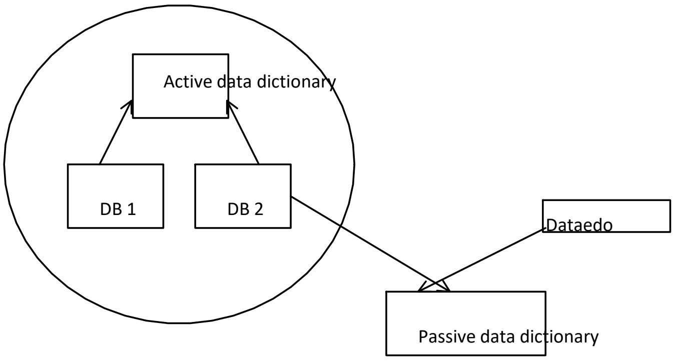
Field Name	Data Type	Field size for display	Description	Example	Dept_id
Employee No	INT	10	Unique ID for the employee	444007	1
Employee Name	VARCHAR(50)	20	Name of the Employee	James Hobb	23

Example of Passive Data Dictionary -

Dataedo -- tools

Column	Data Type	Description
Field Name	10	
Data Type	20	

Dataedo



Different Types of Database

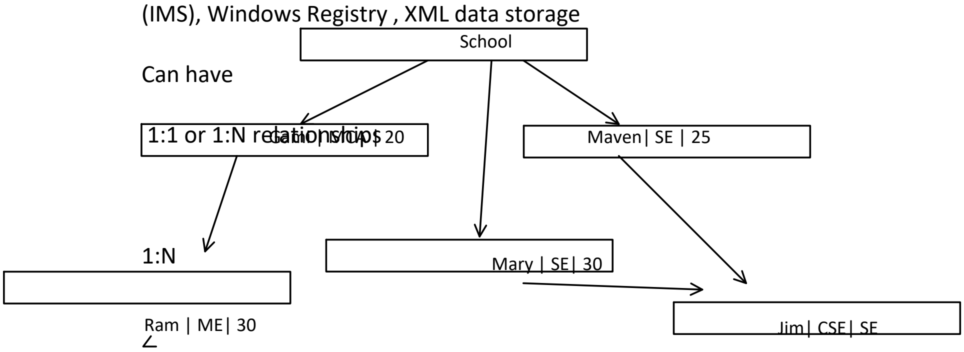
1. Relational Database - consists of set of tables with columns and rows
2. Object-oriented database - information can be presented in the form of objects as in object-oriented programming. Inclined towards into objects e.g. multimedia record in a relational database can be defined as definable data object, MongoDB has offering of Object oriented database
3. Distributed database - consists of two or more files located in different sites, e.g. SQL server mirror databases, distributed dbs

4. Data Warehouses - central repository for data storage, includes a type of database designed for faster query and analysis(SSAS, SSIS)
5. NoSQL databases - non-relational db - support for unstructured, semi-structured data, dynamic schema, flexible and faster data retrieval (Cassandra, Mongo DB, Couch DB, Azure Cosmos DB, AWS Document DB, AWS Dynamo db)
6. Graph Databases - nodes - entity , attribute - relationship

e.g. Apache Tinkerpop , Azure Cosmos db Graph API , Neo4j

7. Cloud databases - Databases as a service (DBaaS) e.g. Azure SQL database, Azure SQL managed instance
8. Document / JSON database - designed storing, retriving and managing document oriented information. (Azure Cosmos db SQL API, document db, AWS document db, data being stored in key-value pairs,

Hierarchical data model - COBOL (DB2) - IBM Information Management System (IMS), Windows Registry , XML data storage



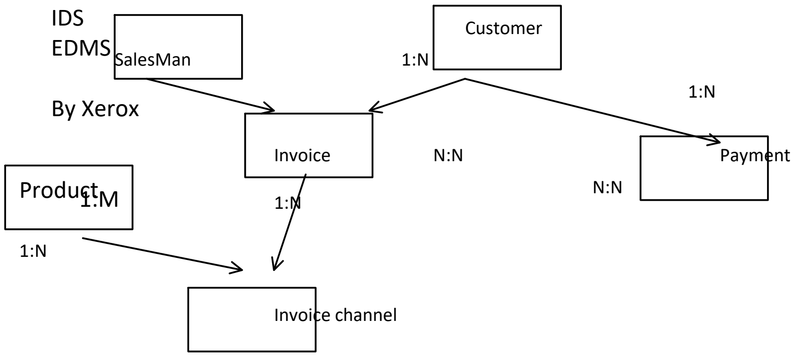
Can have

1:1 or 1:N relationships

1:N

Network data model

1. An owner record which is the same as of the parent in the hiearchical model
2. A member record which is same of child in the hiearchical mode



IDS
EDMS

By Xerox

Employee   Table 1

Fields	Columns(attribute 1) Emp ID	Columns(attribute 2)	Attribute 3
Row 1 (records/tuples)	1001		
Row 2 (records/tuples)	1002		

Employee ID - foreign

Hiearchy Data models

Mainframes DBMS for IBM

IBM IMS and RDM Mobile - embedded db

Employee table

Emp No	First Name	Last Name	Dept
1001	Alan	Turing	Finance

Device table

Serial no	Type	User emp no
001	Monitor	1001

Table 2
EmployeeAddress

Fields	Columns(attribute	Columns(attrib	Attribute

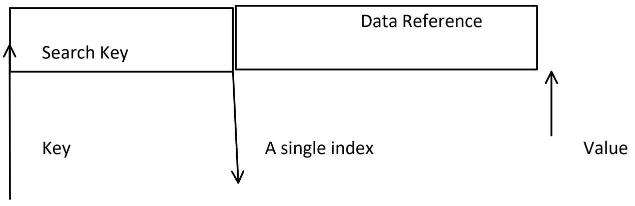
	1) Emp ID	ute 2)	3
Row 1 (records/tuples)	1001		
Row 2 (records/tuples)	1002		

Entity Integrity - ensures the primary key in a table is unique and the value is not set to null

Referential Integrity - requires every value in a specific foreign key column should be found in the primary key of the table from which it is originated.

Index

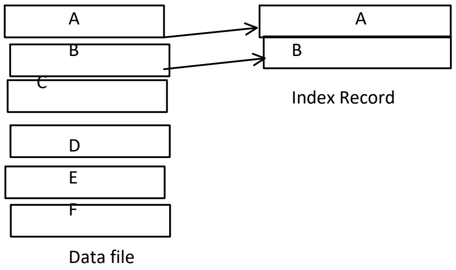
- First column for a table is the Search key which can contain a copy of the primary key of the table. These values are stored in sorted order so that the corresponding data access can be faster.
- The second column is the data reference or Pointer which can contain a set of pointers holding the addresses of the underlying disk blocks where the specific key values are stored.



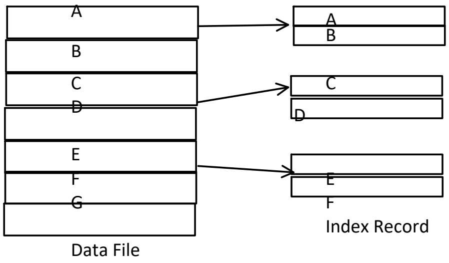
1. Access Types - value based search, range of access over data records
2. Access Time - time required to find the data element
3. Insertion time - time taken to find the specific space and to insert the new data
4. Deletion time - time taken to find an element & to delete it
5. Space overhead - additional space required by an index

File storage mechanism to follow for indexing

1. Sequential file organization -



Dense Index



Sparse Index

Foreign Key

1. The foreign key constraint is used tp prevent actions what would destroy links between tables.
2. A foreign key is a field (collection of fields) on a table refers to primary key in another table.
3. A table with the foreign key is called as child table, and the table with the primary key is called parent/referenced table.

s			

2. Hash file organization - indices are choosen based on values being distributed uniformly. Hash buckets where the value is assigned determined by hash function.

a) Clustered index - you can define an index with upto 16 columns , The max size of this index should be 900 bytes

The columns defining for clustered index is termed as clustering key.

SQL server to order the data in the table in accordance to the clustering key.

a) Non clustered index

- Do not impose a sort order on the table
- Restriction wise max size supported as 900 bytes & can be promoted max 16 columns of a table, max 249 non-clustered indexes can be created on a table.

WHERE clause used to specify the condition while fetching the data from single table or joining from multiple tables. When, a given condition is satisfied, use the WHERE clause to filter the specific records and fetching the necessary records.

Surrogate Key

School A

Reg No	Name	% obtained
201010	Brian	66

Reg No	Name	% obtained
201010	Brian	66
202012	Max	50

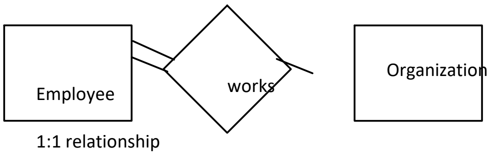
School B

Reg No	Name	% Obtained
CS300	Ava	50
DS500	Maria	60

Merging these two tables in a single sql table

1. Automatically generated by the system
2. It hold anonymous integer
3. It contains unique value for all records in the table
4. Value cannot get modified
5. Easier identification purposes

Surr_id	Registration no	Name	% obtained
1	201010	Brian	66
2	202012	Max	50
3	CS300	Ava	50



1:1 relationship
Students - enrolled to - Courses

M:N M:1

Unary relationship

from single table or joining from multiple tables. When, a given condition is satisfied, use the WHERE clause to filter the specific records and fetching the necessary records.

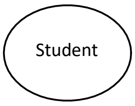
ALTER ID, NAME, SALARY FROM CUSTOMER WHERE NAME = 'XYZ'
DELETE ID, NAME, SALARY from CUSTOMER WHERE NAME = 'ABC'

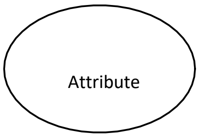
a) Multilevel index

ER Modelling

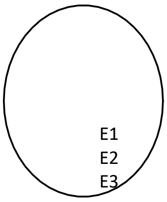
Entity - A Entity is an object with a physical existence - a particular person, car, house,

A entity is an object of entity type & set of all entities is called as entity set.

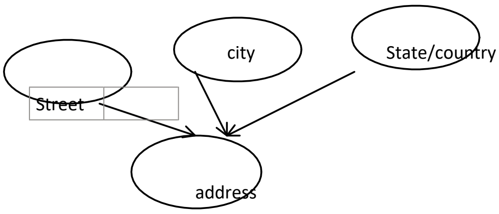




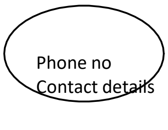
Student_roll_no
Student_name
Student_DOB
Student_Address



Key Attribute - student_roll_no



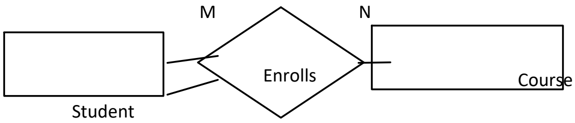
Multivalued attribute



Derived attributes
Student_age,
Employee_bonus

Participation constraint

1. Total Participation - each entity must be participated in the relationship.
2. Partial Participation - entity in an entity relationship may or may not participate in the relationship.



Employees --- Taking leaves -- During vacation (M:N)

PersonId (Primary Key)	LastName	FirstName	Age
1	Bill	Johns	30
2`	Maria	Sophia	21

SQL Command

1. DDL - CREATE, DROP, ALTER, TRUNCATE
2. DML - INSERT, UPDATE, DELETE
3. DCL - GRANT , REVOKE
4. TCL - Commit, Rollback, Savepoint
5. DQL - SELECT

Orders table

OrderID	OrderNumber	PersonID (Foreign Key)
1	77445	2
2	88445	1

The foreign key constraint prevents invalid data from being inserted into the foreign key column, since it has to be one of the contained value in the parent table.

Normalization

09 September 2022 18:33

- Normalization is the process of organizing the data into the database
- Normalization is used to minimize the redundancy from a relation or set of relations. It's also defined to eliminate the undesirable features like insertion, update and deletion anomalies.
- Normalization divides the large unnormalized tables into smaller and links them using relationships.
- The normal form is used to reduce the redundancy from the database level.

Purpose of normalization -

Data modification anomalies can be differentiated into the types:

1. Insertion Anomaly - a new row/tuple cant be inserted into a relationship due to lack of data
2. Deletion Anomaly - The delete anomaly refers to the scenario, where the deletion of data from the row results in loss of some other important data due to lack of proper key based attribute level relationships.
3. Updatation Anomaly - The update anomaly can exist when an update operation of a single data value requires multiple rows/tuples of data to be amended/updated.

	1NF	2NF	3NF	BCNF	4NF	5NF
Conditions	Elimination of repeating of groups	Eliminate the partial functional dependency.	Reduce the transitive dependency	More advanced level than 3NF. More stricter enforced for 3NF.	Eliminates the concepts of multi-values dependency.	Eliminates the concepts of joining dependency
Feature	A relation in first normal form when it consists of only an atomic value. One attribute contains only one value for a specific row.	Tables should be in 1NF + non-key attributes which are fully functional type they must dependent on the primary key	The relation fulfills the criteria for 2NF + no transitive relationship exists.	A table in BCNF, if there is a functional dependency exists x-> y (x is assumed to be the super key). Table should be in 3NF. For every functional dependency, the left side of relationship of the table fulfills the criteria for super key.	The table should in BCNF, it should have no multi-value dependency.	The table should be in 4NF + there cant be any join level dependency exists in the table, joining of the table should not incur any data loss or any joining should not have any particular loss.

Benefits of normalization -

1. Reduce the data redundancy
2. Greater data organization & consistencies.
3. Flexible level of database design
4. Enforce the concepts the referential integrity

First Normal Form (1NF)

- The table should be in the form where the one attribute should contains only one value based on specific row / tuple
- A table should have atomic values (no duplication of values on attributes) , enforce non-repetitive groups/attributes

- The column should have one single valued attribute.

Unnormalized table - Employee table

Employee_id	Employee_name	Employee_Phone	Employee_Email	Employee_HireDate	Employee_Salary
001	Mark	1988233222 111222333	mark@contoso.io Mark.b@fabrikum.com	01/01/2021	4000
002	John	222444111 444333777	john@contoso.com jo@adven.com	03/02/2017	3500

1NF is fulfilled for Employee table

Employee_Id	Employee_Name	Employee_Phone	Employee_Email	Employee_HireDate	Employee_Salary
001	Mark	1988233222	mark@contoso.io	01/01/2021	4000
001	Mark	111222333	Mark.b@fabrikum.com	01/01/2021	4000
002	John	222444111	john@contoso.com	03/02/2017	3500
002	John	444333777	jo@adven.com	03/02/2017	3500

2NF - Second Normal Form

- To be in 2NF, the tables should be in 1st Normal Form.
- All non-key attributes should be fully functional dependent on the primary key of the table

In this table, non-prime attribute Employee_Name is dependent on the Employee_ID which is proper subset of a candidate key.

Employee_detail table

Employee_ID	Employee_Name	Employee_HireDate
001	Mark	01/01/2021
002	John	01/01/2021
003	..	..

Employee_Salary table

Employee_ID	Employee_Salary	
001	4000	
002	3500	

Employee_Contacts table

Employee_ID	Employee_Phone	Employee_Email
001	1988233222	mark@contoso.io
002	111222333	john@contoso.com

Professor table (1NF)

ID	Course	Univ Name	Name
10	CSE	Stanford	
15	IT	Harvard	

15	ME	Harvard	
30	DB	Princeton	
30	CA	Princeton	

2NF

Professor_detail table

ID	Univ Name	Name
10	Stanford	Mark
15	Harvard	Mark
30	Princeton	John

Fulfills the partial dependency.

Professor_subjects table

ID	Courses
10	CSE
15	IT
15	ME
30	DB
30	CA

Third Normal Form (3NF)

- A table can be in 3NF, if it is in 2NF, should not have any partial functional dependency
- It should reduce the data duplication
- It can achive the integrity
- No transitive dependency between non-prime attributes/columns.

A -> B -> C => Column A is dependent on B , B dependent on C, if A is also dependent on C, then it's called as transitive dependency.

Employee_details (2NF)

Emp_id	Emp_Name	Emp_Zip	Emp_State	Emp_City
222	Mark	70045	Arizona	Phoenix
333	Harry	34404	Utah	Lake
555	Jerry	40032	Arizona	Maveric
335	Hannah	33406	New Mexico	Titan

Super Key relation -> (Emp_id), (Emp_id) (Emp_Name), (Emp_id)(Emp_Name)(Emp_Zip)…

Candidate key -> Emp_id

Delhi - 11….
Mumbai - 4….

Emp_State and Emp_City is dependent on the Emp_Zip & Emp_Zip is dependent on the Emp_id. The non -primary key attribute (Emp_State) and (Emp_City) transitively dependent on primary key (Emp_ID). It violates the criteria of 3NF.

Employee tbl

Emp_ID	Emp_Name	Emp_Zip

222	Mark	70045
333	Harry	34404
555	Jerry	40032
335	Hannah	33406

Employee_zipcode table

Emp_Zip	Emp_State	Emp_City
70045	Arizona	Phoenix
34404	Utah	Lake
40032	Arizona	Maveric
33406	New Mexico	Titan

BCNF - (Boyce Codd Normal Form)

- It is stricter than 3NF, more advanced than 3NF.
- A table will be in BCNF if every dependency exists like with a super key to no the attribute, x -> y. (x is the super key of the table).

Employee_details table (3NF)

Emp_id	Emp_country	Emp_dept	Depart_Name	Emp_depart_no
333	US	Manufacturing	Design	001
444	UK	Software	Engineering	002
555	US	Architecture	Development	003
666	US	Machinery	Development	004

Functional Dependency ->

Emp_id -> Emp_Country
Emp_Dept -> Department_name, Emp_depart_no

Candidate key -> (emp_id, emp_dept)

This table is not in BCNF, because neither the Emp_Dept and Emp_id alone the keys.

Emp_country table

Emp_id	Emp_country
333	US
444	UK
555	US
666	US

Candidate key - Emp_id

Emp_department table

Emp_dept	Emp_Department_name	Emp_dept_no
Manufacturing	Design	001
Software	Engineering	002
Architecture	Development	003
Machinery	Development	004

Candidate Key - Emp_Dept

Third table should have both of these candidate keys

(Emp_id, Emp_dept)

Emp_dept_mapping table

Emp_Id	Emp_Dept
333	Manufacturing
444	Software
555	Architecture
666	Machinery

The left side of both the functional dependencies is a key. --> This table is in BCNF.

Forth Normal Form (4NF)

- A table is in fourth normal form (4NF), if it's in already in BCNF & has no multivalue dependency.

A -> B, if for single value of A, multiple values of B exists, then it's called as multivalue dependency.

Student table (3NF)

ID	Course_enrolled	Programming_skills
10	Computer Science	C
10	Engineering Math	java
30	IT	Networking
40	CS	Compiler Design
55	Bioinformatics	C

There's multivalue dependency exists for the student with ID=10

Course_enrollment table

ID	Enrolled_courses
10	Computer Science

10	Engineering Math
30	IT
40	CS
55	Bioinformatics

skills

ID	Programming
10	C
10	java
30	Networking
40	Compiler design
55	C

Fifth Normal Form - 5NF

- A table is in 5NF, if it's already in 4NF & should not contain the join dependency. The joining should not incur any data loss.
- 5NF is satisfied when all the table are being broken into as many as tables as possible to avoid redundancy.
- 5NF is called project level join normal form.

  Employee table

Employee_Name	Employee_HireDate	Employee_Designation
John	01/01/2020	Sr. Software Engineer
Mark	01/03/2021	Software Arch
Mark	01/03/2021	Software Engineer
Celine	04/02/2014	Sr. Software Engineer
Alan	01/03/2021	Network Engineer

P1 level

Employee_Desig	Emp_Name
Sr. Software Engineer	John
Software Arch	Mark
Software Engineer	Mark
Sr. Software Engineer	Celine
Network Engineer	Alan

P2 level

Emp_Name	Emp_HireDate
John	01/01/2020
Mark	01/03/2021
Mark	01/03/2021
Celine	04/02/2014

Alan	01/03/2021

P3 Level

Emp_Design	Emp_HireDate
Sr. Software Engg	01/01/2020
Software Arch	01/03/2021
Software Engineer	01/03/2021
Sr. Software Engineer	04/02/2014
Network Engineer	01/03/2021

TSQL concepts

08 September 2022 18:45

SQL Table Design

1. Data types : A data type is fundamental constraining element of a database which restricts the range of possible values that are allowed to be stored in a column

a) Numeric data type :
- tinyint (0-255)
- Smallint(-32768 to 32767)
- Int (storage space - 4 bytes)
- Bigint(8 bytes)
- Decimal(p,s) fixed precision and s - scale numbers , precision - max total no of decimal digits can be stored , scale - number of decimal digits which are stored to the right of precision point.
- Numeric(p,s) - a constant data value can be automatically converted to a numeric data value. SQL server uses the default rounding options when converting a number to decimal or numeric one with smaller precision and scale.
- Smallmoney (-214748.00… 214748.00) - 4 bytes
- Money - 8 bytes accuracy of 10000 of monetary units.
- Real -3.4 , -1.18 to positive values (4 bytes)
- Float - 4 bytes or 8 bytes

b) Character data type
- Char(n) - 1 byte / character upto 8k bytes
- Varchar(n) - max 8k bytes
- Text - stores upto 2 gb
- Nchar(n) - 2 bytes per character max 4k bytes
- Nvarchar(n) - 2 bytes per character max of 4k bytes
- Ntext - 2 bytes per character stored upto 2 gb
  The (n) characters defined sets the max no of characters allowed to be stored in the column
  Nvarchar and varchar - the amount of storage consumed is equal to the number of characters being stored.
  Varchar(max), nvarchar(max) - 2 gb of data

c) Binary data type
Binary data type can be fixed length or variable length

- Binary (sizes of column data entries are consistent) upto 8k bytes
- Varbinary - variable length binary data
- Varbinary(max) - storage exceeds beyond 8k bytes
- Image - variable length binary data upto 2 gb
- Alternative varbinary(max)

d) Spatial data type:
- Geography - implemented as .net CLR (latitude, longitude)
- Geometry - store points, lines, curves

e) FileStream data type:
BLOB data stored , not restricted to 2 gb of limit of file system

f) HiearchyID data type:
Storing of nodes & edges/vertices of graphs, flowcharts

SQL Server Column properties

g) Sparse Columns
The attribute of a specific row if requires very small values & need small storage space, then can use the Sparse property.

-- Temporal tables

It is a database feature which brings built-in support for providing the information about the data stored in the table in time, rather than only the data which is correct at the current moment of time.

System-versioned temporal table is kind of user table designed to keep a full history of data changes, allowing for easy point-in time analysis.

Temporal tables have two explicit defined columns with datetime2 data type, these columns are called as period columns.

Benefits

- Auditing all data changes and performing data forensics
- Calculate the data column change trends over the time
- Maintain a slowly changing dimension for the decision support apps
- Recover from accidental data damages and errors

```
string connectionString = "Data Source:MSSQL1;"+"Initial Catalog=sampleDB;Integrated Security=SSPI;" + "MultipleActiveResultSets=True"
```

Session cache -> logical session
SqlClient driver (c#) caches the MARS session within a connection. 10 MARS session.

SQL Server Clauses:

1. SQL Order By Clause:

- Order the result set of a specific query by the specified column list and optionally it also limits the rows returned to a specified range. The order in which the rows are returned in a result set are not been guaranteed unless an ORDER BY clause is defined
- Determine the order om which Ranking function values are applied to the result set. --> Ranking function helps to return a ranking value for each row in a partition.

ORDER BY clause is not supported for CREATE TABLE AS SELECT(CTAS) statements in Azure Synapse & AAS.

ORDER BY expressions
 [collate collation_name]
 [ASC | DESC]

2. HAVING Clause -

- Specifies the search condition for a group or an aggregate. HAVING clause can be used only with the SELECT statement. HAVING is typically used with the GROUP BY clause. When the GROUP BY clause is not used, there's an implicit single, aggregated group.

[HAVING <search_criteria>] - one or more predicates for groups/aggregates to meet.

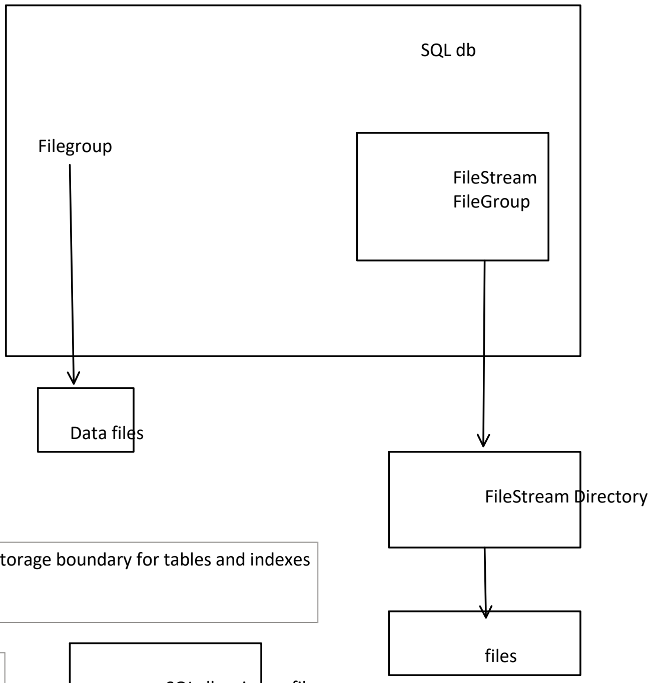
The text, image and ntext data cant work with HAVING Clause.

FileStream in SQL Server

Database structure

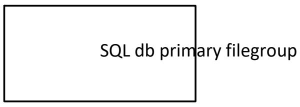
- .mdf file - primary database
- .ndf file - secondary db
- .ldf file - transaction log file
-
- Data and log file for SQL server

a) Physical file name Fi
b) Initial file size
c) File growth factor
d) Maximum size
e)
f)



data filestream full-text	Filegroup is responsible to create storage boundary for tables and indexes

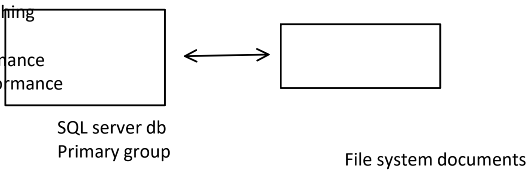
Emp id	Emp_name	Emp_phone (varbinary(max))
1	Alan	0*E98886AX998CC
2	Matt	0*F588886AX998CC



Emp id	Emp_name	Emp_phone (varbinary(max)) FILESTREAM
1	Alan	0*E98886AX998CC
2	Matt	0*F588886AX998CC

Documents are stored in the file system and the db has a particular FILESTREAM

- Does not have to use high memory and memory buffer pool for caching Large objects.
- FILESTREAM enables caching at the system cache providing performance Benefits for large media files without affecting core sql server performance



Temp tables advantages

- Store data temporarily, large datasets needed to perform data transformation and modification

-- in-memory based optimized tables, schemas and data required to store until the db restarts.
-- required to store these tables in memory pools
-- retriction in terms of memory usage but storage for disk

Index Operators

OFFSET FETCH

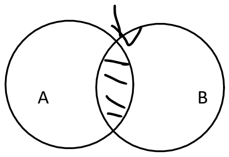
ID	Name
1	…
2	…
3	…
4	…
5	…
6	…

OFFSET clause - specifies the number of rows to skip before starting to return rows from the query.

FETCH clause - defines the number of rows to return after the OFFSET clause has been processed.

Skip first two rows and fetch next 4 rows only, we can use OFFSET and FETCH clauses with ORDER BY clause.

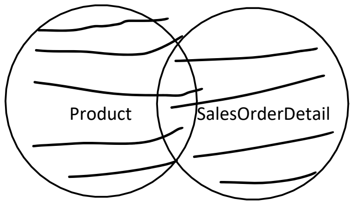
INNER JOIN -- helps to create a new table by combining rows which has matching values in two or more tables.



Outer Join - to join or match the rows between tables, want to get the matched rows along with unmatched rows from one or both tables.

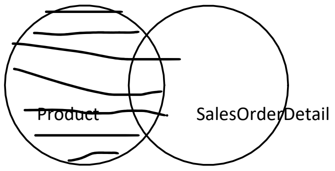
- SQL full outer join
- SQL left outer join
- Sql right outer join

Full Outer Join - In full outer join, all of the rows from both of the tables are included, if there's any unmatched rows, it will show NULL values from them.



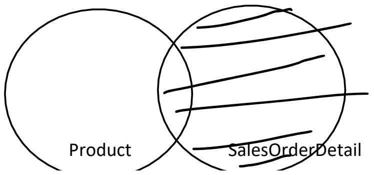
Left Outer Join - in left outer join, we can get the specific rows from the output.

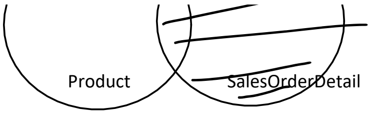
- It gives the output of the matching row/rows between both of the tables.
- If no records are found to have matching, it will show such records with null values.
- Based on the joining clause on the two tables are specified, all data is returned from the left table.
- On the right table, the matching data is returned in addition to the NULL values where a record exists in the left table, but not in the right table.



Right Outer Join - Based on two tables, specified in the JOIN clause, all data is going to return from the right table. On the left table, the matching data is returned in addition to NULL values where a record exists in the right table but not in the left table.

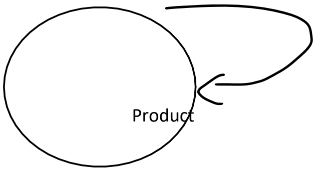
- It gives the output of matching row between two tables.
- If no records are matching from the right table, it will show these records with the NULL value.





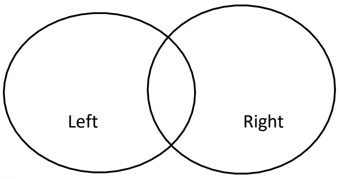
Self join - in any practical circumstances, the same table is specified twice with two different aliases in order to match the data within the same table.

Self join when it's a requirement to create a result set joining the records in the table with some other records in the same table.



Cross join - Based on two tables specified in the Join clause, a Cartesian product is created, if a WHERE clause does the filtering for the rows. The size for the cartesian product is based on the multiplication of the number of rows from the left table by the number of rows in the right table.

- Cross join returns all rows for all of the possible combinations for two tables.
- It generates all the rows from the left table which is then combined with all of the rows from the right table.
- This kind of joining is called Cartesian product (A*B)



Left = 100 rows

Right = 100 rows

100 * 100 = 10,000 rows

Employee table

Emp_Name	EmpSalary	Rank_id
Alan	500	1
Alan	800	1
alan	400	1
Rachel	600	4
rachel	400	4
Tony	300	6

Row No
1
2
3
4
5
6
7
8
9
10

Pivot table:

Student table

Student	Subject	Marks
Jacob	Maths	100
Jerry	CS	70
Mark	Science	80

Student	Maths	CS	Science
Jacob	100		
Jerry		70	
Mark			80

VendorID	Year	
	2001	
	2002	

VendorID	2001	2002	2003

	2003	
	2004	

SQL Server View

A View in SQL Server is simply a SELECT statement which has been given a name and stored in a database.

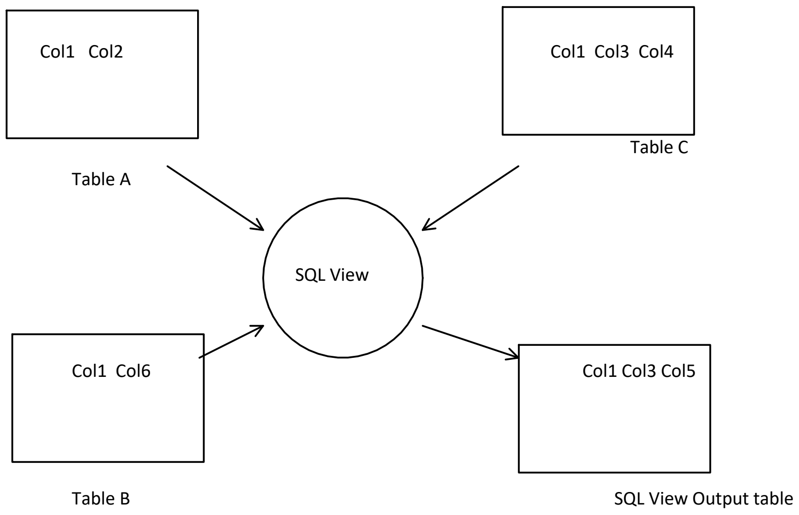
Data is stored in RDBMS in the form of tables, stored procedures, views etc.

Drawbacks:

- Normalization is a database process which is used for organizing the data in the database by splitting the large tables into smaller tables.
- These multiple tables in SQL server are linked using the relationships.
- Developers who are writing queries to retrieve those data from the multiple tables and columns, they need to perform multiple joining and complex queries.

To overcome all of these challenges, SQL server has the concept of Views.

- A view in SQL server is a virtual table which contains the data from one or multiple tables.
- Similar to like SQL table, the view name should be unique in the database.
- View contains a Set of predefined SQL queries to fetch the data from the database itself.
- So, a view contains database tables from the single or multiple databases as well.
-



- SQL Server View can retrieve the data from multiple tables
- It can show the View output in the output table

1. Create a SQL View

Create view view_name
As
Select column1, column2, column3…. columnN from tables
Where conditions;

Features of SQL Server View:

- Since View is a stored name for a SELECT statement, the SELECT statement which is defined for the View , can reference tables, views, and functions.

Core Features

- **The select statement contain the COMPUTE and COMPUTE BY clause**
- **USE the INTO keyword**
- **Use an Option clause**
- **Reference a temp table or variable of any type**
- **Contain an ORDER BY clause unless a TOP operator is specified.**
- **The View can contain multiple SELECT statements as long as can define the UNION and UNION ALL operators.**

SQL Server Stored Procedures

SQL Server stored procedure is a batch of statements grouped together as a logical unit and stored in the database.

The stored procedure accepts the parameters and executes the T-SQL statements in the procedure in SQL Server.

Benefits for Stored Procedure

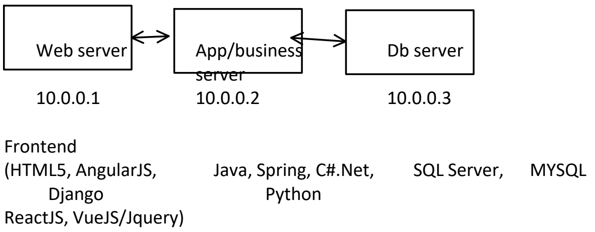
-- It can be easily modified : we can easily modify the code inside the stored procedure without the need to restart or deploying any application. For e.g. Whenever the logic needs to change, we just to execute the procedure with a simple ALTER PROCEDURE command.

-- Reduced network traffic - procedures can passed over the network instead of the Whole tsql code.

-- Reusability - stored procedures can be executed by multiple users or multiple client Apps without the need to write code again.

-- Security -- Stored procedures can reduce the threat and vulnerabilities by eliminating Direct access to the tables. Applies the encryption by encrypting the stored procedure.

-- Performance efficiency - The SQL Server stored procedure while executed for the first time, it creates A plan and stores it in the memory buffer pool so that the plan can be used in the next time when the same query / procedure is executed.



Drawbacks of Stored Procedure
-- Testing & Debugging : testing of logic encapsulated in Stored procedure is difficult.

-- Debugging -- not possible in stored procedure

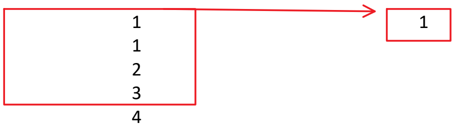
-- Version Control --- not supported in SP

-- Cost --
-- Portability - in terms of Versioning and Vendor product aspect Oracle -> SQL Server

A sql index is a quick lookup table for finding records for users as required to search without going for the entire table scanning.
SQL indexes are kind of performance tool which helps to optimize the searching of records from the tables without row by row search operations.

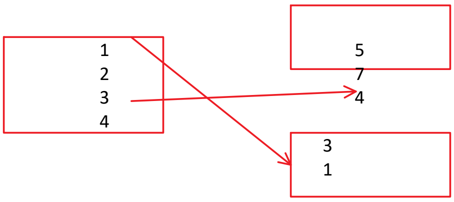
Clustered index

a) The data we can move in memory has to be in sequential or sorted order
b) There should be a key value means it cant have repeated values.
c) It will perform sorting on the tables only with clustered indexes
d) Only one clustered index in a table for general cases
e) Same like dictionary where data is arranged in alphabetical orders.
f) Index contains the pointer to the block but not direct data



Non-clustered index

a) The data is stored in one place and index is stored in another place.
b) Since , in non-clustered index, data and the pointer is stored seperately.
c) Hence, it's very common to have multiple non clustered indexes in a table



SQL Functions

11 September 2022 22:06

Functions
In-built functions for SQL Server

Special Functions in t-SQL
- Row Number Function
- Rank and Dense Rank Function
- Calculate Running Total in t-SQL
- NTILE Function
- Lead and Lag Functions
- FIRST VALUE Function
- Window Functions
- LAST VALUE Function
- PIVOT and UNPIVOT
- CHOOSE Function
- IIF Function
- EOMONTH Function
- DATEFROMPATS Function

SQL Server STRING Function

CHAR	Convert an ASCII value to character
CONCAT	JOIN two or more strings into one string
DIFFERENCE	The difference (values) of two strings
FORMAT	Return a value formatted with the specified format and optional values
LEN	Returns a number of characters of a character string
Lower	Returns a string on lowercase format
REPLACE	Replaces all occurrences of a substring within a string with another substring
REPLICATE	Returns a string repeated a specified number of times
RIGHT	Extract a given a no of characters from a string starting from Right

REPLACE(input_string, substring, new_substring) -- any string expression to be search
Substring -- is the string which has to be replaced
New_substring -- is the replace string

REPLICATE('MANGO', 20) results;

RIGHT(input_string, no_of_characters)
Input can be literal string, variable, column

Input string - can be string, variable & data type can be anything except TEXT, NTEXT / VARCHAR()
-- no_of_characters - is a positive integer which defines the actual no of characters of the input_string to be returned.

SELECT RIGHT('SQL Server', 6) RESULTS

Results
Server

SQL Sequence is available for SQL Server , Azure SQL db.

(3,5,8,9,19)….
(2,3,4,5)… (2,4,6…)

In SQL Server, A Sequence refers to a user-defined schema bound object which generates a sequence of numbers according to the specified specification. A Sequence in SQL Server contains numeric values which can be ascending or descending order at the defined interval & may cycle as if requested.

Create sequence [schema_name] .sequenceName
[AS integer_type]
[START WITH start_value]
[INCREMENT BY increment_value]
[{Minvalue[min_value]} | {No_Minvalue}]
[{Maxvalue[min_value]} | {No_MaxValue}]
[Cycle | {No_Cycle}]

-- A SQL server sequence is also should be unique in the current db

-- use any valid integer type for creating sequence e.g. tinyint, smallint, int, bigint or decimal and numeric with scale of 0

Start_value should be in the ranges of min_value and max_value

EOMONTH function - return the last day of the month of a specified date with optional offset.

EMONTH(start_date [offset]);

User-defined Functions

1. User-defined functions are routines function-body can accept parameters, they can perform actions, complex calculations, can return the results as value. These return values could be either be a single scalar set or result set

Benefits of user-defined functions

1. Customized functions with modular programming -- create the function only once, store it in the database, then use it any times. You can modify independently of the program source code.
2. Faster execution - tsql based user-defined functions (udfs) can reduce the compilation cost by caching the results in the plans and reuse them in repeated execution.
3. reduce network traffic - the function can be invoked in the where clause to reduce the number of rows sent to the client.

SQL Server UDF are of three types

- Scalar function - this function returns a single data value of the type defined for the RETURN clause.

- Table-valued function - returns the table data type.
- System function - String, System - CAST, CONVERT, ISNULL, Ranking (Row_number, Rank, Dense_Rank, Ntile) , Dynamic Management view (DMVs)

Features of Function Category

a) Deterministic function - returns the same result every time when they are called with a set of input values and same state of the db.
   Example: AVG() returns the same result for a specific input dataset

b) Non-deterministic function -- may return different values each time when they are called with a specific set of input values,
   Example:
   GETDATE() - returns a different value every time

   -- Limitations of SQL Server UDFs.

   -- SCHEMABINDING -- SQL Server UDFs can also be created with SCHEMABINDING, we wont be able to delete the function.

- Cant delete the function if there're computed columns are available in the function and indexing.

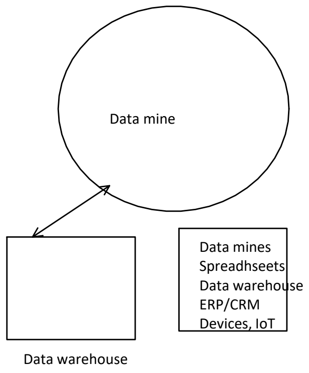
Data Warehouse Concepts

08 September 2022 18:46

1. Datawarehouse definition
2. What is Data Marts
3. What are the Data Lakes? Why should we design a Data Lake?
4. Examples of Data Warehouse..
5. Examples of Data Lake & Data Mart.
6. Data Warehouse Architecture
7. Tables design in datawarehouse
   - Star Schema design in SQL Server
   - Fact table
   - Dimension table

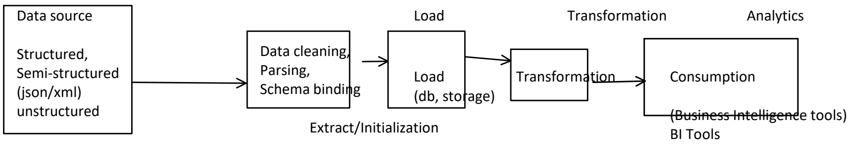
   Benefits of Data ware house
8. Defining the concepts on data modeling
   - Star Schema (demo)
   - Snowflake schema
9. Definition of data integration
10. OLAP (Online Analytical processing) , benefits, use case
11. Difference between OLAP (Data warehouse) and OLTP (sql database)
12. Introduction to SQL Server Analysis service (SSAS)
13. Data mining concepts , cube, dimension.

Tools :
SSMS (SQL Server engine)
SSAS - SSAS - SQL Server Analysis Service
Visual Studio (2008, 2012, 2015, 2017, 2019/2022)



1. Each section of data mines consists all sorts of product, product information, schemas, store information, product numbers
2. Banking/BFSI, Healthcare, Retail, Manufacturing
3. Data warehouse acts as central repository to get information from different sources and consolidates data through loading, processing and transforming.

Data Pipeline Architecture



BI Tools

1. PowerBI tool
2. SSRS (SQL Server Reporting Services)
3. Tableau
4. MSTR (Microstrategy)
5. QLIK

ERP (Enterprise Resource Planning)
SAP
CRM (Customer Relationship Management) - Dynamics 365, Salesforce CRM
IoT (internet of Things) weather forcasting, climate changing, sensor information in manufacturing

<weather>
<temp>31</temp>

Extract - Load - Transform (ELT) --
Extract - Transform - Load (ETL) -- PowerBI

Query
Customer+ Product

ETL and ELT are different. Depends on from scenario to scenario of use cases,

1. After data extraction and cleaning the data can be loaded into database and storage before transformation (ELT)
2. After data extraction and cleaning, when the data is transformed (querying, aggregation, stored procedures) then finally the data is loaded and moved into data warehouse (central repository of the company) (ETL) , BI dashboards can developed from taking data from the warehouse or analytics later.

ELT + ETL

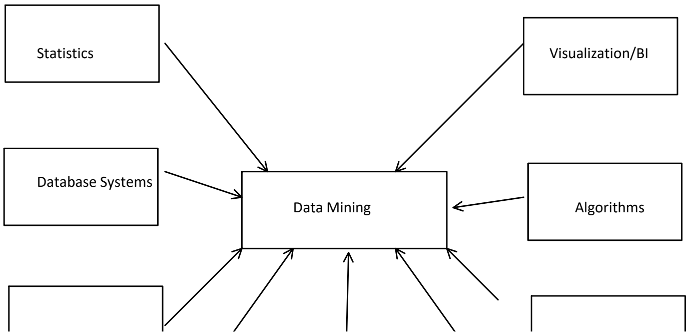
ETL = Extract, Transform and Load (data is loaded and stored in data warehouse) then moved to BI.
ELT = Extract, Load, Transform (data is transformed and stored in the database, data warehouse or data storage) then move to BI

Requirement of Data Warehouse

1. Data ware house is built to overcome the limitation of database
   Databases are in (GB , MB) size max of a TB.
2. 500 - 600 TB, 1024 TB (1 PB)
3. Reporting, Analysis purpose we need Data warehouse
4. Business decision, analytical trends information has to be stored in Data warehouse

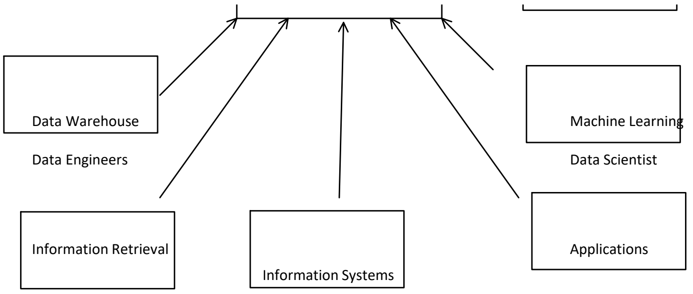
Examples of applications of Data Warehousing :

1. Social Media websites : analysing the large datasets, stored in a central repository. Data warehouse
2. BFSI - large datasets are stored in the central repo. That is Data warehouse
3. Retail - product recommendation based on large datasets stored in data ware houses. The analysis is performed on this data warehouse datasets to provide this kind of real time recommendation.



1. Decision making systems
2. Recommendation systems
3. Predict models on hidden patterns
4. Predict the future trends
5. Informed Business Decisions

1. Data Warehouse
2. Transactional database
3. Time series db
4. Social media websites

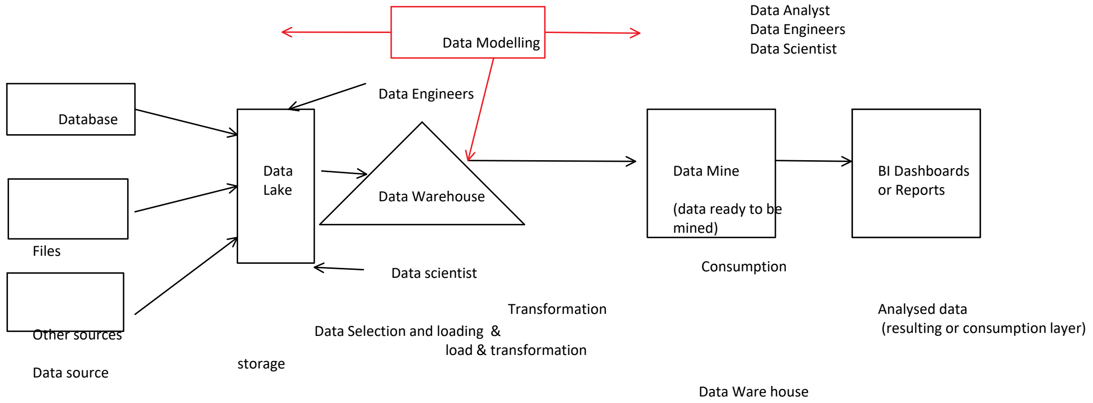


1. Data Warehouse
2. Transactional database
3. Time series db
4. Social media websites

Age	Region	Previous Purchased Product	Score	Recommendation
		Medcines	8	Cloths
		Book reader	6	Books, Digital app

50- 60 Rural
40-50 Urban

Data Analyst
Data Engineers
Data Scientist



1. Data - preprocessing - cleaning, integration, selection
2. Transformation
3. Data Mining
4. Data evaluation, visualization & presentation

1. IBM DB2
2. AWS Redshift
3. Azure SQL Data warehouse
4. Snowflake
5. Oracle Exadata

a) Volume (TB -> PB)
b) Velocity
c) Variety (structured, semi-structured, unstructured)
d) Veracity (formats, csv/tsv, .tar.gz)

Data Marts:

Data Warehouse is divided into smaller subsections as per business function, purpose and usage.

Customer -> Data Warehouse

1. Customer Inventory ---> Data Marts
2. Customer Demographics --> Data Marts
3. Customer Order -- > Data Marts.

a) Data Marts are easy to create
b) Less complex
c) Accelerate the business process

Data Lake:

Repository which stores all type of data whether structured, unstructured or semi-structured with any volume.

Principles

a) Volume (TB -> PB)
b) Velocity
c) Variety (structured, semi-structured, unstructured)
d) Veracity (formats, csv/tsv, .tar.gz)

1. Data Warehouse is for Analytics purpose
2. Data Lake is for data storage purpose

a) Batch -- SQL db
b) Real-time -- IoT devices, weather data

Examples:

1. Azure Data Lake
2. AWS Data Lake with S3
3. Informatica
4. Snowflake
5. Teradata
6. SAS

Data Modelling

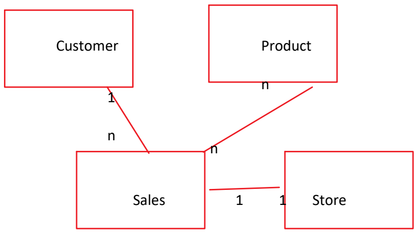
It's a process of creating a visual representation of either a while information system or it could involve collection and creating data visuals from different data sources/segments.

--- Data Models are built around the business needs.
-- Data can be modeled at various levels of abstraction
-- Data Modelling encompasses the standarized schema and formal techniques to manage the data resources in accordance to consistent and, predictive manner.

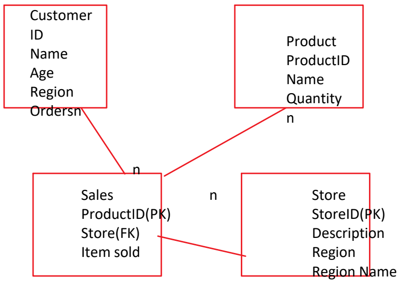
Types of Data Models

a) Conceptual Data Model
b) Physical Data Model
c) Logical Data Model

a) Conceptual Data Model - As per the domains identified - Banking, Retail.



b) Logical Data Model - More clear visuals and based on attributes the relationships has to be derived.



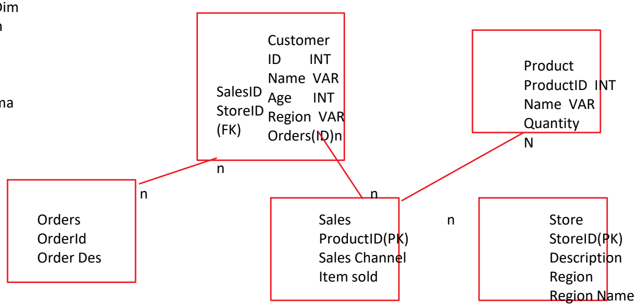
-- Physical Data Modelling -- They provide a schema for how the data will be physically stored within a database. They can offer a finalized design which can be implemented on relational db.

Kinds of tables

Key table is the Fact table
Smaller relationships maintained based on business contexts defined in Dimension tables.

Customer_Fact
Product_Dim
Store_Dim
Sales_Dim

Star Schema



1. Identify the entities
2. Identify the key properties or attributes of the entities
3. Get relationships between the entities and attributes

-- Erwin
-- ER studio

-- Reduce the errors in analytics pipeline
-- improve the app and data pipeline performance

-- ease of data mapping throughout the org
-- improve the communication between BI and developers.

Demo

Building a Star Schema in SQL Server db through SSMS

No	Dimension
1	Product Dimension
2	Store Dimension
3	Order Dimension
4	Date Dimension
5	Territory Dimension

1.	Fact table	Granularity: by each sales order	
	Fact_Sales	1) Sale subtotal 2) Tax amount 3) Shipping cost	
		Granularity: by each product on very order 1) Quantity ordered / product 2) Product subtotal 3) Discount	

Benefits of Star Schema

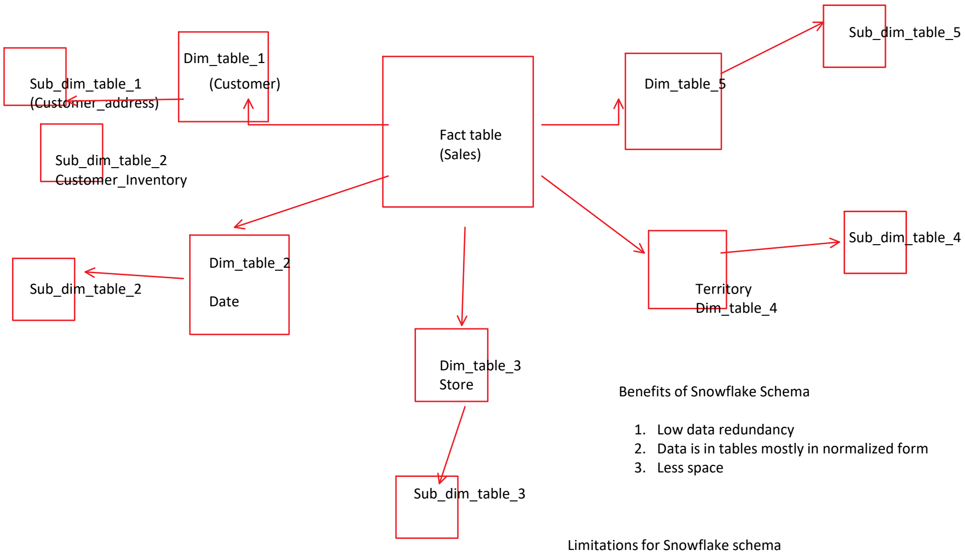
1. Takes less time for the query execution
2. Design is very simple
3. Query complexity is low.

Constraints of Star Schema

1. Takes more space in terms of storage
2. Data is highly redundant since no normalization is done

Snowflake Schema

1. It consists of fact and dimension tables
2. The dimension tables as well as sub-dimension tables are contained.



Benefits of Snowflake Schema

1. Low data redundancy
2. Data is in tables mostly in normalized form
3. Less space

Limitations for Snowflake schema

1. Design is very complex
2. Query complexity is higher than star schema
3. More number of foreign keys
4. Foreign keys are more

Facts Table

-- Features:

1. The Fact table contains the measuring of the attributes of a dimension table.
2. In the fact table, there're more number of records than dimension table.
3. Fact table forms a vertical table.
4. The attribute format of fact table is in numeric and text format.
5. Comes after the dimension table
6. The number of fact table is less than the dimension table in a schema.
7. It is used for analysis purpose and decision making.

Dimension table

-- Features

1. While in Dimension table, there're more number of attributes compared to fact table.
2. Dimension table forms the horizontal table.
3. The attribute format for dimension table is mostly text/varchar format
4. It comes before the fact table.
5. The number of dimension tables are more than fact table in a schema.

Measures:

Measures are the set of aggregates which we want to calculate such as sum of orders or the amount of total sales in a region etc.

A Dimension table stores the data required to define dimensions, dimension attributes and fact table is used to define the measures.

Types of OLAP

1. Relational OLAP (ROLAP) - Star Schema based - data is stored in relational database. Data can be stored multidimensionally on order to view multidimensionally.
   Adding WHERE clause in SQL statement

Star schema

2. Multidimentional OLAP - stores the data on disk in a specialized array structure. OLAP is performed on relying to the multidimensional capacity of arrays.

Here the multidimensional array is stored in a linear collection according to the nested traversal if the axis in pre-determined order.

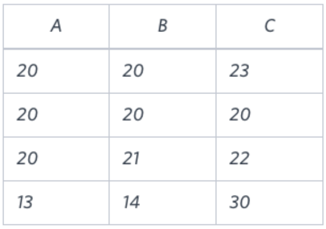
SQL Server Cube. -- boundary to the measures

3. Hybrid OLAP - mixes the features of both relational OLAP and multidimensional OLAP. Allows to store huge volume of data with greater scalability than Relational OLAP.

Benefits: has the faster performance due to facilities of SQL server Analysis service Cube & stored the detailed data, Performance wise much better. Databases are stored in most functional way.

4. Web based OLAP - used for web applications
5. Desktop based OLAP - used for desktop analytical processing
6. Mobile OLAP - used for Mobile apps & analytics

SELECT Case expression
When expression1 then result1
When expression2 then result2
When expression3 then result3
Else result



A	B	C
20	20	23
20	20	20
20	21	22
13	14	30

End

1. Isoscales -

Azure Cloud Fundamentals

08 September 2022 18:45

1. Benefits of Cloud Computing

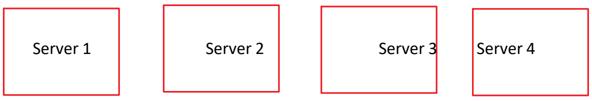
a) Cloud is easy to access for everyone. All the resources like Servers, disks, network, IP address, storage, databases, data warehouses all are accessible to everyone globally through internet.
b) Developing new application and services, storage, databases on cloud
c) It is useful to migrate the existing applications, databases, storage to the cloud as per supportability of the cloud vendor
d) Software on demand
e) Analysis of data
f) Streaming of audio, video, media

Cloud migration --
Capex -> Opex (Operational expenditure)
(capital expenditure model)

a) Scalability - increase the size or number of instances for the servers, databases or storage for the resources on cloud.

-- horizontal scalability (scale out) -- recommended to go for horizontal scalability for long term business.
-- vertical scalability (scale in) --

Use Case 1



b) SLA -- Service Level Agreement -- 99.9% of uptime (managed services) a downtime of 0.001 sec / week/month.
Azure VMs a SLA of 99.99% of uptime -- (a downtime of nanosec/less than millisecond per month)

Associated with each and every cloud resource for any cloud provider.

Cloud provider can compensate for any downtime for missed SLA/ guaranteed uptime.

This is part of cloud based business model for the cloud providers to their customers.

Private Cloud

Features

a) Private cloud is also known the internal cloud or corporate cloud.
b) Private cloud provides a high level of security data is accessible over the internal network only to limited set of users based on accessibility.

e.g. HP data centers, Ubuntu cloud, Azure Stack (Microsoft private cloud), Anthos,
Elastra -private cloud, canonical private cloud, Vmware private cloud

Advantages of private cloud -

a) More control - more control over the resources and hardware rather than public cloud, because it's only available To the selected users.

b) Security & privacy - private cloud has more improved security as compared to public cloud.
c) Improved performance - gives better performance with improved speed and capacity.

Cons:

a) High cost - the cost is higher since the resource set up and hardware resources, software, apps, network, storage/db are all managed internally for a particular org level
b) Restricted area of operations - private cloud is accessible only to a particular org, within the org boundary, Hence the operations are limited.
c) Limited scalability - private cloud can be scaled only with the capacity of the internal hosted resourced within the Org boundary.

Horizontal scalability -

Increase the number of server instances based on requirement -- scale out
Decrease the number of server instances based on the requirement -- scale in

Hybrid Cloud

Advantages

1. Flexible and secure - it provides flexibility of mix of public cloud and secure resources because of private cloud
2. Cost effective - hybrid cloud costs less than the private cloud. It helps the organization to save costs for both of the infra and application support
3. Adaptable - A hybrid cloud is capable of adopting to the increasing demands as per company requirement for disk storage, memory, and application infrastructure.

Constraints:

1. Networking - on-prem network has to connect to the public cloud network, lot of complex network config and settings required.
2. Reliability - depends on the cloud service provider
3. Infra compatibility - with the dual level of infra , a private cloud + public cloud model there is a chance that they are running in different data centers.

	Cloud Model	Benefits	Example
IaaS	Rent for respective infra required for business - Compute-	Less complex Less development cycle	AWS EC2 AWS VPC AWS Subnet AWS EBS

	- Storage/disk - Network/ip - Memory is only managed by the cloud provider & their security Not managed - -- OS patching -- OS image -- application security		Azure VM Azure Network Azure Disk
PaaS	Managed services (compute, storage, database, data warehouse, analytics services) - Compute - Storage/disk - Memory - Network - OS - OS patching - Security for infra managed by cloud providers	Less burden on infra provisioning Developers can focus more into their app/business logic Language agnostic (.net, java/jsp, springboot, react/angular, golang, ruby, python/django)	AWS EBS (Elastic beanstalk), Azure App service (Web apps, Mobile apps, API apps), Azure SQL db, Azure Hadoop services, Azure SQL DW, Azure Analysis services, Azure Data lake services
SaaS	Managed services (compute, storage, database, data warehouse, analytics services) - Compute - Storage/disk - Memory - Network - OS - OS patching - Security for infra managed by cloud providers - Application - Application security All of these will be provided by the cloud provider	Almost zero burden on the end-user in terms of app dev and deployment No burden on app scalability, reliability, resiliency , app backup, DR	Office 365 Salesforce
Serverless	The apps are hosted into this serverless resources where you cant access those servers	Zero downtime Almost 100% scalability Programming Language agnostic	Azure Function AWS Lambda

Use Case 2

Web Server 1
data base server

Memory - 32 gb -> 128 GB
Disk - 5 tb -> 32 TB

30-40 raised request for loan through the app,
Increasing the compute capacity for one specific resource, it's called vertical scalability

Azure Fundamentals

08 September 2022 18:46

Snowflake

1. Create SnowFlake db and Schema

```
create or replace database Retail;
```

```
2.
select current_database(),
current_schema();
```

3. Create SnowFlake Datawarehouse

```
create or replace warehouse retail
with
  warehouse_size='X-SMALL'
  auto_suspend = 180
  auto_resume = true
  initially_suspended=true;
```

```
4.Create table

create or replace table emp_basic (
  first_name string ,
  last_name string ,
  email string ,
  streetaddress string ,
  city string ,
  start_date date
  );
```

1. Azure Fundamentals
2. Azure Global Infrastructure -> Azure Regions, Data center
3. Azure Resource Manager
4. Azure Resource Group
5. Azure Web App
6. Azure Storage
7. Azure Virtual Network
8. Azure Virtual Machine
9. Azure SQL db

Scenario 1: IaaS (Infra as a Service)

Web/App Scenario

SharePoint App + includes the data from
3rd-party API (Mulesoft)

Azure Virtual Machine
Network (IP CIDR)
Public IP address

- Service deployment **Shared Responsibility Model**
-
- Taken care by the Cloud Provider

a) CPU
b) Memory
c) Disk storage
d) Network
e) Security

- Not Taken care by Cloud provider / have to take care by Customer
- Underlying OS
- OS patching and update
- Application deployment
- Security of the Apps and data

Database Scenario

SQL Server db + Includes dependency
like .net objects, CLR stored procedures
SQL Server on Azure VM

Scenario 2: PaaS (Platform as a Service)

Web / App Scenario +

A sample .net/java application includes 3-tier architecture, it includes lot of
.js, html, css files in front-end and lot of worker process(windows batch) process in its business logic

The app deployment on Azure should be cost-effective.

Azure Web app

Shared Responsibility

Managed by cloud vendor

- CPU
- Memory
- Disk
- Network
- Security of Infra

Database

SQL Server db

200-400 GB db size
Scalability -
High availability
Zero downtime

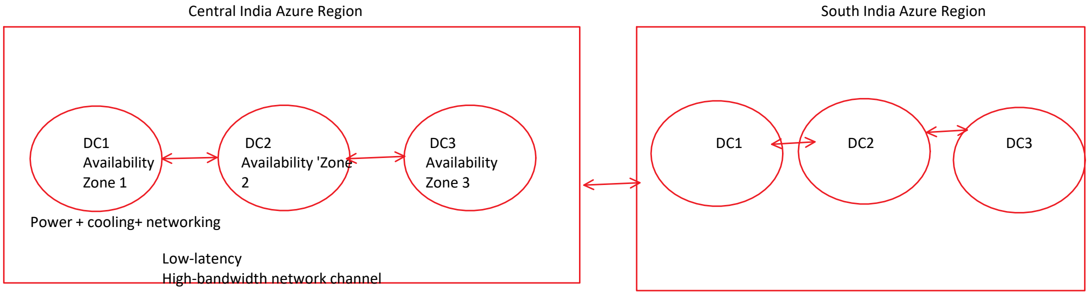
Read Replicas are also required
Azure SQL server + SQL db (Region 1) (primary)
Azure SQL Server + SQL db (Region 2) (Secondary)
(read replica)

Azure SQL database

- Guest OS
- Guest OS update/Patching
- Deployment tools

Not managed by cloud vendor

- Application code
- App security
- Database coding (SQL queries, Stored procedure, Views.. Functions)
- Data security



LRS - 3 copies of storage account in the region (dev/demo)

GRS - Geo-redundant storage (3 + 3) = 6 copies of storage accounts are created across primary and secondary region

India based Azure Regions

Azure Geography - (Market basis it includes one or more regions)

-- Allows the customer the specific data residency
-- manage compliance as per app and data needs to close
-- Geographies are fault tolerant to withstand/prevent the complete regional failure through Dedicated high bandwidth networks.

-- GDPR (compliance)

Azure Availability Zones

-- Availability Zones (AZ) are unique physical buildings within Azure region to protect apps and data from facility level issues.
-- it offers high availability to protect your application + data from the datacenter failure
-- Each Zone is comprised of one or more data centers containing independent power, cooling and networking.

Azure Global Network

Refers to all the components in networking & is part of Microsoft global WAN (wide area network), fibers, cables, routers, switches, PoPs, etc.

Azure Resource Manager

-- The request for creation, updating or deletion of resources On Azure is handled through Azure Resource Manager.
-- Then, Azure resource manager handles the request by authentication and authorizing the request before forwarding It to the specific service

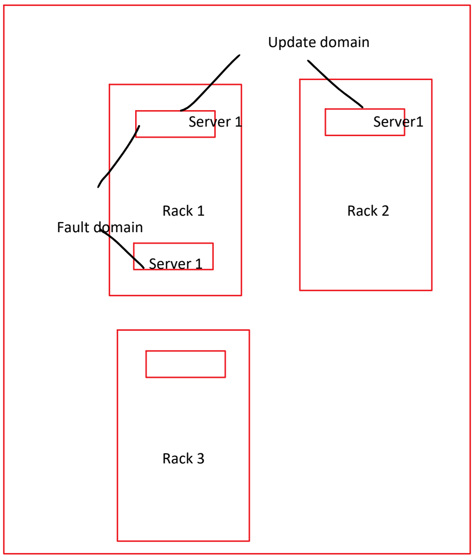
It consists of the following components

a) Resources - A manageable item in Azure
   - Azure VM
   - Azure Database
   - Azure Storage
   - Azure Network
   - Azure App services
   - Subscriptions

b) Resource group - A logical container which holds all of the resources together.
   - Resources to be part of a Resource group
      - VM
      - Storage
      - Web App

Resource Group

- You can decide which resource is to be part of which resource group
- The azure resource to be deployed in the same region of the resource group

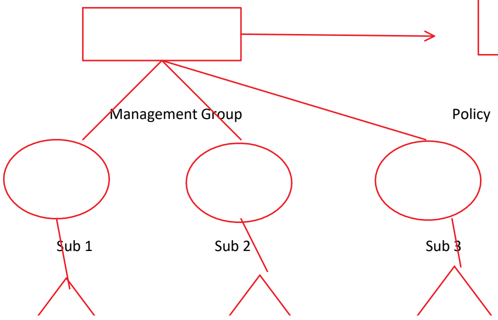


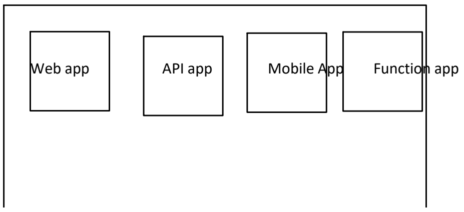
c) Resource Provider - A service which supplies Azure resources. As End user/customer, we don't have to create/configure anything.

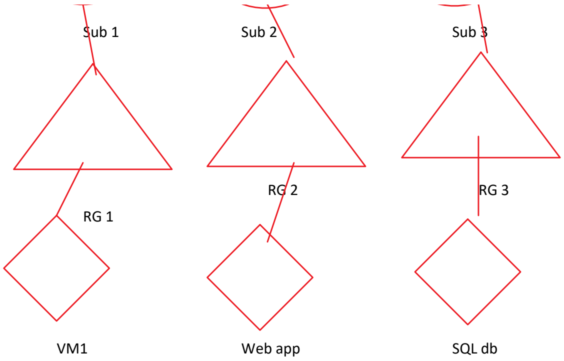
Microsoft.Compute - VM related resources
Microsoft.Web - web app related resources
Microsoft.Storage - includes all storage related resources (blobs, tables, queues, files)

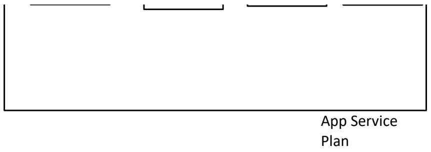
Benefits of Resource Manager

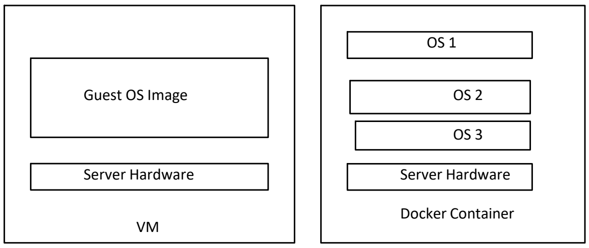
-- Manage your infrastructure as a unit as a template rather than a script
-- deploy, manage and monitor the resource as a group
-- redeploy the solution throughout the development lifecycle (SDLC)
-- Apply tags to your resources and resource groups based on organization requirements (uniquely identify the resources in the container and also it's used for billing purpose as a org unit)











Virtual Network

Fundamental building block for the private network in Azure.

It helps to create many resources accessible over specified network

a) Azure VM
b) Azure Web App
c) Azure Storage
d) Azure SQL db

Vnet is also similar to a traditional network that you can operate in your own datacenter.

Benefits of Azure Vnet

a) Enables secure connectivity to Azure resources from on-prem to this resource through internet
b) Vnet can also help traffic filtering, routing network traffic and integration with other Azure services.

.dockerimage file

Run webserver
Port 8080
Config
Container url:

Pre-requisites	Docker Desktop (Windows/Linux) VS code / Visual Studio
Profile	Create a profile in Docker hub
publish	Publish your local customized images to docker

Basic PowerShell Scripting

08 September 2022 18:46

PowerShell Desired State Extension

1. Configuration management

Example: deployment requires some properties for the server (size, network, storage) and the config of the OS.

2. Compliance

Example: you want to audit or deploy settings to all machines in scope either reactively to existing machines or proactively to new machines as they are deployed.

```
"metadata": {
 "category": "Guest Configuration"
"guestConfiguration:{
"name:"AzureWindowsbaseline"
"version" "1.*"
```

Difference between Cmdlet and Command

1. Cmdlets are .net framework class objects & not just standalone executables.
2. Cmdlets can be easily constructed from a few lines of code
3. In Cmdlets, the parsing, error representation, output formatting are not handled by cmdlets. It's done by Windows PowerShell Runtime.
4. Cmdlets are the process which works on objects not on text stream. Objects can be passed as output for pipelining.
5. Cmdlets are record based as they process a single object at a time.

For(I= initialization; i<= length -1 ; i++)

A regular expression is special sequence of characters that can help to match or find other strings or set of strings using a specialized pattern.

They can help to search, edit, manipulate text and data.

Subexpression	Matches
^	Matches the beginning of the line
$	Matches the end of line
\A	Beginning of entire string
\z	Ending of entire string
\Z	End of the entire string except the allowable final line terminator
[…]	Matches any single character in brackets
[^..]	Matches any single character not in brackets
\w	Matches the word characters
\W	Matches the nonword characters
\s	Matches the whitespace
\S	Matches the nonwhitespace
\d	Matches the digits
\D	Matches the nondigits
\G	Matches the point where the last match finished
\n, \t	Matches newlines, carriage returns, tabs etc.

Providers in PowerShell can provide access to data and the components
That couldn't be accessible easily from the command line.

The data is represented in a consistent format through providers.

Providers are .NET programs which provides access to specialized data stores
For easy viewing and management.

- Alias provider
- Drive : Alias

Certificate provider
 - Drive: cert

- Objects: Microsoft.PowerShell.Commands.X509CertificateStore

Environment provider

- Drive: env
- System.Collections.DictionaryEntry

FileSystem provider
 - Drive: C:
Objects: System.IO.FileInfo, System.IO.DirectoryInfo

- Variable provider
- Drive: variable
- Object: System.Management.Automation.PSVariable

Registry Provider
Drive: HKLM, HKCU
Objects: System.Win32.RegistryKey

Apache Hadoop Overview

20 September 2022 17:56

Constraints and Reasons for evolving Big Data in Software Development

1. Volume aspects - KB, GB, TB, PB, ZB (10to the power9-11) , traditional RDBMS to Big Data pillar

e.g. Twitter feed data, (social sentiment analytics) -> 10 TB , weblog, clickstream

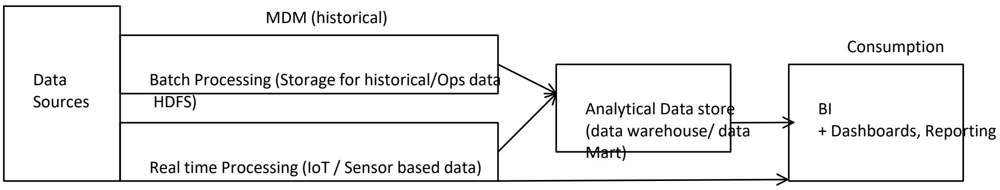
2. To get insights from data - to solve the real time analytical queries and answers as per the business requirements.
3. Data variety - RDBMS + weblog + clickstream + BI -> dashboards (semi-structured data, quasi-structured and unstructured data) , media (audio + video)
4. Velocity - IoT, sensors (latitude, longitude, temperature) , fast rate the data should be processed and transformed. It is defined as the rate in which data is received and acted on. All of data in traditional RDBMS is stored on disk where as the real time data streams can be directly processed in memory instead of disk.

Volume

Velocity

Variety

Velacity -- measurement of different aspect of data, twitter feed data, clickstream , weblog

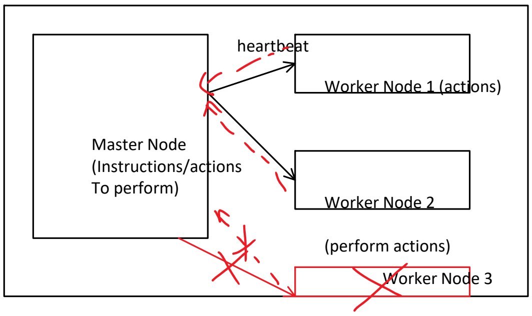

MDM (historical)
Consumption
Data Sources
Batch Processing (Storage for historical/Ops data HDFS)
Real time Processing (IoT / Sensor based data)
Analytical Data store (data warehouse/ data Mart)
BI
+ Dashboards, Reporting

Azure - Hadoop(HDFS), Event Hub (real time streaming engine for processing)

(Social Media,
Website(Clickstream, weblog)
Weather data from Sensors

Big Data Architecture - Pipeline Ver 1

Distributed Systems

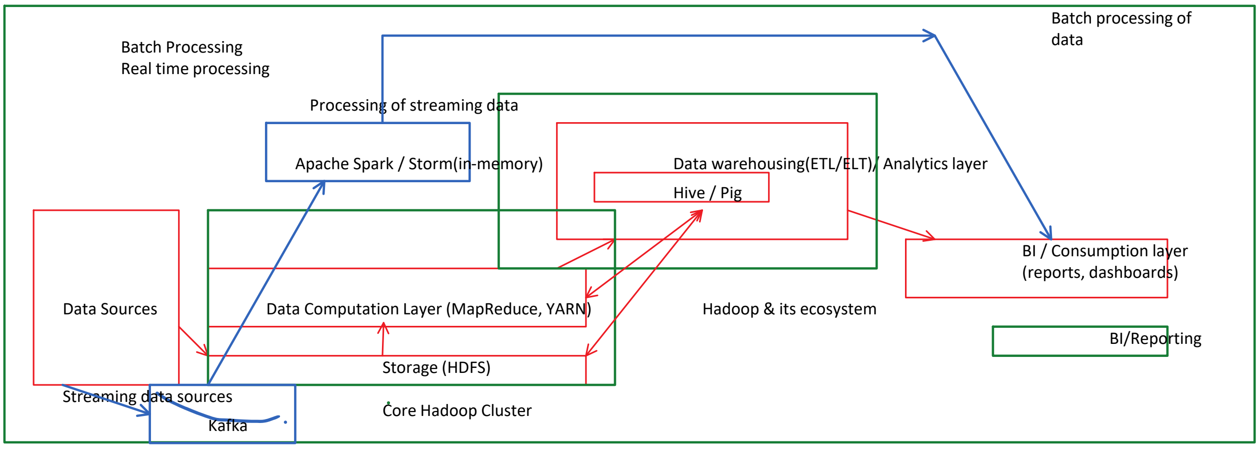


Apache Hadoop

a) Hadoop Distributed File System (HDFS) -> Storage layer
b) MapReduce -> (Computation layer)
c) YARN -> (Yet another Resource negotiator) (MapReduce V2)

Rest of Apache Hadoop ecosystem components

a) Apache Hive - Data warehouse and analytics tool in Hadoop
b) Apache Pig - data querying tool with scripting language (Load, Transform & DUMP)
c) Kafka - Real time stream processing engine



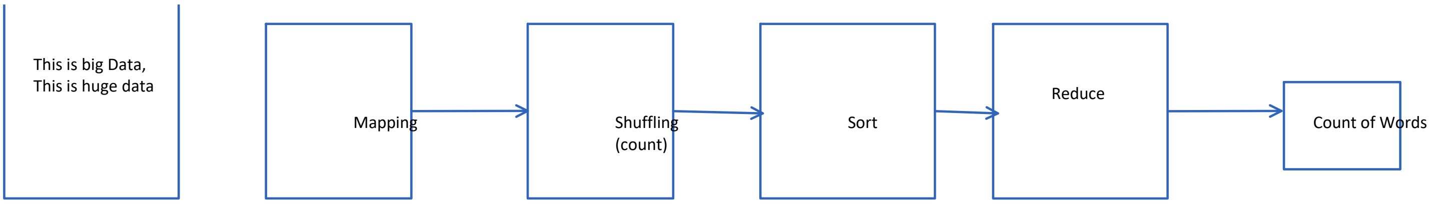
Data ingestion, extraction, initial processing

Big Data Pipeline - Ver 2

Batch Processing = Hadoop & its ecosystem (HDFS, Mapreduce, Pig and Hive)
Real time processing = Kafka + Spark

Apache Spark
Apache tez

Count the number of words



500 - 1 PB

Mapper jobs

Job_1 = big, bigger, not big
Job_2 = data
Job_3 = is

Mapper class

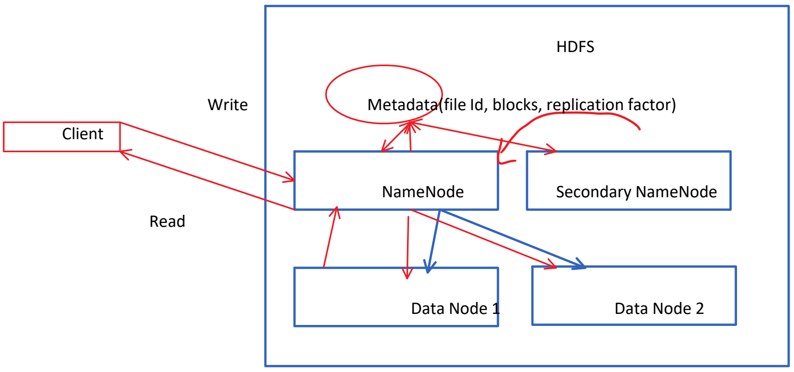
Big = 100
Not big = 300
Bigger = 500
Is = 300
This = 500

(bubble sort, quick sort)

Big & not big, Bigger = 900
Is = 1000
This = 2000

Big Data Use Cases

1. Product Development - Social media
2. Predictive maintenance - Aerodynamics,
3. Customer experience - Telcom, Banking
4. Fraud and Compliance - Banking, Financial
5. Drive innovation - BFSI, Healthcare, marketing
6. Operational efficiency -



Easy to debug and readable
HDFS is the default storage for Hadoop ecosystem components

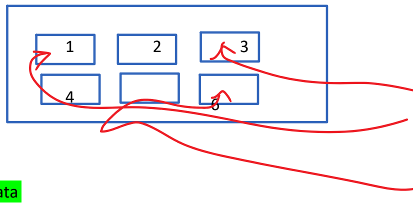
Namenode - (Master node)
a)check the heartbeats from the datanodes
b) Sends the instructions to the datanode

Secondary Namenode- (Master node)
a) Works as a replication for primary namenode
b) In case of primary namenode failure, the secondary namenode Becomes the primary node
c) Once recovered from failure, the old primary namenode becomes the new New Secondary namenode.

Datanode - (Worker/slave node)
a) Performing all kinds of data storage in the storage blocks
b) Performing actions for data storage as per namenode

Data storage principles

a) In HDFS, data is stored in blocks which are called as data blocks.
b) Data blocks are small as much as individual granular block with max size of 126 MB (e.g. Hadoop v2)



Data is appended as per blocks

This is big data
This is good data 10*1024*1024 / 128 = 81920 blocks in HDFS datanode

Computation should take place in the location of data

Neither data should travel to the location of compute.

Apache Pig

a) It can handle structured, semi-structured or unstructured data & stores the data into the HDFS.
b) Every task or query running on pig is executed through MapReduce.

Benefits:

a) Ease of programming:
b) Optimization of complex data processing (Load, transform, DUMP)

c) Flexible and with in-built operators (sort, filter, join, foreach)
d) Supported for Avro based file format (row wise)

MapReduce	Apache Pig
It is a low level data processing	It is a high level data flow tool
Complex programs through java/python	Simple pig latin which are simpler also less in no of lines
Data operations are difficult compared to pig	Pig latin these built-in operators are available
It does not allow nested data type	It provides the nested data types like tuple, bag, and map

Apache Hive

Built on top of Apache hadoop, it provides the following features

a) Tools to enable easy access to data via SQL. SQL like query interface which is called HQL.
b) Enables us to perform data warehouse tasks such (extract, transform, load) (ETL) operations, analysis & reporting.
c) Query execution is done through MapReduce, Apache Tez (framework for faster data processing in Pig and hive), Apache Spark.
d) Procedural language with HPL-SQL
e) Sub-second based query retrieval support when using with Hive LLAP, Apache YARN.
f) Hive supports different file formats with lot of built-in connectors like (CSV/TSV), Apache Parquet, Apache ORC (columnar file)
g) Traditional data warehouse workloads.

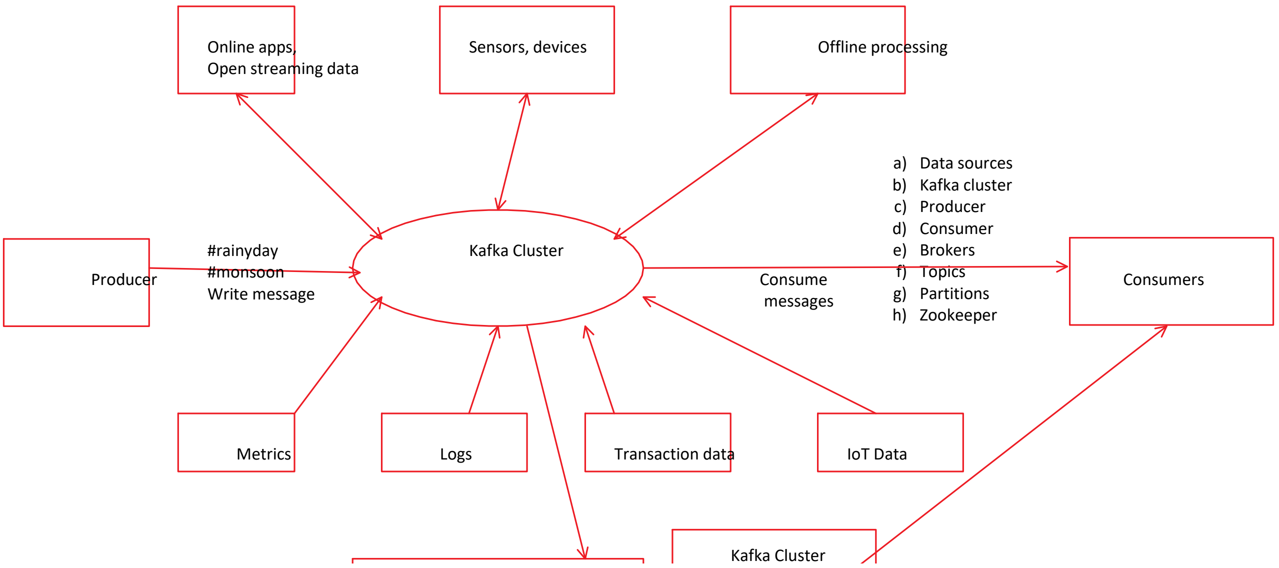
Apache Pig	Apache Hive
Pig operates on the client side	Hive operates on the server side of the cluster
Pig uses the pig-latin script for analysis	Hive uses hiveql (hql) language
Pig latin is purely procedural language	Hive is declarative SQL support
Pig has support for Avro file format	Hive has support for Parquet & ORC
Pig is suitable for complex and nested data structure	Hive is suitable for batch processing OLAP systems
Pig does not support schema to store data	Hive supports schema which can used for data insertion into table
Pig does not have support for JDBC/ODBC	Hive has the support for JDBC & ODBC
Pig does not have any metastore (metadata store)	Hive has metastore with support for various DDL language (SQL, mySQL, postgreSQL)
Pig operates quickly & data loads quickly	Hive loads data slowly
.pig is extension for pig-latin scripts	Any file formats (.hql) file format

Azure HDInsight is 100% core Apache Hadoop platform

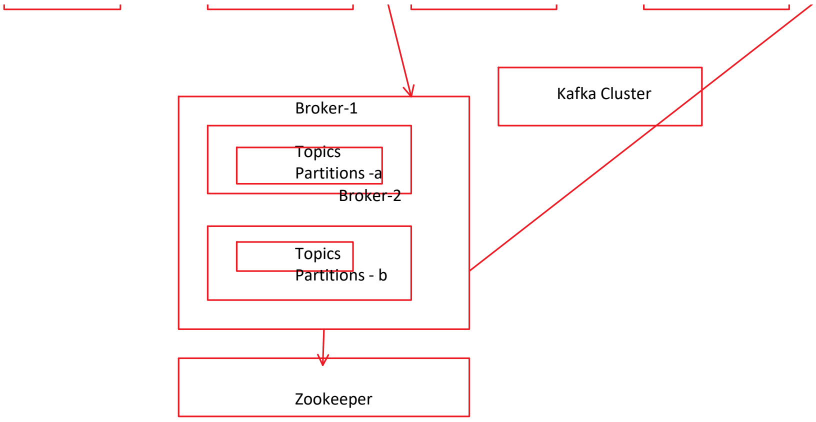
Hadoop ecosystem Tools Supports

- HDFS
- MapReduce
- YARN
- Pig
- Hive
- Kafka
- Spark
- Storm
- Sqoop

Apache Kafka works as a real time streaming engine which helps in the ingestion of data. It provides a fault-tolerance, distributed real time streaming platform where the producers can produce the data from the various data sources, then get processed to get it consumed by the consumers.



a) Data sources
b) Kafka cluster
c) Producer
d) Consumer
e) Brokers
f) Topics
g) Partitions
h) Zookeeper



Broker-1
Topics
Partitions -a
Broker-2
Topics
Partitions - b
Kafka Cluster
Zookeeper

Apache Hadoop (Deep Dive)

08 September 2022 18:46

Reason behind choosing different file formats support in HDFS

1. Faster read time
2. Faster write time
3. Splittable files
4. Schema evolution support (modify the data fields)
5. Advanced compression support (gzip, LDAP, snappy)
6. Snappy data compression can lead to high speed and reasonable less data latency travelling over the network
7. File formats can help to manage the diverse data set.

Why Data serialization for storage formats?

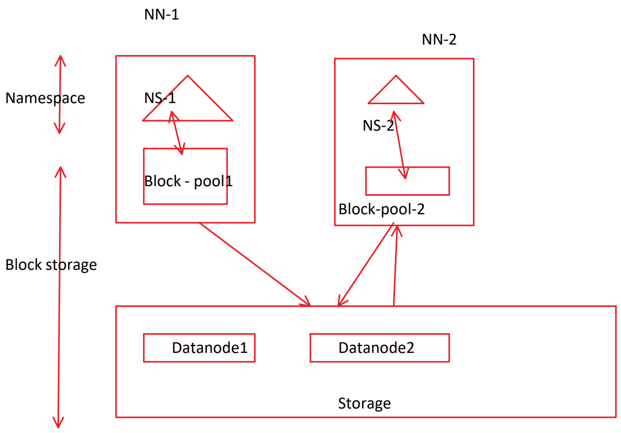
1. To process the dataset faster
2. Whenever proper data formats are required to maintain and transmit the data over the network without schema support on another end.
3. Data without structure or format whenever requires to process, complex errors can occur, data deserialization can help through metadata.
4. Serialization can help data validation over transmission.
   Benefits

1. Compact
2. Fast
3. Extensible and interoperable
4.

File Formats	Benefits (pros/cons)
CSV file format	Requires compression support, using CSV in HDFS can increase the reading performance cost, schema support is also within the limit.
JSON	Better performance than CSV, records are retrieved even faster
AVRO	Row oriented file format and data serialization framework
Sequence	Complex reading operations
RC	(Row-Columnar) file format where write operations are comparatively slower, optimized RC (ORC) they can have better support , read operation is average
Parquet	As a columnar file format, optimized for storing data, processing and analysis

HDFS Namenode block management

1. Provides the datanode level cluster relationship by handling individual registrations, periodic heart beats.
2. Processes various data blocks and maintains the location of these data blocks within the datanode
3. Supports block related operations such as CRUD ops, of the blocks
4. Manages the data blocks replication, block level replication for under replicated blocks and allowing core read \write operations.



Block-pools are a set of blocks responsible to belong to a single namespace, whereas the Datanodes store blocks for all the block pools in the cluster. Each and every block pool is Managed independently, allows the ns to generate the blockids for new blocks to co - ordinate with other ns.

A ns and block pool together known as namespace volume.

ClusterID - uniquely identify the nodes in the cluster. When a namenode gets formatted, the identifier is either provided or it can be auto generated. This cluster ID is used for formatting other namenodes in the cluster.

Key benefits for HDFS federation

1. Namespace and namenode scalability - federation adds the namespace horizontal scalability. Large deployments or deployments using a lot of small files can be benefitted by namespace scalling allowing more namenodes.
2. Performance - file system throughput is not limited by a single namenode, adding more nn can scales the cluster for better throughput and read/write ops
3. Isolation - multiple namenodes can help to manage different categories of apps and users can be isolated to different namespaces.

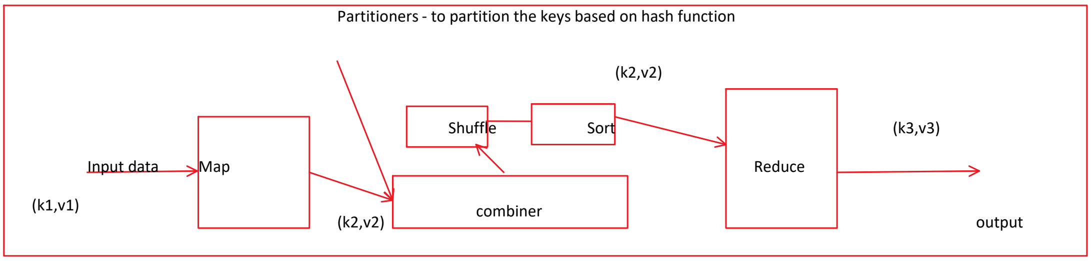
ALT + CNTL = coming to primary
CNTL + G = insert to VM

1. ResourceManager -- Master node
2. NodeManager -- Worker node
3. MRAppMaster - per application

Mapreduce Applications need to specify input/output locations and map and reduce functions via implementations of appropriate interfaces and/or abstract classes. There're other job parameters, defined as job configuration.

Mapreduce as a framework consists of <key, value> pair model. The framework views the input to the job as a set of <key, value> pairs and produces a set of <key, value> pairs as the output of the job.

- Key, value pairs has to be serializable by the framework and need to implement Writable interface. The Key classes have to implement the WritableComparable interface to facilitate the sorting by the framework.
- (input) <k1, v1> -> map -> <k2, v2) -> combine -> <k2,v2) -> reduce -> (k3, v3) -> output
- Combiner to save the bandwidth as much as possible by minimizing the number of key/value pairs which will be shuffled across the network and to be provided as an input to the reducer.



Partitioner - paritioning of keys coming from the intermediate map output and is being controlled by Partitioner. The partitioner is used to derive the partition.
On the basis of key-value pair, each map output gets partitioned.
Default partitioner (Hash partitioner) computes a hash value for the key and assigns the partition based on the result.

In a MR job, the mapper executes first , then combiner then partitioner

Mapreduce has its own data type called Writable.

This Writable data type implements the WritableComparable interface.
The WritableComparable interface is a combination of Writable and ComparableInterface.
The Writable data types works with the following data types:

Integer -> IntWritable = it is the hadoop variant of integer. It is used to pass the integer nos and key or values.
Float -> floatWritable = used to pass the floating point numbers as key or values
Long-> LongWritable = used to pass hadoop variant of long data type as key/values
Double->DoubleWritable = pass the double to store double values
String -> Text = hadoop variant of string to pass characters as key/value
Byte-> ByteWritable= hadoop variant of byte to store sequence of bytes
Null-> NullWritable-> hadoop variant of null to pass null as key/value.

The Comparable interface is used for comparing when the reducer sorts the keys, and Writable can write the results back to local disk.

Hive Internal Table	Hive External Table
Managed table where the data gets loaded directly from your local drive	Hive external table is more efficient where it follows the loose coupling and the data gets loaded from HDFS
Entire lifecycle of hive table is managed by hive itself, DDL, DML operations and underlying datasets	Data gets loaded into the table from HDFS
Once the internal table is dropped, the underlying table's data gets deleted	Once the external table is dropped, the underlying table's data doesn't get deleted, because the data is stored in HDFS
Create table 'tablename'	Create external table 'tablename'
Data Path has to be specified from local drive	Data path has to be specified by 'HDFS path'

Partitioning in Hive - Apache Hive organizes the tables into partitions for grouping same type of data together based on a column or partition key.

- Each table in hive can have one or more partition keys to identify a particular partition.
- Using partition, it makes faster to the querying of the data or slicing of the data.

E.g. using the stu_dept column is a partition column / partition key.

Pros & Cons

Pros -

1. It helps to distribute the execution load horizontally
2. In partition, faster execution of queries can happen with low volumes of data.

Cons -

1. There's a possibility that, too many small partitions can be created, -- too many small directories
2. Partitioning can be effective low volume data (GB-TB), for huge volume of data (PB) level, it takes time to process the data through query

Bucketing - Once the partition is done, then the hive tables or partition can be further subdivided based on hash function of a column in the table to give an extra structure to the data for efficient querying purpose which can be called as bucketing

Problem: Datanode is not running

pre-requisite

```
# stop all running hadoop services
stop-all.sh
```

Steps:

1. Remove the contents from datanode & directory

```
sudo rm -r /home/ani/hadoop/hdfs/namenode/datanode
```

2. format the namenode

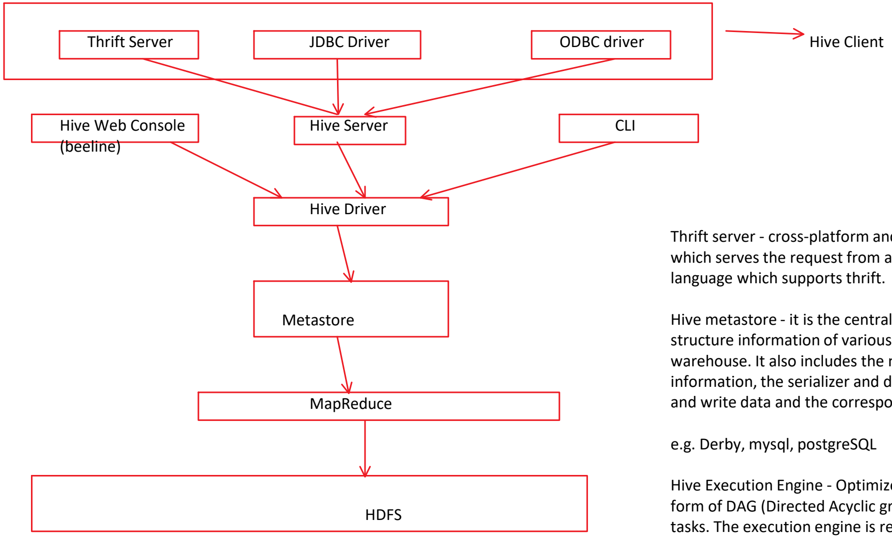
```
hdfs namenode -format
```

3. start-dfs.sh
start-yarn.sh

Pros -

- It provides faster query response like partitioning
- In bucketing due to equal volume of data in each partition, joins at the Map side will be quicker.

Cons -
We can define a number of buckets during the table creation. But loading of data into the buckets has to done manually by developers.



Thrift server - cross-platform and cross-language service provider which serves the request from all the supported programming language which supports thrift.

Hive metastore - it is the central repository which stores all the structure information of various tables and partitions in the warehouse. It also includes the metadata of the column and its type information, the serializer and deserailizers which are used to read and write data and the corresponding HDFS files where data is stored.

e.g. Derby, mysql, postgreSQL

Hive Execution Engine - Optimizer generates the logical plan in the form of DAG (Directed Acyclic graph) of map-reduce tasks and HDFS tasks. The execution engine is responsible for executing the incoming hiveql queries in the order being executed.

Datatypes in Pig

Type	Description	Example
Int	Signed 32 bit integer	4, 40, 300
Long	Signed 64 bit integer	15L
Float	32 bit floating point	2.5f, 5.5F
Double	32 bit floating point	1.5, 1.5e2
charArray	Character Array	Hello World
Tuple	Ordered set of fields	(12,43)
Bag	Collection of tuples	{(12,43), (55,100)}
Map	Collection of tuples using characters	[hello#world]

Operators in Pig

LOAD - LOAD is a relational operator, used to load the data from the file system

LOAD from 'hdfs://localhost:9000/pig_data' as (id:int,name:chararray) USING PigStorage;

DISTINCT Operator

Eliminate the duplicate tuples.

FILTER Operator

Is used to remove duplicate tuples in a relation, also sorts the given data and then eliminates the duplicates.

FOREACH Operator
Generate the data transformations based on columns of data. It is recommended to use foreach operator to work with tuples of data.

GROUP Operator

GROUP operator is used to group the data into one or more relations. It groups the tuples which contains a similar group key. If the group key has more than one field, it treats as tuple otherwise it will be the same type as that of the group key. Hence, GROUP operator provides a relation that contains one tuple per group.

ORDER BY Operator

Sorts a relation based on one or more fields. It also maintains the order of tuples.

LIMIT operator

Is used to limit the number of output tuples.

JOIN - to join two or more relations
COGROUP - to group the data into two or more relations
CROSS - to create cross product of two or more relations
UNION - to combine two or more relations into a single relation
SPLIT - to split a single relation into two or more relations.
DUMP- to print the contents of a relation in a console

Relation_name = LOAD '<input_file_path>' USING Function as schema;
DUMP relation_name;

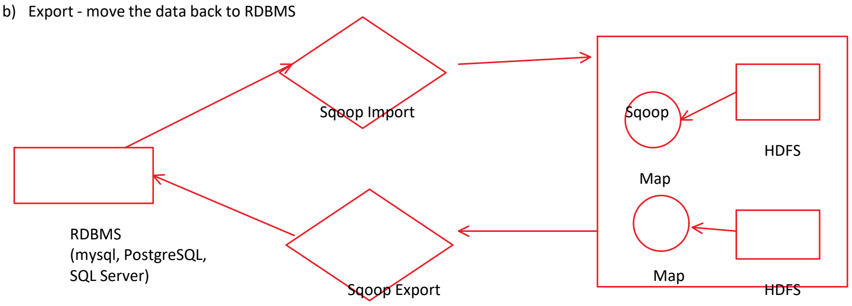
Relation_name = the relation in which we want to store the data
<input_file_path> = defines the hdfs data directory

Function - choose the set of functions from Apache Pig framework (PigStorage, TextLoader, JsonLoader, BinStorage)
Schema - define the schema of the data, (col1:data_type1,col2:data_type2….);

Apache Sqoop

1. Sqoop helps to connect the result from the SQL queries into the HDFS
2. Sqoop also helps us to load the processed data directly into the hive / Hbase
3. It performs the security operations of data with the help of Kerberos
4. With the help of Sqoop, we can perform compression of the processed data
5. Sqoop is also highly powerful and efficient in nature
6. Sqoop helps to perform ETL operations in a very fast and cost-effective manner
7. With the help of Sqoop, we can perform the parallel processing of the data which leads us to fasten the overall process.
8. Sqoop uses the MapReduce mechanism for its operations which also supports the fault tolerance

a) Import - importing data from RDBMS to hadoop cluster
b) Export - move the data back to RDBMS



File formats

a) Delimited texts (default format) (--as-textfile), non - binary data types
b) Sequence Files - binary format used for individual records in custom record-specific data types. (VARBINARY)
c) Avro data file format are compact, binary format
d) BLOB, CLOB data type (--mysql-delimiter) , --lines-terminated-by-<char>, --escaped-by<char>, --inline-job--limit

Conditional Import

1. Append
2. Lastmodified

--incremental argument can be specified to define the incremental import to perform

Append mode can be defined when importing a table where new rows are continously being added with increasing row_no values. We can check the column containing the row_no value with --check-column argument. Sqoop imports the rows where the check column has a value > one defined in the --last-value option

Flume Event

A flume event is a basic unit of data which needs to be transferred from source to destination

Flume Agent
Flume agent is an independent JVM in Apache Flume. Agent receives events from clients or other Flume agents and passes it to the next destination which can be a sink, or other agents.

Agent -> source, channel, sink

Source - Avro source, thrift source, twitter 1% source , Exec source

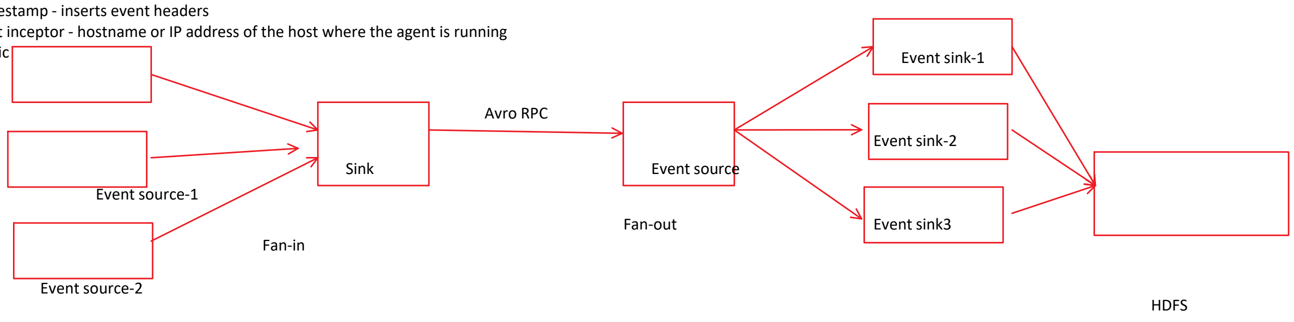
Channel - File system channel, JDBC channel or Memory channel

Sink - HDFS sink

Flume Inceptors

a) Channel selectors - which channel to be selected for transferring the data when multiple channels are existing.
   -> default
   ->Multiplexing
b) Sink Processors - invoke a particular sink from a group of sink

a) Timestamp - inserts event headers
b) Host inceptor - hostname or IP address of the host where the agent is running
c) Static



Event source-3

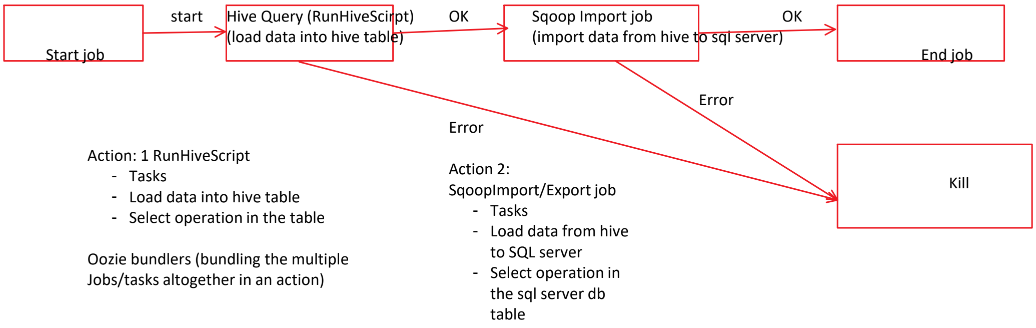
Flume consolidation and multi-hop flow

c) Static interceptor - append a static header with static value to all events, it's possible with the static flume interceptors.

d)

```
A1.sources.r1.interceptor.type = static
A1.sources.r1.interceptor.i1.key= datacenter
```

Oozie workflow


Start job
start
Hive Query (RunHiveScirpt)
(load data into hive table)
OK
Sqoop Import job
(import data from hive to sql server)
OK
End job
Error
Error
Kill

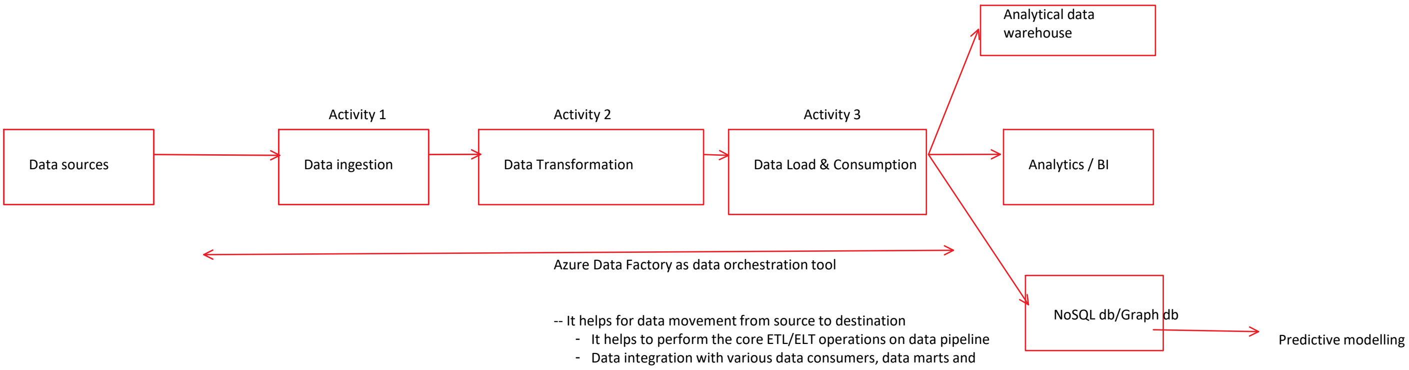
Action: 1 RunHiveScript
- Tasks
- Load data into hive table
- Select operation in the table

Oozie bundlers (bundling the multiple Jobs/tasks altogether in an action)

Action 2:
SqoopImport/Export job
- Tasks
- Load data from hive to SQL server
- Select operation in the sql server db table

Azure Data Factory

08 September 2022 18:47



-- It helps for data movement from source to destination

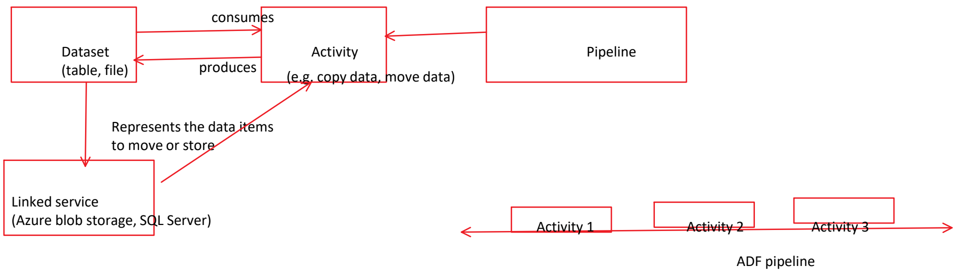
- It helps to perform the core ETL/ELT operations on data pipeline
- Data integration with various data consumers, data marts and data warehouses
- Azure data factory allows us to create data driven workflow
- In Azure data factory, we can create data pipeline from data ingestion, transformation to consumption
- We can use any Azure based services, 3rd - party services are also supported for importing and exporting data.
- Code free data analytics service which can help data ingestion, data pipeline end to end from data ingestion, consumption through its data flows, control flows we can monitor and schedule.

1. Linked Service - creates a linking connection between the data source and the Azure Data Factory pipeline.
2. We must have to create the linked service to link up the data source to the Data Factory

e.g. database connection strings which define the connection information required for the service to connect to the external resource

- Azure storage linked service connecting the storage account to the ADF service. (connection string of the storage account)

Is a logical consolidation of multiple activities

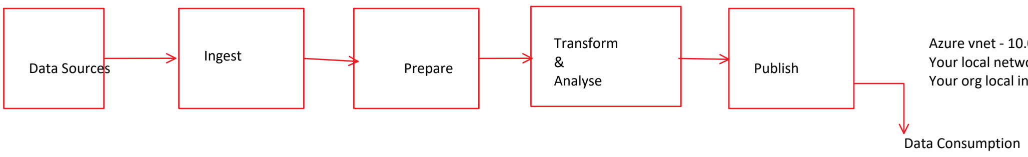


Activity - It is the resource defines us which action/operation to perform on the data.

(copy data, move data,
Aggregation of data,
Schema enrichment,
Looping , joining, combining,
Complex data analysis,
User-defined customized operations)

Microsoft.Sql
Microsoft.Storage
Microsoft.Synapse
Microsoft.DataLakeStorage
Microsoft.DataLakeAnalytics
Microsoft.Databricks
Microsoft.AnalysisServices

Copy Data Activity - A hybrid data connectors



Azure vnet - 10.0.0.0/24
Your local network - 172.16.35.25
Your org local intranet - 192.168.20.45

- The copy activity is executed on an Integration Runtime

a) Accessible over the IP address, integration runtime is responsible to connect to the underlying data sources based on the private cloud/network securely, reliably along with scalability.
b) When we are copying the data to and from the data sources located on-premise on in network access control (Azure vnet, private cloud network), in case of on-premise or private cloud network, we have take help of self hosted integration runtime of data factory

Data copy
a) (self-hosted integration runtime)

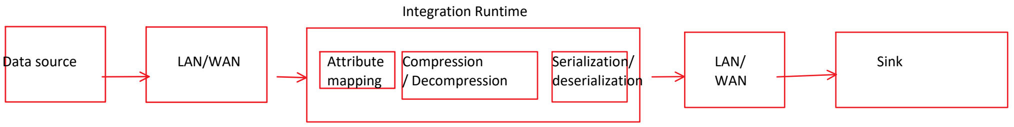
- On-premise SQL server/mysql/openshift based db --> azure database

---> private cloud based data sinks

b) (Azure Integration runtime) (default)

- Azure , AWS, any cloud based resources data transfer/data copy operation
-

c) Azure-SSIS

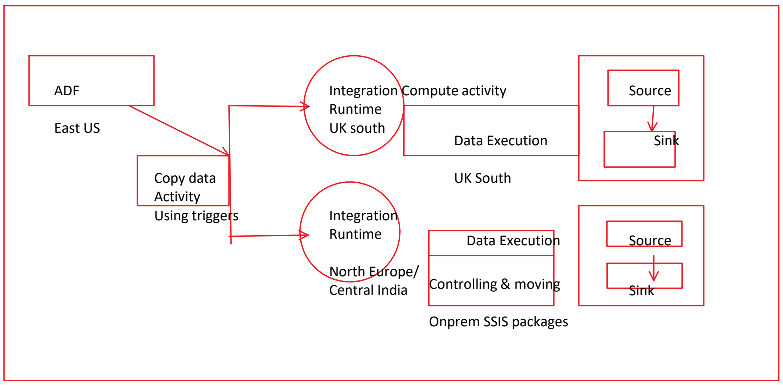


- Read the data from the data source
- Perform the serialization / deseralization of the data transformation, compression/decompression, attribute mapping, schema enrichment, schema drift checking , performing the configuration of input dataset, output dataset and manages the end to end copy activity in the ADF pipeline
- Writes the data back to the sink (based out same public cloud/private cloud or network)

Integration Runtime is the compute infrastructure used by ADF to provide the facility to connect to various data sources and sinks across different regions around the globe for data integration purpose.

It provides the benefit of connecting across the different network environment through the ADF integration runtime.

a) Azure Integration Runtime -- data copy, move, transfer, transformation activity
b) Self-hosted integration runtime -- data copy/move/transform activity through on-prem/private cloud network
c) Azure-SSIS integration runtime -- data copy/move/transform activity for SSIS data packages to Azure



Azure Integration Runtime (IR)

Azure	Public network	Private link
	Data Flow	Data Flow
	Data Copy	Data Copy
	Data Movement	Data Movement

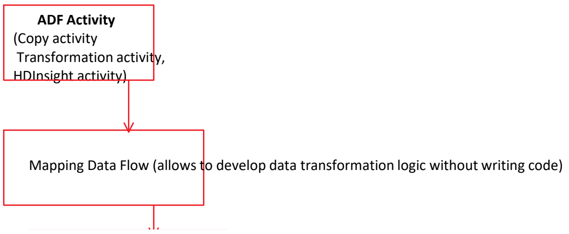
- Able to connect to multiple data sources and sinks across public and private network (private endpoint on Azure over private link)
- Run copy activity (from Azure storage account deployed over private endpoint of private network to the another cloud resource (AWS S3 bucket deployed over AWS private VPC/ private network)
- Various transformation activity - copy activity, pig activity, MapReduce activity, spark activity, Azure Data Lake activity, web activity, copy activity
- Fully managed, serverless compute resource in Azure
- Elastic scalability

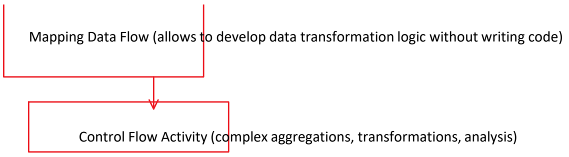
Lab-2:

Copy Data Pipeline (Goal for this lab)

a) Create Azure SQL server and Database (sample)
b) Keep the data in .csv format within ADF pipeline
c) Create the sink and ADLS Gen2 (Azure data lake storage account) (created in previous lab)
d) Copy the data from SQL db to ADLS gen2 in csv format

Pre-requisites (ADF v1)
Visual Studio 2013 / 2015
Azure Data Factory tool for VS 2015


ADF Activity
(Copy activity
Transformation activity,
HDInsight activity)
Mapping Data Flow (allows to develop data transformation logic without writing code)



Mapping Data Flow (allows to develop data transformation logic without writing code)

Control Flow Activity (complex aggregations, transformations, analysis)

Mapping Data Flow in ADF

- Mapping data flows are core part of data transformation in ADF.
- The resulting data flows can be executed as activities within the Data Factory pipeline that use the scaled -out apache spark clusters.
- Data flow activities can be operationalized using the existing ADF schedule, control flow and monitoring capacity.

Authoring Data Flow Components :
Mapping data flow has a unique authoring canvas designed to build transformation logic easy. The data flow canvas us separated into three parts -

a) The top bar - searching of options like data flow debug mode
b) Graph bar - displays the transformation stream while apply data processing logic. It also shows about the lineage of the source data as it flows to one or more sinks.

We can select "Add source" to add a new data source,
We can select "add new transformation" to add a new transformation logic

c) Configuration panel - shows the settings specific to the selected transformation. All the parameters can be defined onto it.

Schema Drift

It is the scenario for the data sources when often there's a change in metadata, fields, columns and types which can be added, removed or modified.

Without handling the schema drift, the data flow becomes vulnerable to upstream data changes, as part of transformation logic

Typical ETL patterns fail when the incoming columns and the fields get changed because they tend to tied to the typical data source names.

-- define sources which can provide mutable data types, field names and values/sizes
-- define transformation parameters which can work with the data patterns instead of hard-coded values and fields.
-- define the expression which can understand the patterns to match incoming fields instead of using named fields.

Flowlets

A reusable container of ADF activities which can be created from an existing mapping data flow or started from scratch.

With the flowlets we can create logic for doing data operations like address changing/string trimmer, mapping the input and output to the columns based on the calling data flow for a dynamic streaming (used for code reusability purpose).

Pre-requisites for Data Flow Lab

1. Azure Data Factory (create using Azure portal)
2. Azure Data Lake Storage (ADLS gen2) created (lab 1) (managed identity permission to be given)
3. Azure SQL db with sample adventure works database (lab2)
4. Create the Linked Services for ADLS and Azure SQL db (lab2) in ADF
5. Create the dataset for ADLS and Azure SQL db (lab2) in ADF
6. Create the pipeline with Copy data Activity

Create the ADF Mapping Data Flow - Lab 4

a) Created few datasets with ADLS
b) Created the actual Data flow in ADF
c) Connected with the ADLS dataset sources & did the basic transformation(select -> column mapping, remove duplicates, Lookup -> joining with product stream with productcategory and productmodel stream)
d) Created the Sink on the Dataflow using ADLS gen2 dataset within the same container which contains the output from the dataflow using parquet format, highly performance efficient, compression enabled, columnar based file format which is faster processing
e) To run the dataflow, an ADF pipeline has created with the data flow activity there used the existing built data flow, then we debug the pipeline to run the data flow.

Control Flow Activity

The Control Flow activity allows to build complex , iterative processing logic within the ADF pipeline

Activities	Description
Append variable	Append variable activity could be used to add a value to an existing array variable defined in a ADF pipeline
Set Variable	Set Variable activity can be used to set the value of an existing variable of type String, Bool or array defined in the ADF pipeline
Execute Pipeline	The Execute pipeline activity allows a ADF pipeline to invoke another pipeline
If condition	If condition activity allows directing pipeline execution based on the evaluation of certain expressions
Get Metadata	Get Metadata activity can be used to retrieve metadata of any data in ADF
ForEach	The Foreach activity can be defined a repeated controlling process in the ADF pipeline
Lookup	Lookup activity can be used to retrieve the dataset from any of the Azure supported data sources

Activity	applying lookup stream for retrieval of data mainly from left side table. Similar to left outer join in SQL. It is used to reference the another stream
Filter Activity	Can be applied on the pipeline to define a filtering expression to an input array
Until Activity	Until Activity allows to execute the set of activities to put in a loop until the condition associated with the activity evaluates to True.
Wait Activity	Allows to pause the pipeline execution for the specified time period
Web Activity	Web activity can be used to call a custom REST endpoint from the ADF pipeline
Azure Function	Allows to run the Azure Function in the ADF pipeline. Azure Function is a serverless core Azure PaaS service part of Azure App service. It allows to build cross-platform apps using any standard OOPS language, it helps to trigger the app/data pipeline to real time based on certain conditions e.g. Complex data processing scenario, an alert message can be triggered from the ADF pipeline based on new conditions, workflow processing as defined.

ADF Pipeline Variables

The process of creating ADF pipeline variables is similar to create the parameters. Unlike the parameters, the ADF variables can only be three data types.

- String
- Boolean
- Array

Add Dynamic Content - This window allows to build dynamic expressions interactively using the system variables and functions.

We'll use two kind system variables - a) Pipeline name
b) Pipeline Run Id
Function - Collection functions, Conversion functions, Date Functions, Logical functions, Math functions, String functions - Concat(), append(),

Lab 5: Build a reusable Pipeline

Pre-requisites

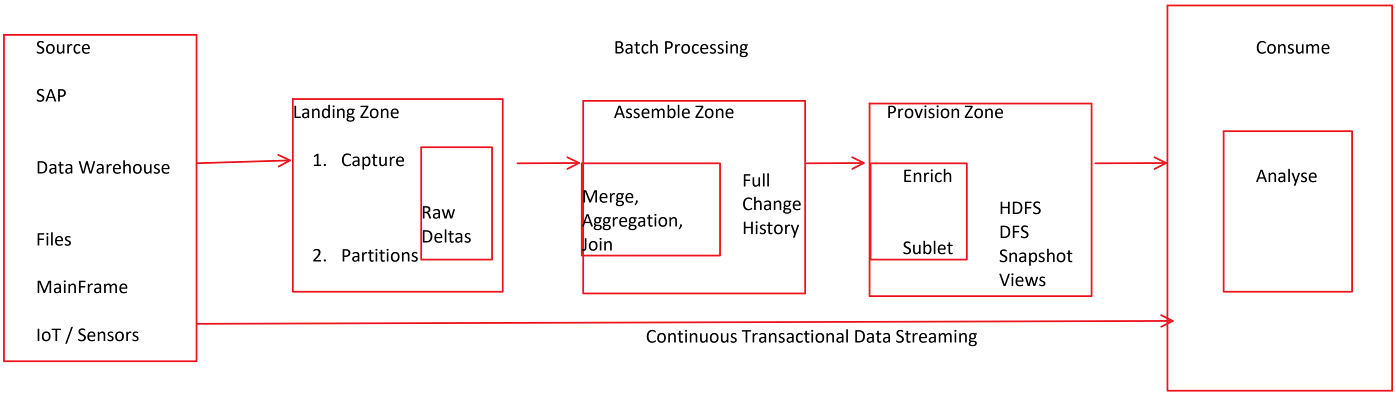
1. Azure Data Factory (create using Azure portal)
2. Azure Data Lake Storage (ADLS gen2) created (lab 1) (managed identity permission to be given)
3. Azure SQL db with sample adventure works database (lab2)
4. Create the Linked Services for ADLS and Azure SQL db (lab2) in ADF

Goal:

a) Create a reusable dataset (using dynamic content)
b) Create a reusable pipeline to retrieve the data from SQL dataset dynamically (10 different tables) & sink to ADLS storage.

Azure Data Lake Gen2

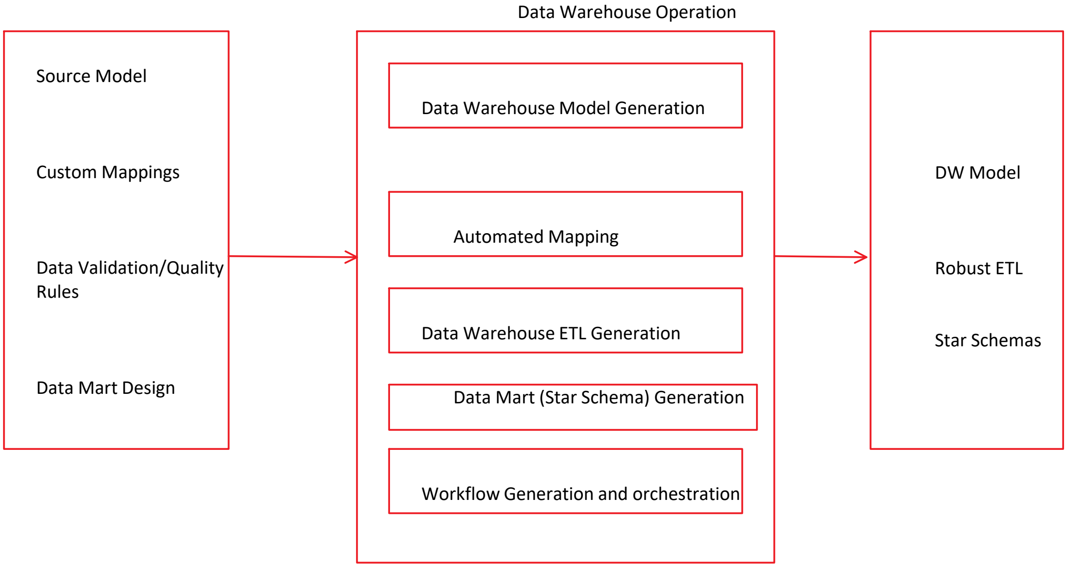
08 September 2022 18:47



Benefits:

a) Massive volumes of structured and unstructured data like ERP transactions and weblogs can be stored in cost effective manner
b) Data is available for use for faster transactions by keeping it in a raw state
c) A broader range of data can be analysed in new ways to gain unexpected and previously unavailable insights.

Data Warehouse


Data Warehouse Operation
Source Model
Custom Mappings
Data Validation/Quality Rules
Data Mart Design
Data Warehouse Model Generation
Automated Mapping
Data Warehouse ETL Generation
Data Mart (Star Schema) Generation
Workflow Generation and orchestration
DW Model
Robust ETL
Star Schemas

Benefits :

a) Little or no data prep is required, making it easier for us to do analysis and business users to access and analyse the data
b) Accurate, complete data which can be available more quickly, so that business can turn the information into insight faster
c) Unified, harmonized data offers a single source of truth, building out the data insights and decision making across the business lines.

	Data Lake	Data Warehouse
Data Storage	A Data Lake contains all of an organization's data in a raw, Unstructured format and can store the data for indefinite time Even accessible immediate or future use	A data warehouse contains the structured data such that it can be cleaned and processed, ready for strategic analysis based on predefined business needs.
Users	Data from a data lake, with its large volume of unstructured data can be used by data engineers and data scientists who prefer to study data in its	Data from a data warehouse is typically accessed by managers, business stakeholders looking for the quick insights from the business KPI, and as the data has been structured to

	raw form to gain new and unique business insights	provide answers to the pre-determined queries for analysis
Schema	Schema is defined after the data is stored in the data lake, unlike data warehouse making the process of capturing and storing the data faster	In DW, the schema is defined before the data is stored, the lengthens the time it takes to process the data, but once it's completed, the data which made in DW, is ready as consistent, confident to use across the org
Processing	ELT (Extract, Load, Transform) , in this process, the data is extracted from the source for storage in the data lake, and its structured only while required	ETL (Extract, Transform, Load), in this process, data is extracted from its sources, scrubbed, parsed, then make it structured for business -end analysis
Cost	Storage costs are fairly cheaper in a data lake, also less time consuming to manage which reduces the operational costs	Data warehouses cost more than the data lakes, and required more time to manage, resulting in additional operational costs.

ADLS Gen2

1. Hierarchical namespace - it organizes the objects/files into the hierarchy of directories for efficient data access. There's a common object store naming convention used with slashes in the name to define the hierarchy directory structure.

This hierarchy structure becomes real with ADLS Gen2, operations such as renaming or deleting of a directory, there's no need to enumerate and process all objects which share the name prefix of the directory.

a) Performance wise improves the directory management operations, which overall makes efficient the job performance
b) Management wise earlier we can organize, manipulate the files through directories and subdirectories
- c) Security is enforceable we can define POSIX permissions on the directories and individual files.
d) ABFS driver is used to access the data in the ADLS gen2, it's available within all Apache Hadoop environment. The env includes Azure HDI, Azure DataBricks and Azure Synapse Analytics.

Azure Blobs	Container	Virtual Directory	Blob	
ADLS Gen2	Container	Directory	File	

.Root
/ Child
/folders
/file1, file2… fileN

Difference between ADLS Gen1 And ADLS Gen2

Difference	ADLS Gen1	ADLS Gen2
Account Root Permissions	Permission required on Account - root - RX (minimum) read-only/ read-execute or RWX (read-write-execute) to get an account root content view	An user with or without permission on container root can view the account root content
RBAC roles and access control	All users in RBAC owner role are superusers. All other users (non-superusers) need to have permission which should abide by the file folder level ACL.	All users in RBAC-Storage blob data owner role are superusers. Rest of other users can be provided with different roles (contributor, reader) etc which can govern their read, write and delete permission
Store default permission	Permission for an item (file/directory) cant be inherited from the parent directory to child.	File store permissions can be inherited if the default permission is set, on the parent directory/items before the child directory/items are being created. The file store permission can be inherited from parent to child directory/folder..
Nested file or nested directory creation for non-owner user	Check whether write-execute (wx) permission for owner is imposed in the sub directory	Does not add wx permissions in the subdirectory
User provided permission	When a file/directory is created , file/directory is created, the final permission will be same as the user provided permission	File/directory is created , the final permissions will be calculated based on the [user provided permission + umask) value.
UMask	Clients can apply umask on the permission on new file/directory before request sent	Clients can provide umask as the requested query params during the file and directory creation. The default umask will be applied 027 on the file/directory level.

Access Control Permissions set up for ADLS gen2
RWX -- read + write + execute
R-X -- read + execute
R-W - read + write
R-- Read
--- no permissions

ADLS umask

For a ADLS Gen2 container, the mask which is applied for ACL level of the root directory, ("/") defaults to the value of 750 for directories, the value of 640 for files.

	Directories	Files
Owning user	rwx	Rw--
Owning group	r-x	r--
Other	---	…

Lambda Architecture

1. Batch processing Layer - ADLS Gen2, Azure Blob storage, HDInsight Hive activity, pig activity
2. Speed data processing layer - Azure event hub, stream analytics, Spark streaming
3. Serving layer - Analytics data store as well as includes the Azure Analysis service, BI and analytics layer.

1. Model --> on data access layer is built
2. View --> (.cshtml, .vbhtml)
3. Controller ---> takes all the data models fetched from the asp.net mvc model & defines the logic (Itemcontroller, HomeController)

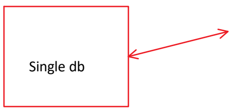
Azure SQL database

08 September 2022 18:47

TCO = Total cost of ownership

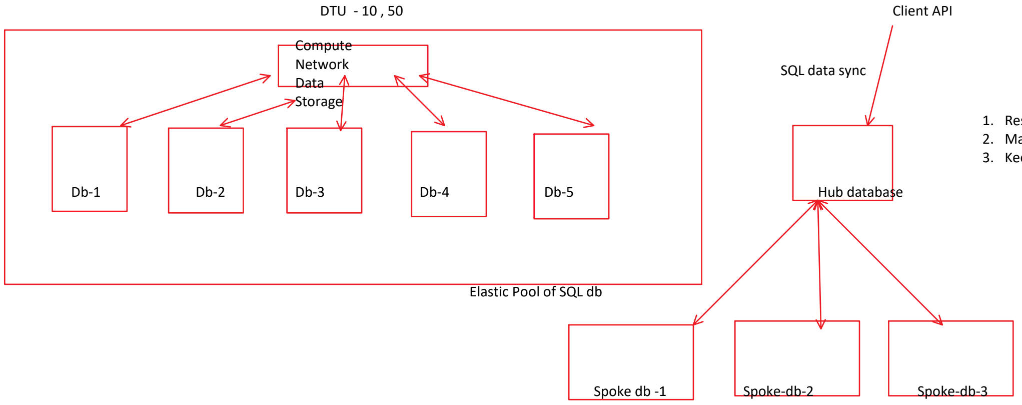
Total cost of ownership is referred where the cost of cloud/ Azure resources can be optimized according to its underlying infra, network, storage, backup, high availability features. Also, it's defined the options of how the cost of the specified cloud resources can be reduced without bringing down the core functionality/ business requirements.

SQL Server on Azure VM(IaaS)	Azure SQL Managed Instance (PaaS)	Azure SQL database (PaaS)
Simple migration of application where the core SQL Server on-prem features are required. (e.g. .net framework runtime, CLR, SQL Server Agent, SQL Server Broker, Log shipping) etc.	Lift - shift of database migration From on-premise to Azure where We can get all of the core SQL db benefits Over cloud , without managing the underlying compute, storage, network, disk	Purely managed SQL database offering where no licensing required, no underlying administration of server, SQL database, network is required. End to end database migration is feasible with just a click focusing into the core sql development features.
License is required, either through BYOL, license has to be procedure	No license required, entire SQL db on-prem feature can be availed with the managed platform service	No license is required, only SQL db dev specific features are available, there's a limitation of db sizes are there (max 5 TB -> 100 TB)
SSAS, SSRS, SSIS, SQL Server broker, .net framework runtime, CLR integration, distribution transactions, database mail etc. all features are supported.	.net framework, functions, distributed transactions, ACID, automated backups, HA are also supported along with no administration required	No support for .net framework runtime, distributed transactions, ACID, CLR related functions, stored procedures based on Windows runtime etc. are not supported.
SLA - 99.9% Automated backup, Azure site recovery for disk level backup of database	SLA - 99.99% , 99.95% Automated backup, point-in time restore, geo-replication, high availability, automated patching	SLA - 99.99% Automated backups, active-geo replication, high availability & DR, replication on both AZ and regional level.



1. Compute
2. Network
3. Storage
4. Data

Core infrastructure compute unit allocated for Azure SQL db - DTU
Core infrastructure compute unit allocated for Azure SQL elastic db - eDTU



1. Responsible for managing the centralized db operations
2. Managing all the database synchronization
3. Keep all the spoke db instances in sync

Azure Storage usage scenarios

Blob storage - store all kinds of object based stores, like media files, documents, large files
File share - store all kinds of filesystems, on-premise file share can be migrated to azure file share
Queue storage - messages coming from messaging platform - ActiveMQ, Kafka,
Table Storage - non-relational dataset that not compatible with Codd's rules
Disk storage - disks in form of virtual hard disks can be stored in Azure Storage and can be re-utilized for usage in other VMS

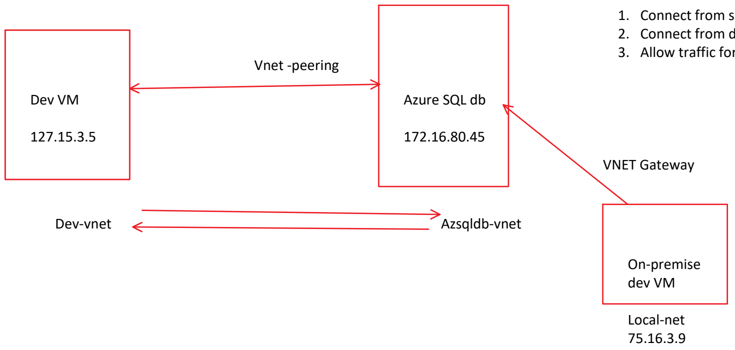
Lab 1 and Lab 2 :

1. Provision an Azure SQL elastic pool db
2. Connect to the database from SSMS / ADS
3. Create Azure SQL db using CLI
4. Create Azure SQL db on VM
5. Migrate on-premise SQL db to Azure SQL db using Azure Database migration assistant tool (Azure Data Studio)

Data Migration from on-premise SQL server db to Azure SQL db

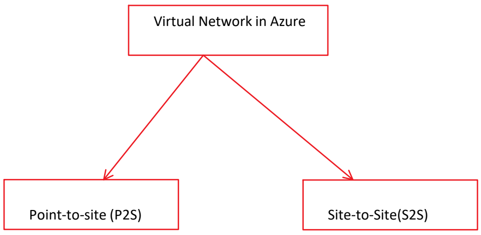
On-prem	Azure SQL db	Azure SQL Managed Instance	SQL Server on Azure VM
adventureworks	Data Migration Tool	Azure SQL extension of Azure Data Studio	Azure SQL extension of Azure Data Studio
adventureworks	SQL data import-export wizard (SSMS)	Data Migration Tool	Data Migration Tool

Key - AES 256 bit (Customer-managed key)
SHA 128/256 bit



1. Connect from source vnet to destination vnet
2. Connect from destination vnet to source vnet
3. Allow traffic forwarding capability and allow remote gateway

Express Route - a Managed service for connecting your on-premise network to Azure Virtual Network (in AWS Direct Connect)



1. Multiple users part of same domain of the corpnet tries to access the bunch of Azure resources on Azure Vnet.

1. A single user is connected over Versatile resources on Azure Vnet

1. Principle of least privilege - to provide the minimal permission to the user based on their accessibility level and to do respective work on Azure

Encryption at Rest	Azure SQL db	Azure SQL managed instance
Transparent data encryption (TDE)	Real-time encryption and decryption of the db, backups, transaction log files	Real-time encryption and decryption of the db, backups, transaction log files
	Enabled by default for encryption of data at rest	Enabled by default for encryption of data at rest
	During encryption , the TDE encrypts the storage of the entire database by using a symmetric key called the db encryption key (DEK). During the db startup time, the encrypted DES is decrypted and used for decryption and re-encryption of the db files in the sql db engine.	During encryption , the TDE encrypts the storage of the entire database by using a symmetric key called the db encryption key (DEK). During the db startup time, the encrypted DES is decrypted and used for decryption and re-encryption of the db files in the sql db engine.
	During the usage of AKV, for customer managed key - TDE protector is used to protect the data encryption key. Is stored as asymmetric key in key vault	During the usage of AKV, for customer managed key - TDE protector is used to protect the data encryption key. Is stored as asymmetric key in key vault
Encryption in Transit	Enabled through TLS 1.2 protocol	Enabled through TLS 1.2 protocol

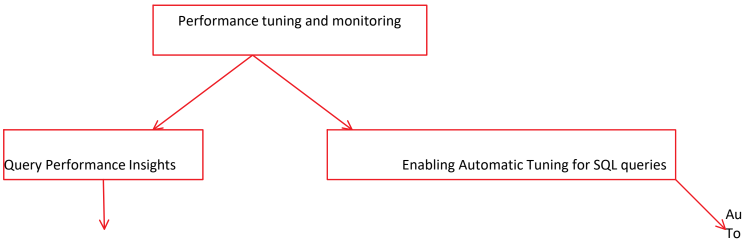
1. Create and configure Azure SQL db over private link

a) Existed VNET/ Create a new VNET and allow bastion host
b) Create a VM
c) Azure SQL server to be enabled over the private endpoint
d) Test the connectivity to the SQL server private endpoint

Performance tuning and recommendations

1. Create index over Azure SQL database tables (clustered or non-clustered)
2. Apply Schema
3. Automatic tuning for Azure SQL db tables

Performance tuning and monitoring



Performance tuning and monitoring

Query Performance Insights

Enabling Automatic Tuning for SQL queries

Automatically manage the data service
To continuously monitor the queries and identifies the action which we can
Use to optimize the Performance of workload.
We can review recommendations and manually could apply them
which is known as automatic tuning mode.

It provides query analysis for single or pooled databases which helps to identify the Top resource consuming and long-running queries in the workload.

- Deeper insight s into the database (DTU) consumption
- Details on top database queries by CPU, duration, and execution count
- The ability to drill down into the details of a query, to view the query text and the history of resource utilization.
- Azure advisors for database recommendation

MAXDOP - the maximum degree of parallelism (MAXDOP) is a server configuration option for running SQL Server on multiple CPU. It controls the number of processors used to run a single statement in parallel plan execution.

The default value for MAXDOP is 0 which enables the SQL server to use all possible processors.

In Azure SQL db, we can check and configure the dmv named as sys.database_scoped_configurations system catalog view.

```
Select [value] as currentMaxDop from sys.database_scoped_configurations
WHERE [name] = 'MAXDOP';
```

Azure Blob Storage

08 September 2022 18:48

Scenario	Azure Storage Solution
Massively scalable and secure objects for storing into cloud and utilize for cloud-native workloads, archives, data lakes, high performance computing, and to store massive volumes of data In the form of objects	Azure Blob storage
Massively scalable and secure data lake solution for the high performance analytical workloads which requires the support for both blob and file storage together and can provide the granular access control and policy	Azure Data Lake Storage
Simple, secure and serverless enterprise-grade cloud based file share, migrate on-premise file system or connect on-premise file share with Azure	Azure File storage
Store high performance real time streaming messages in publish-subscribe mechanism with fault tolerance and scalability support	Azure Queue storage
Store the random dataset non-compliant to RDBMS standard (no primary key/unique key, no consistency and redundancy), non-compliant to Codd's rule. The storage requires the proper data partitioning for consistency	Azure table storage
High performance, durable block storage in the scenario of business-critical apps	Azure Disk Storage
Synchronization of on-prem file share with Azure file share with caching support for on-prem data as a hybrid cloud file share	Azure File Sync
Appliances and solutions for transferring data into and out of Azure quickly and cost-effectively	Azure Data Box (batch processing and real time streaming data)

Types of Storage Account	Supported Storage services	Redundancy Options	Real time usage
Storage Account Standard tier (GPv2)	Blob storage (ADLS Gen2), Azure Queue storage, Azure Table Storage, Azure File share	LRS GRS RA-GRS (read access geo redundant storage) ZRS (zone redundant storage) GZRS RA-GZRS	Standard storage account type for blobs, file shares, queue and tables. Recommended for most of scenarios of data storage in Azure. For support of NFS, in Azure Files, the premium file share account
Premium Block blobs`	Blob storage(ADLS Gen2)	LRS ZRS	Premium storage account types supported are the block blobs and the append blobs. - High transaction rates of objects/data requires to be stored - Low latency is expected - data is stored in SSD, optimized for low latency, high throughput
Premium file share	Azure Files	LRS ZRS	Premium file share works for file share only, recommended for enterprise or high performance scale applications required file storage support. - Support for both NFS and SMB file shares - File transfer is faster because data stored in accessible memory chips (RAM)
Premium Page Blobs	Page blobs only	LRS	Premium storage account type for page blob only

Interactive analytics
Iot/streaming analytics
AI/ML

Azure Storage Lifecycle management provides rule based policy which can use to transition blob data to the appropriate access tier to even expire the data at the end of the data lifecycle.

- Automatically transition blobs from cool to hot tier immediately when they're required to be accessed to optimize the performance
- Transition current versions of a blob, previous versions of a blob, blob snapshots to a cooler storage tier (if it's not accessed/modified, to optimize the cost), the lifecycle management policy, the objects can be moved from hot to cool, hot to archive tier or from cool to archive tier.
- Delete the current versions of the blob, previous versions of the blob, blob snapshots at the end of the lifecycle (automation tool)
- Define the rules which can executed at the storage account level
- Apply rules to containers , to blob level using the blob name prefixes or index tags as filters.
- Helps to design best cost effective storage solution as per business requirement by automatically

↓ Cost

- Hot
- Cool
- archive

↑ Access Time

moving the storage in appropriate tiers

- Block blobs, append blobs, storage account gpv2, ADLS gen2
- Collection of json rules

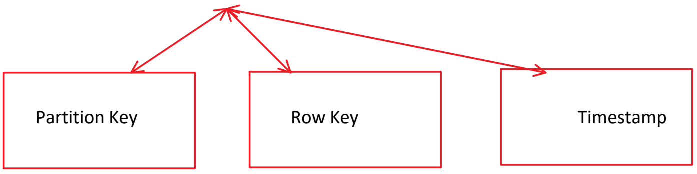
-- Connect Azure File share to Windows / Linux

a) Deploy a VM
b) Connect to VM
c) Mount the azure file share to the VM
d) Create and delete a share snapshot

Supports for SMB protocol with gpv2 and LRS/ZRS redundancy tier

Azure Table Storage Concepts

a) Azure Storage
b) Azure Table
c) Entity -> Property



Partition Key + Row Key + Timestamp = Primary Key of Azure Table storage

An entity can include upto 252 properties to store data
Each entity has three system properties which specifies a Partition key, a row key and a timestamp.

- Entities within the same partition key can queried more
- Efficiently, inserted/updated data in atomic operations.
- An entity's row key is unique identifier within a partition

Row -> entities
Columns -> properties

Partition Key

a) Tables are partitioned to support load balancing across the storage nodes,
b) A table's entities are organized by partition.
c) A partition is a unique identifier of the particular partition within a given table as specified in the PartitionKey property
d) The Partition key forms the first part of an entities primary key which can be size of upto 1 KB.
e) Any CRUD operations, for Azure table storage we need to include the PartitionKey property .

Row Key

a) Unique identifer for an entity within a given partition
b) Together partitionkey and rowkey uniquely identify the every entity within a table
c) The row key is a string value upto 1 KB of size
d) We can include the RowKey property in every CRUD operations

TimeStamp Property

a) The Timestamp property is a DateTime value maintained on the server side to record the time of entity last modified.
b) The table service includes the Timestamp property to provide optimistic concurrency

Entity Group Transaction

- All entities subject to operations as part of transactions must have the same PartitionKey value
- An entity can appear only once in the transaction, and only one operations performed against it
- The transaction can include at most of 100 entities, and its total payload may not more than 4 MB in sizes
- All entities are subjected to limitation like based on semantics - a change set on a group entities performing CRUD
- A batch operation performing create entity, merge entity, update entity, delete entity etc. includes one or more change sets and query operations.

Azure Analysis Service

08 September 2022 18:48

Tabular Model (SSAS)

Tabular model of SQL Server Analysis service was introduced with SQL Server 2012. Tabular model in SQL Server uses a different engine (xVelocity), it's designed to be build queries faster based on columns, because it uses the columnar storage in additional to better data compression.

In Tabular Model, the data is stored in memory, it's very important to have a lot of memory to be given in your server and for faster processing of queries as very fast CPUs.

The disks are not as important on tabular model.

Multidimensional model (SSAS)
The multidimensional db is very different structure than a relational db and allows us to generate reports very fast. The Multidimensional model was the only solution used in the past to create multidimensional databases.
The multidimensional model are supported in SSAS. It's useful to build OLAP modelling constructs (cubes, dimensions and measures). Multidimensional model also supports the import of data from external sources.
The amount and type of data that required to be imported can be a primary consideration when deciding which model best fits for your data.

Feature	Tabular	Multidimensional
Hardware	It's very important to clarify, the hardware used for multi-dimensional model cant be used for tabular model. Tabular model is a memory dependent solution. The tabular model is optimized with more memory, the better performance with the more memory. Without sufficient RAM, tabular model will fail CPU speed is also very important for tabular db,	For Multi-dimensional scenario, where the database requires a lot of disk space like >5 TB, there have to built multidimensional model
Advantage	Tabular model is faster for some queries and compresses the data even more than the multidimensional solutions.	Multi-dimensional model is not that faster
Data Import	Tabular Model can import data from relational data sources, data feeds, some document formats. We can also use OLE DB and ODBC providers for building tabular model.	Multidimensional solutions can import data from relational data sources using OLE DB native and managed drivers
Data sources	SQL Server relational db, Access databases, Oracle db, Teradata, Sybase, DB2	Most of similar data sources through OLE db provider, .net framework data provider, TOLEDB ole db providers etc.
Query	DAX calculations, DAX queries, MDX queries.	Supports the MDX calculations, MDX queries and DAX queries, ASSL.
PowerShell	Analysis Services powershell is supported for both tabular and multidimensional model and db	

a) For Tabular model, the in-memory models are default.

DirectQuery is an alternative query method for models which are too large to fit
Into the memory, or when the data includes a reasonable processing strategy

b) AAS is built on top of SSAS, is an instance of SSAS configured for Tabular server model
c) Deployment of tabular models can be managed in SSMS or by using different tools like excel / PBI

1. Integrated workspace - default working database is created in-memory using VS own implicit instance. Integrated workspace also significantly reduces the complexity of authoring tabular projects because of an explicit server is not required.

2. Workspace server - specify the workspace property, resides as localhost or local named instance of an SSAS server, we can even use remote instance to host the workspace database.
   Cons

- There's a potential latency
- The data backup cant be set to backup to disk
- Cant import data from an excel workbook

Evaluate 'table name'

Any expression is to be evaluated for each row of the result as long as is valid.
Order by '<table name> <column name>'

START AT keyword is used inside an ORDER BY clause which defines the value at which the query result has begun,

Azure Analysis Service is a fully managed platform as service (PaaS) that provides the enterprise-grade data models in the cloud. Use advanced mashup and modelling features to combine the data from multiple data sources, define the metrics, secure the data in a single, trusted tabular semantic data model. The data model provides an easier and faster way for users to perform ad hoc data analysis using the PowerBI and excel

- Developer

(Recommended for evaluation, dev and testing purpose, a single plan includes the same functionality of the standard tier, but it's limited in processing power, Query processing unit (QPU), and memory size)

-- Limitation - Query replica scale out is not available in this tier. This tier does not offer an SLA.

- Basic

This tier is recommended for production, solutions with smaller tabular models, limited user concurrency, and simple data refresh requirements. Query replica scale-out is not available in this basic tier.
Perspectives, partitions and Directquery is not supported

- Standard

  The tier is recommended for business critical and mission critical purpose, for applications that require elastic user-concurrency, and have rapidly growing data models. It supports the advanced data refresh for near real time data model updates, and supports all tabular model features.

  Scale out (Horizontal scalable) resources for faster query response

  With the scale-out approach, client queries are distributed amount multiple query replicas in a query pool, query replicas have synchronized copies of the tabular models. By spreading the query workload, response times during the high query workloads can be reduced. Model processing operations can be separated from the query pool, ensuring the client queries are not adversely affected by query processing.

  We can create the query pool with upto seven query pool replica, the number of query replica that can include in the pool depends on the choosen plan and region. Query replicas cant be spread outside of the server's region, whereas the query replicas are billed as the same rate of the server.

Core Features of Tabular Model

a) Tabular models in both in-memory and DirectQuery modes are supported. In-memory mode tabular model support for multiple data sources, because this model data is highly compressed and cached in-memory. This mode provides the fastest query response over large amounts of data, it also provides the greatest flexibility for complex datasets and queries.

b) Partitioning enables incremental loads, increases parallelization, and reduces memory consumption, other advanced data modelling features like calculated tables, columns, measures and all DAX functions are supported.
c) In-memory models must be refreshed or processed to update cached data from data sources.
d) Tabular model also supports DirectQuery mode, it leverages the relational db for storage and query execution. Extremely large datasets like SQL server, SQL server DW, Azure SQL db, Azure Synapse Analytics and Oracle, Teradata and various data sources are supported.
e) Complex data models refresh scenarios in AAS tabular model directquery mode is not required. Exceptions like limited data source types, DAX formula limitations or unsupported features

Azure Analysis Services - Server provisioning

a) Azure AD - create an user dedicated for AAS (global admin privilege)
b) Azure AD - create a group dedicated for AAS (user to be part of this group)

c) Create AAS service
d) Add the Azure AD user / additional Admins
e) Firewall access
f) Connect with SSMS
g) Configure the server admins and user role
h) Connect with PBI desktop from AAS tabular db

Request your subscription owner to give permission for Contributor access on Azure Subscription
OR
Request for Azure AD access as contributor level

Tenant Root Group (Admin)

Management Group (dept, costcenter, purpose)

Subscription (Default Directory) (your Azure account + policy + permissions)

a)
Airline Analytics Team:

1. With AAS, we cant build technically and deploy cube, we need SSAS. In SSAS, we can deploy the cube which is not part of our actual curriculum.
2. For this issue, the data load problem in AAS cube, we cant move into DAX queries
3. Even to Tabular model building in SSMS using hierarchies, dimensions.

Can it be checked with the actual technical topics as part of our curriculum?

Project Document

To be prepared by each and every member of the team as per contribution

Items to be included in the project document:

a) Problem statement
b) Your solutions prescribed on the problems which are trying to accomplish
c) High level design
d) Low level design
e) Tools and resources used
f) Code glimpse running on SSMS
g) PowerBI dashboard visuals and connectivity details

During presentation
1. Presentation deck in relates to Case study
2. Demo
3. Project document

DAX in Tabular Models

Data analysis expressions is a formula language used to create custom calculations in AAS, PBI, Powerpivot. DAX formulas include functions, operators, values to perform advanced calculations on the data in tables and columns

- **Calculated columns** - is a column which you can add to an existing table and then create a DAX formula which defines the column's values.

- **Calculated tables**- computed objects based on the DAX query or expressions and it can be derived from a full part of others tables in the same model

- **Measures** - Measures are the dynamic formulas where the results can change depending on the context. Measures are used in reporting formats which support combining and filtering model data by using multiple attributes such as PBI report, excel powerpivot or pbi dashboard.

  A formula in a measure can use to implement standard aggregation like COUNT, SUM or even your custom formula in DAX

- **Row filters** - defines which rows in a table should be visible to the members of a particular role. It is to be evaluated which rows can be returned by the results of a query by members of a particular role. In DAX, the row filters are evaluated as true/false.

Objective for this demo - Tabular model

1. Create of a new tabular model project
2. How to import data from SQL Server db into the tabular model project
3. Mark a table as Date table
a) Rename a column of dimDate table
b) Set the Mark as Date table

4. Create and manage relationships between the tables in the model
5. Create and manage the calculated columns, measures
6. KPIs for analysis
7. perspectives to help users to browse the data model
8. Create the hierarchies and
9. Create the partitions dividing the table data into smaller logical parts.
10. Security model objects
11. Deploy a tabular model to an AAS.

Pre-requisites

a) Visual Studio 2019, 2017 community edition
b) (VS 2022) and compatible SQL Server edition - 2019 or higher
c) SSMS
d) PBI desktop
e) AdventureWorksDW
f) An Azure Analysis Services instance, SSAS

a) AAS instance
b) Azure SQL Server data warehouse /
c) download the AdventureWorksDW sample database for on-premise SSAS instance. (Azure VM for SQL) on-prem data gateway is only for local server/system and local VM.
d) Analysis Services project extension for VS

Steps:

a) Provision Azure VM
b) Provision SQL Server 2017 with SSDT
c) Install SSMS and PBI desktop
d) Download the sample AdventureWorks DW db

Visual Studio Community Edition - 2017, 2019

Measure can be created by using the Measure name, followed by a colon, followed by the formula expression.

DaysCurrentQuarterToDate:=COUNTROWS(DATESQTD('DimDate'[Date]))]

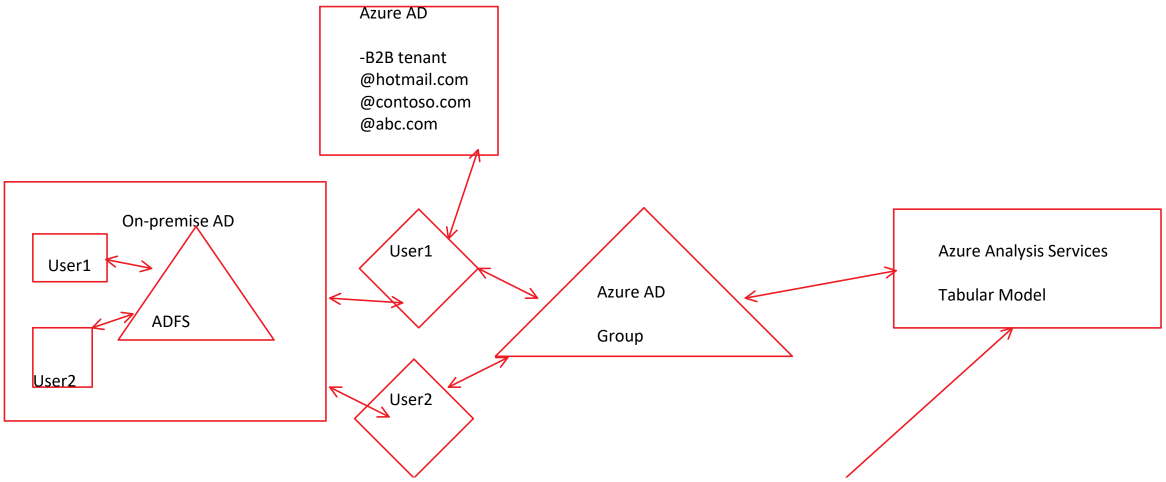
Custom measures in SSAS / AAS

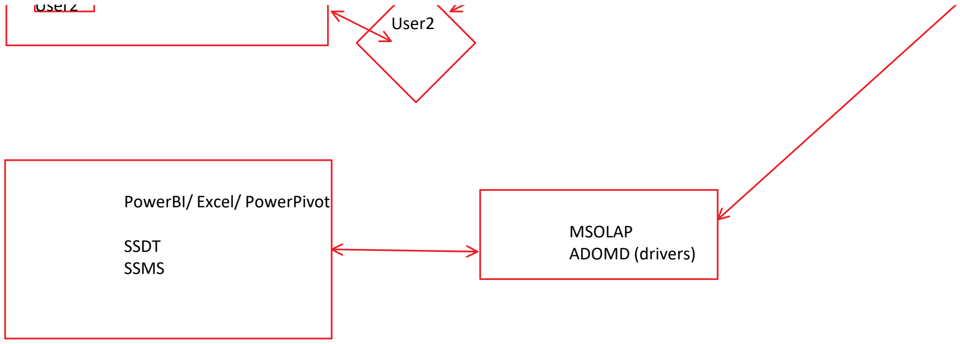
a) Built on measures being created on the columns of the table
b) Perform complex calculation on the previously implemented measures in the custom measure

Year wise analysis of a team --

a) Total count of runs for a team in a month
b) Total count of runs for a team in 12 months
c) Max
d) AVG

Authentication and user permission





AAS Authentication

a) MSOLAP
b) ADOMD
c) AMO

- All of these three client libraries has got the support for both Azure AD interactive flow and non-interactive authentication methods. The two non-interactive methods, like AD password and AD integrated authentication methods can be used for apps using MSOLAP or ADOMD client libraries.
- The traditional BI tools like PBI desktop, VS, SSMS support the Active Directory universal authentication, and interactive method which has got support for Azure AD MFA (multi-factor authentication).

a) MFA authentication can happen through Microsoft Authenticator app, google authenticator app, hardware tokens, text or verification phone call on the registered number

d) Azure Analysis services through Windows authentication, universal authentication is not supported. For scenario, connecting to Excel from AAS, the ADFS is required, universal authentication is not required.

e) Through SSMS, AAS supports for both interactive and non-interactive authentication methods.
f) Allows Azure B2B users invited to the Azure AS tenant. When connecting to a server, guest users must select AD universal authentication when connecting to the server.

User Permission in AAS

- **Server Administrators**

The server administrators are specific to an AAS server instance. They connect with tools like portal, SSMS, VS to perform the tasks like config settings and managing the user roles.

The user role can also creates the server which is automatically added as Analysis Services server administrator.

- **Database users**

The db users can connect to model databases by using client applications like excel or PBI. Users must be added to database roles, database roles can define administrator , process or read permissions for a database.

It's important to understand the database users in a role with admin permissions different than the server administrators.

By default, the server administrators are also the database administrators

- **Azure resource owners**

Resource owners manage the resources for an Azure subscription. Resource owners can add Azure AD user identities to Owner or Contributor roles within a subscription by using Access Control in portal, or with ARM templates.

- **Database Roles**

Roles are defined for a tabular model are database roles. This role contains members consisting of Azure AD users and Security groups which have specific permissions that define the actions those members can take on a model database.

A database role is created as a separate object in the database, applies to only the database in which role is created..

By default, while creating a new tabular model object, the model project does not have any roles. Roles can be defined by using Role manager dialog box in VS, When roles are defined during the model project design, they're applied only to the model workspace db.

When the model is deployed, the same role are applied to the deployed model. After the model has been deployed, server and db administrators can manage roles and members by using SSMS.

Client Libraries available in AAS

a) MSOLAP (v15) (Analysis Services library for OLE DB provider)

Native client library for Analysis Services db connections, used indirectly by both ADOMD.net and

AMO, delegate the connection requests to the data provider. We can also call the OLE db provider directly from the app code.

The AAS OLE db provider is installed automatically by most tools and client apps used to access analysis services db. It must be installed on local system used to access the AAS data.

b) AMO (v19.12)

Managed client library used for server administration and data definition.

SSMS uses the AMO to connect to Analysis Services,

c) ADOMD (v19.12) - available for both .net framework and .net core (cross-platform & light-weight)

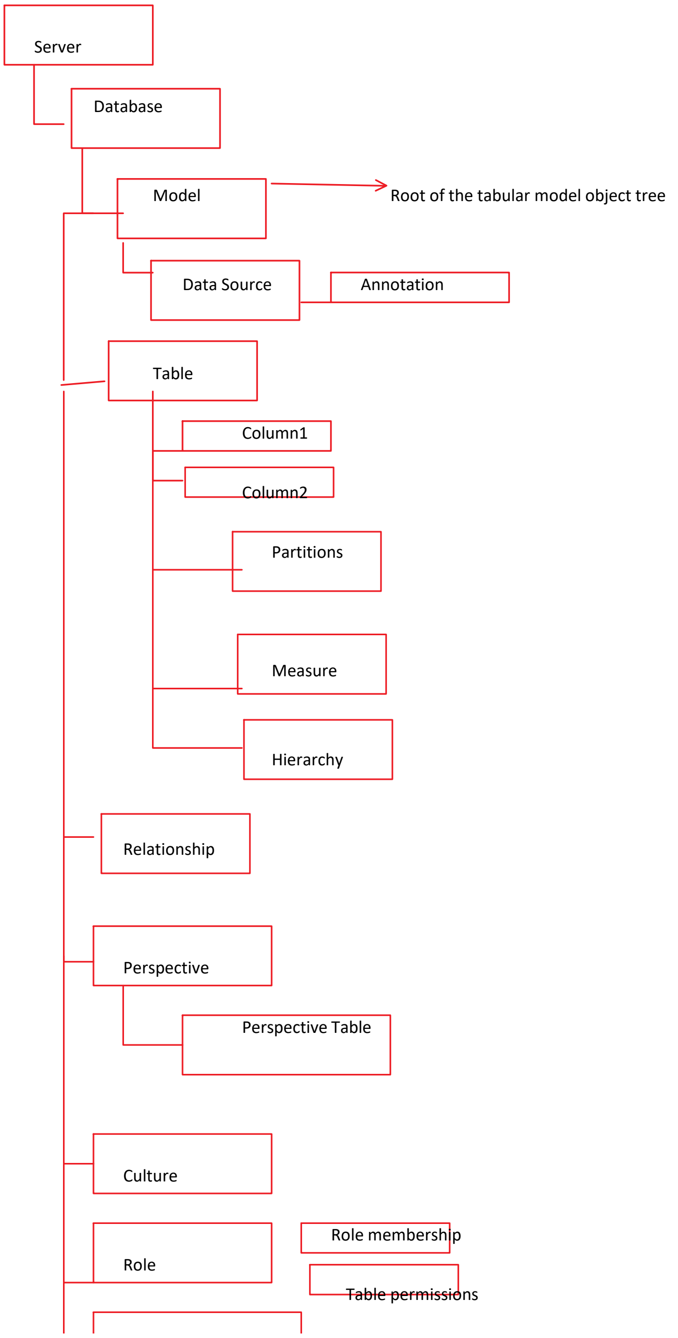
ADOMD.net is a managed data client library used for querying Analysis services data. It's installed and used by tools and client apps.

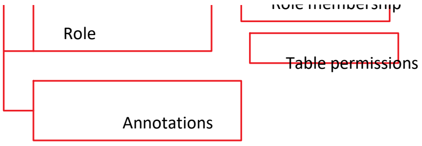
Tabular Object Model (TOM)

Its an extension of the Analysis Management Object (AMO) client library, created to support programming scenarios for tabular models created at the compatibility level 1200 and higher.

As with AMO, TOM provides a programmatic way to handle administrative functions like creating models, importing and refreshing data and assigning roles and permissions.

- TOM exposes the native tabular metadata like model, tables, columns and relationships objects. A high level view of this object contains the tabular objects like "Microsoft.AnalysisServices.Tabular.dll" assembly.





a) Create Service Principal in Azure AD
   - Pre-requisite - Register an application with Azure AD
   - Assign role to the app
   - Once the service principal got created, we can assign permission to AAS server admins through Service Principal

It is the application registered in Azure AD which helps for a particular user to sign in and authenticate with the Azure directory and to be able to provision the Azure resources.

Components of Service Principal (during authentication from on-prem resources to Azure) have to provide these credentials

a) Application (client) ID
b) Tenant ID (Directory)
c) Subscription ID
d) Client Secret (password)

In real -time enterprise scenario, we give permission for Azure resources accessibility (get, put, list, delete etc.) to Service principles.

Benefit of Service Principle

Service principles can hold multiple users identity and roles. Hence, we don't have to provide access permissions to multiple users individually, we can directly provide the access permission the respective service principle/app for creating and managing a certain Azure resource.

KPIs

a) Create a KPI named InternetCurrentQuarterSalesPerformance

Definition of KPI

b) A KPI is used a tabular model of AAS and SSAS services. A KPI is used to gauge the performance of a value, defined by a base measure, against a target value, also defined by a measure or by an absolute value.

Components of KPI

a) Base value (measure) - defines the measure which resolves to a value
b) Target Value - A target value is called by a value to define maximum threshold value.
c) Status threshold - defines the a range of values between low, medium and high threshold values.

Benefits of KPI

1. In BI, the KPI is used to define a quantifiable measure of the evaluation for business performance.
2. A KPI is evaluated frequently over time to time basis to understand the performance of Sales, marketing, profit margin etc.

Perspectives

A Perspective defines a viewable subset of a model which provides focused, business-oriented or application specific viewpoints. When a user connects to a model by using perspective, they can see only those model objects(tables, columns, measures, hierarchies and KPIs) as fields being defined in the perspective.

InternetSales perspective which will exclude the DimCustomer table object. When you create a perspective which excludes certain objects from view, the object still exists in the model. Calculated columns and measures are either included in the perspective or not can still calculate from object data which is excluded.

Hierarchies

Hierarchies are groups of columns arranged in levels.

A geography hierarchy might have sub-levels for country, state, locality, country, city, address

Hierarchies can appear separated from other columns in a reporting app list.

1. Create Category hierarchy in DimProduct table

category ->
    name
    subcategoryname
    model

product

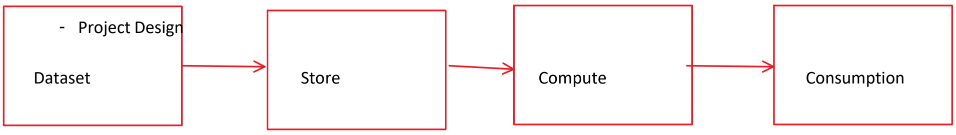
Data Source	Deployment Options	Applicable for	
On-premise (local, SQL db on VM etc.)	Configure SSAS & deploy on SSAS server	SSAS	
Migrate your db to Azure SQL db /SQL DW	Deploy it over AAS	AAS	

1. Migrate the SQL Server db to Azure SQL db using Data migration assistant (schema + data)
2. Write the SQL queries
3. Show up the output in PowerBI

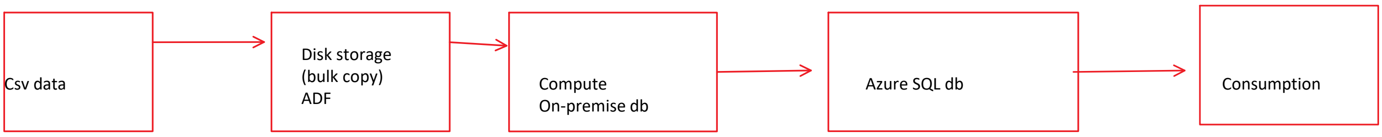
Components in Project doc

- Project headline
- Introduction:
- Problem Statement:
- Pre-requisites:
- Project Design

HLD



LLD



Migration

Data migration assistant

- Implementation Steps:

1)

- Sample queries and Screenshots:

- Conclusion
- Appendix

Sprint 1 Presentation

- Presentation Deck (3-5 mins)
- Project Document (10-15 mins)
- Hands-on demo (10-15 mins)

Azure Synapse Analytics

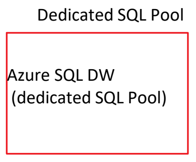
08 September 2022 18:48

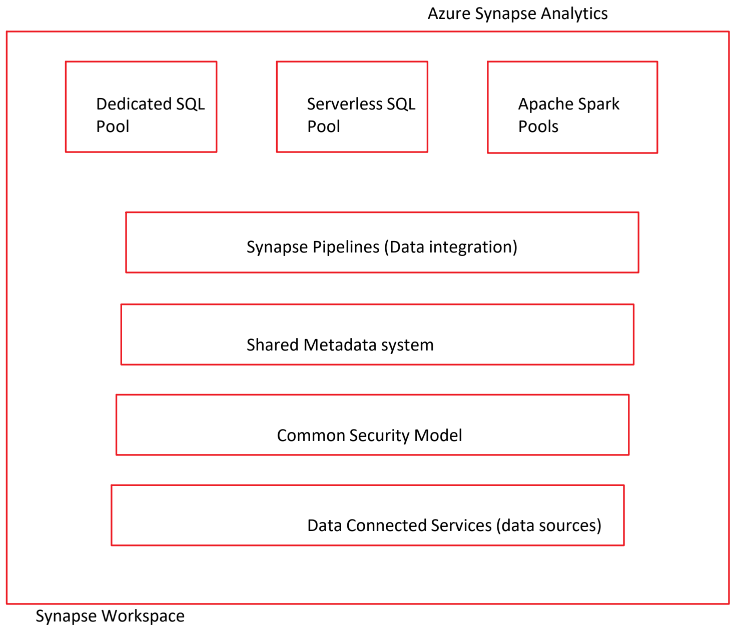
Type of Analytics	Pattern
Descriptive Analytics	Tells you what happened in the past (e.g. last six month's sales demographics)
Diagnostic Analytics	Helps to understand why something happened in the past (e.g. performing the actual root-cause analysis of the data, Why the customer churn happen about 60% over last three months) data discovery, data mining and data drill through
Predictive Analytics	Predicts what is most likely going to happen in the future (e.g. what is % of chance the customer will go to buy the product next, Building out recommendation engine to provide choices to customer for next available products/solutions to buy
Prescriptive Analytics	Recommends actions which you can take based on this outcomes (e.g. end to end product recommendation for the customer
Cognitive Analytics	Modern AI solutions encompassing the benefits towards customized vision, speech services, text services, chatbot services with advanced ML and AI knowledge.

Azure Synapse Analytics (Dedicated SQL DW)

Azure Synapse Analytics is an analytics service which brings together enterprise data warehousing and Big Data Analytics platform. Dedicated SQL Pools (earlier SQL DW) refers to the enterprise data warehousing features available on Azure Synapse Analytics.

1. Azure Synapse (dedicated SQL pool) represents a collection of analytics resources which are provisioned using Synapse SQL. The size of a dedicated SQL pool is determined by the data warehouse units (DWU).
2. Once the dedicated SQL pool is provisioned, we can import big data with simple Polybase sql queries. Then we can use the power of SQL DW query engine to run the high performance analytics.





Benefits of Dedicated SQL Pool

1. Dedicated SQL Pool uses a node-based architecture, applications can connect and issue t-sql commands to a control node.
2. The Control node is the central node(brain) of the DW, hosts the distributed queries in the query engine. It optimizes the queries for parallel processing. Passes the operations to compute nodes to do the work in parallel.
3. The compute and storage nodes form the decoupled storage and compute cluster. Which makes it easy to scale out the compute and storage independently.
4. Grow and shrink (scale out and in) compute power within a dedicated sql pool, without moving the data.
5. Pause the compute capacity to stop billing and cost effective solution, without affecting the underlying ADLS data storage.
6. Resume the compute capacity of SQL DW during business hours.
7. SQL DW is built on MPP architecture (Massively parallel processing) architecture, which encompasses the benefits of compute and storage in decoupled layers.

Data Movement Service (DMS) in SQL DW

1. The Data movement service (DMS) is a system-level internal service which moves the data across the nodes as required to execute the queries in parallel and return the appropriate results.

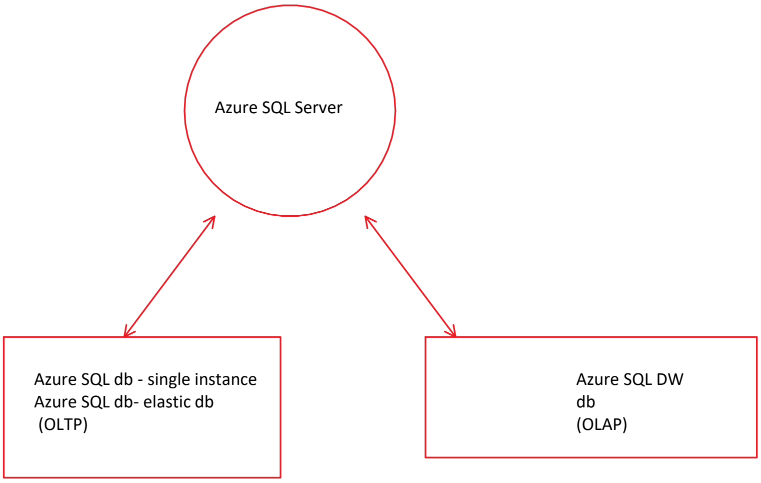
2. Azure Storage / ADLS gen2

Dedicated SQL pool/ SQL DW uses the ADLS to keep the data safe and scalable. In SQL DW, the cost is two-fold. Compute + Storage = DWU. The data is shared into distributions to optimize the performance of the systems. Which partitions/sharding pattern to use to distribute the data when loading into the table.

a) Hash
b) Round-Robin
c) Replicate

Based on this data sharding pattern, these three types of tables are available in SQL DW.

- Hash distributed table
- Round-robin distributed table
- Replicated tables



SQL Server Fragmentation

- This fill factor and the fragmentation limit
- SQL monitor find out the page level splits and index fragmentation to alleviate the performance issue.
- The wrong performance comes when wrong indexing fragmentation happens. The index become gradually fragmented, as the data gets updated, added or removed.
- Index fragmentation involves with wrong table design, wrong choice of datatype and keys and rate of data changes.

Polybase

Polybase enables the SQL server instance to query the data with t-sql queries directly from SQL server, Oracle, Mongodb, Teradata, hadoop clusters, cosmos db, s3-cloud based data storage without separately installing any client library/tools. A key use case for Polybase feature is to allow the data to star in its original format and location.

We can virtualize the external data through the SQL server instance, so that it can be queried in place like any other table in SQL Server, This process overall minimizes the need for ETL processes for data movement. We can write queries using t-sql to join the data from the external sources to relational tables in an SQL server instance.

- SQL Server 2016
- SQL Server 2017
- SQL server 2019
- Azure Synapse Analytics
- PDW hosted in APS

Benefits of Polybase

1. Helps to load bulk volume of data from traditional data sources to SQL server/ Azure Synapse without third party client libraries or softwares.
2. Polybase helps in the bidirectional data transfer of the data between Synapse/ dedicated SQL pool and external resource to provide fast data load performance. Data does not need to be copied into the SQL pool in order to access it.
3. We can use external tables through which the data stored in external file storage can be queried.
4. Polybase also allows to join data from a SQL server instance with external data.

Hands on Lab

1. Loading data into Azure SQL datawarehouse - dedicated SQL pool

a) Create a user designated for loading data
b) Create the tables for the sample dataset
c) Use the COPY statement to load the data into SQL datawarehouse
d) View the progress of data as it's loading (through DMV)

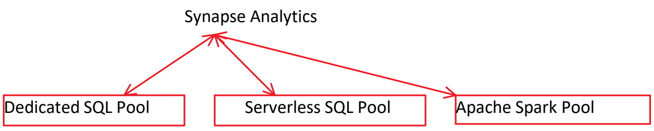
Distributions over SQL data warehouse

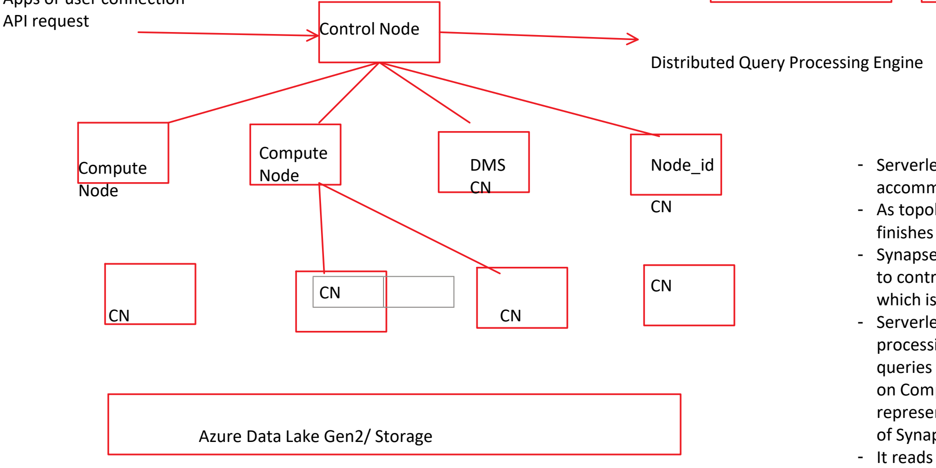
a) **Round-robin distribution** - is useful for improving loading speed. Significant impact over query and data loading performance
b) **Hash-distribution** - improves the query performance for large fact tables (columns)
c) **Replicate** - it has the full copy of the table accessible on each compute node. Replicating a table removes the need for transfer data among Compute nodes before a join/aggregation.

Benefits of COPY statement in Azure SQL DW

a) Use lower priviledged users to load the data without needing any strict CONTROL permission on the dw.
b) With a single t-sql statement without being having to create any additional database objects, execute and load the data into SQL data warehouse tables.
c) Properly parse and load the CSV data where delimiters (string, rows) are escaped within the string delimited columns
d) Specify a custom row terminator for CSV files
e) Leverage SQL server data formats for CSV files.
f) Customize default values for each target columns and specify the source data fields to load the data into specific target columns.

Azure Synapse Serverless SQL Pool





- Serverless SQL pool, scaling is done automatically to accommodate query resource requirements
- As topology changes over the time, for adding, removing and finishes successfully,
- Synapse SQL uses a node-based architecture, apps can connect to control node by issuing t-sql commands, to a control node which is the single point of query for Synapse sql
- Serverless SQL Pool control node utilizes the Distributed query processing (DQP) engine to optimize and orchestrate the user queries by splitting into smaller queries which will be executed on Compute nodes. Each small query is called task and represents the distributed execution unit for serverless sql pool of Synapse.
- It reads the file from storage, join results from other tasks, groups or orders data retreived from other tasks.

Serverless SQL Pool benefits

1. With decoupled storage and compute, when using Synapse SQL, enterprise can get benefitted from the independent sizing of compute power irrespective of the storage requirements.
2. For Serverless SQL pool, scaling is done automatically,
3. We can scale out or scale in the compute power within a dedicated SQL pool, without moving data
4. Pause compute capacity while leaving the data intact so that only we can pay for storage
5. Resume back the compute capacity during the operational hours.

Control Node

1. In Serverless SQL pool, the DQP engine runs on Control Node to optimize and co-ordinate distributed execution of user query by splitting into smaller queries which will be executed on Compute nodes, it also assigns the sets of files/storage to be processed by each node.

2. Whenever tsql query gets submitted, the control node can optimize and co-ordinate the parallel queries.

Compute Node

- Provides the computation power

- Distributions map to the Compute nodes for processing. As pay for more compute resources, pool remaps the distributions to the available compute nodes,
- Total no of compute nodes can range from 1-60 determined its underlying service tier
- Each every node is associated with the node_id
- Each compute node is also assigned task and set of files to execute the task on.
- Tasks is distributed query execution unit which is actually being the part of the query submitted by user.
- Automated scaling is in effect to confirm enough compute nodes are utilized to execute the user query.

Data Movement Service (DMS)

- Its a data transport tech stack in dedicated sql pool which co-ordinates the data movement between the compute nodes.
- Some queries require data movement to confirm the parallel queries should return the correct results.
- When the data movement requires, DMS ensures the right data gets moved to the right compute nodes.

Data Loading Best Practices in dedicated SQL Pool

- Prepare data in Azure Storage

1. To minimize the latency, collocate the storage layer and the dedicated sql pool.
2. To avoid out-of-memory error, while exporting data into ORC, RC, Gzip format
3. All file formats having different performance characteristics, for fastest load, use compressed delimited text files.
4. Split large compressed files into smaller compressed files

- Run loads with enough compute compute power

1. Create loading users designated for running the data load operations
2. Assign each loading users then run the load
3. The load runs with the users resource class.

- Load into a staging table

1. To achive the faster loading speed, for moving data into DW table. Load data into a staging table which is heap table use round-robin distribution
2. First load the data into the staging table then loading into the production table.

- Load to a clustered column store index

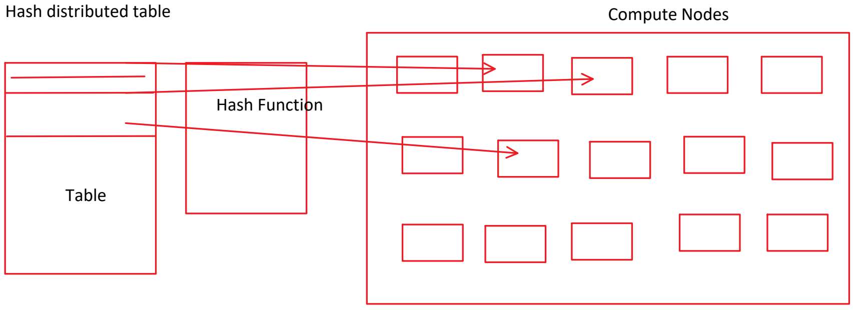
Distributed Table

A distributed table appears as a single table, but the rows are actually being stored across 60 distributions. The rows are distributed with a hash or round-robin algorithm.

- How large is the table?
- How often is the table refreshed?
- Do I have fact and dimension table in a dedicated SQL pool?

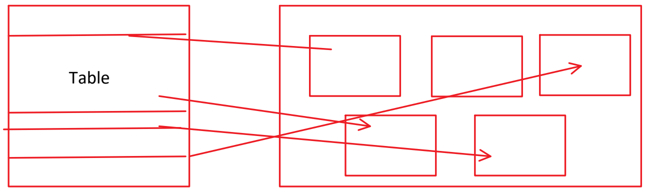
a) Hash distributed
b) Round-Robin distributed
c) Replicated

Hash distributed table



- A hash-distributed table rows maps across the Compute nodes by using a deterministic hash function to- assign each row to one direction
- Identical values always hash to the same distribution unit.
- SQL DW analytics has a built-in knowledge of the row location of tables.
- In dedicated sql pool, this knowledge is utilized to minimize the data movement between the queries.
- Which also improves the query performance.
- For Star schema, Large fact tables are designed with the hash-distributed table format
- When the table size on disk 2 - 5 GB, table has large no of rows and frequent insert, update, delete operations.

2. Round-Robin Distribution



- Distributes the table rows evenly across all distributions.
-
- The assignment of rows to distribution is random
- Rows with equal values are not guaranteed to be assigned in the same direction

Scenarios

1. Design the dimension table with large no of columns
2. There's no specific joining key
3. When the table is temporarily stored staging table
4. There's no common joining key available with other key tables

Table Distribution	Hash-distributed	Round-Robin	Replicated
Scenario	Large Fact table of large of no of Rows	Dimension tables with large no of columns	Dimension tables with full copy option from round-robin table
Size	Size of Fact table is at least 5 GB or more	Dimension table size is at least 2 GB	Replicated table 2 GB or less
Performance Efficiency	Most performance efficient since hash distribution key is imposed using the distribution column	Less performance efficient since the table rows are distributed randomly over the compute nodes of the cluster, not matching same order the compute nodes while distribution of rows	Full table copy is performed over the same compute node, less data movement (DMS) , maximizes the throughput and performance

Hands-on Demos

Copy data / load the data from Azure SQL database to Azure Synapse Analytics using Azure Data Factory
BCP utility
SSIS packages

Task 1:

Copy data / load the data from Azure SQL database to Azure Synapse Analytics using Azure Data Factory

- **Managed services environment**
- **Rich data source support**
- **Encryption at rest in transit of data**
- **Performance imposed using polybase**

Pre-requisites
(same Resource Group + Region) = Cost effective

1. Azure Synapse Analytics
2. Azure SQL db (sample db)
3. Azure Data Factory
4. Azure Storage account (staging blob)

Task 2: Perform data loading to Azure Synapse Analytics dedicated sql pool db using Synapse Studio (Assignment)

Troubleshooting for ADLs gen2

1. **Check your Microsoft.Synapse resource provider**
2. **Check your IAM permission for ADLs gen2**

3. **Give a unique name for ADLS gen2**

Determination of Table Category in Azure SQL data warehouse

A Star Schema consists of tables like Fact and Dimension tables.

Table	Fact Table	Dimension Table	Integration Table
	Contains the quantitative data which are commonly Generated by a transactional system. Then data gets loaded into the dedicated SQL pool. e.g. Sales fact table	Contains the attribute data which might change but usually changes infrequently. e.g. CustomerID can be sharable by both Fact and Dimension table. Instead, the fact table and Dimension table can be joined query over the two tables to associate a customer's profile and transactions	Provide a place for integrating or staging data. We can create an integration table as a regular table , an external table or a temporary table. e.g. loading the data into the staging table, perform transformations on the staging data and insert the data into the production table

Table Persistence Concepts

Data for SQL Pool is stored in Azure Storage, temporarily stored in ADLS/Storage, in a data store external to dedicated SQL pool

a) Regular Table - A regular table stores the data in Azure Storage as part of dedicated SQL Pool. The Table and the data persist regardless of whether the session is open.

a) Temporary Table- A temporary table only exists for the duration of the session. We can use a temporary table to prevent other users from seeing temporary results and also to reduce the need for cleanup.

b) External Table - An external table points to the data located in Azure blob storage or ADLS. When used with the CREATE TABLE AS SELECT statement (CTAS), selecting from an external table imports data into the dedicated SQL pool.

Data Types

- Int
- Smallint
- Tinyint
- Nvarchar(max)
- Varchar
- Decimal
- Numeric
- Money
- Float
- Real
- Date
- Datetime

Create Table As Select (CTAS)

Is one important tsql feature. It is a fully parallelized operation which creates a new table based on the output of a SELECT statement. CTAS is the most simplest and fastest way to create a copy of the table.

- Benefits of CTAS statement

1. Recreate a table with hash-distributed column
2. Recreate a table as replicated
3. Create a columnstore index or just some cols in the table
4. Query or import the external data

Index Considerations on SQL pool tables

Dedicated SQL Pool offers several indexing options like
- Clustered ColumnStore indexes - created by SQL pool table when no index options are specified
- clustered indexes - indexes are applied on the list of columns
- non-clustered indexes

Scenarios where not to use Clustered Columnstore index

a) Columnstore tables do not support varchar(max), nvarchar(max), varbinary(max) data types. If you need to define such data types for your table, ensure to use clustered/ non-clustered index.
b) For temporary table, do not use clustered columnstore index
c) Small tables with less than 60 million rows, consider to use clustered index

Benefits of Clustered ColumnStore index

a) Clustered ColumnStore index organize the data into segments.

Partitioning Concepts on Dedicated SQL Pool:

Table partitions enable us to divide the data into smaller groups of data. In most of scenarios, table partitions in dedicated SQL pool are created based on Date column,

Partitioning is supported on all dedicated SQL pool table types, including the clustered columnstore indexes, clustered indexes and heap.

Partitioning is also supported on all table distribution types - Hash distributed, Round-robin distributed and Replicated.

Partitioning can benefit data maintenance and query performance. Whether it benefits both of just one is dependent on how data is loaded and whether the same column can be used for both of the purpose. Since partitioning can be applied on one column of sql pool table.

Primary Benefits of SQL pool table partitioning

1. Improve the efficiency and performance of loading data by use of partition deletion, merging, switching.

- A partition switching can be used to quickly remove or replace a section of table.
- Using partitioning during the overall load process can substantially improve the performance
- Recommended approach while deleting the rows over a million rows of table, just to optimize the time and reduce the risk of large transactions which can take a long time to rollback, optimized approach is to drop the oldest partition of data.
- Deleting rows one by one take long time, while deleting a partition takes a sec.
- Applied a filter to the partitioned data can limit the scan to the qualifying partitions while overall improves the query performance.

2. Ensure the appropriate number of partitions are getting created and use of partitions to be carefully chosen.
3. Creating partitions on clustered columnstore tables.

Hands-on Lab - 1

1. Create a Synapse Analytics Workspace
2. Create the external data source (using CREATE EXTERNAL DATA SOURCE)
3. Configure the data format (CREATE EXTERNAL FILE FORMAT) specify the format of the data file going to be stored in the text files of the tables, also include the field delimiter, date-format, type-default etc.
4. Create the external tables
5. Create another schema
6. Load the data into the new tables from the old tables using CTAS statement
7. Sample queries to test
8. Check the performance statistics
9. Apply the performance tuning and optimization principles

Statistics over Synapse SQL

1. Statistics is applied for query optimization purpose. The more dedicated SQL pool knows the data location, the faster it can execute the query against it. After loading the data into the dedicated SQL pool, collecting statistics on the data is one of important aspect to optimize the queries.

The dedicated SQL pool query optimizer works as a cost-based optimizer. It compares the cost of various query plans, then chooses the plan with the lowest cost. In most of the cases, the plan with the fastest query execution is selected.

AUTO_CREATE_STATISTICS is flag is configured on dedicated SQL pool, then it will trigger the automated statistics creation.
UPDATE
DELETE
EXPLAIN

2. Hands-on Lab 2:

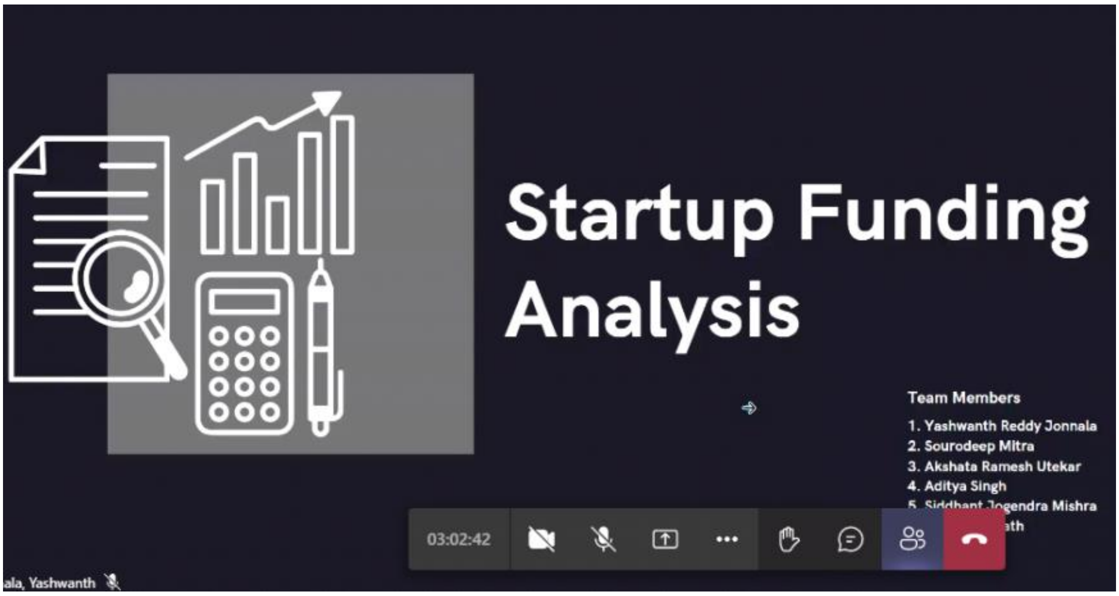
a) Create a serverless pool db
b) Create External Data Source
c) Create view based on external dataset with OPENROWSET Function with BULK data load option
d) Connect PBI desktop to the SQL pool db
e) Create report based on the view

Case Study

20 October 2022 11:38

1. Zomato Analysis (Zomato Bengaluru Dataset)
   - Twishampati Ghosh
   - Srihitha
   - Deepak Yadav
   - Ashwin Sonare
   - Prashik Mayur
   - Anubhav Saha

2. Startup Funding Analysis



Startup Funding Analysis

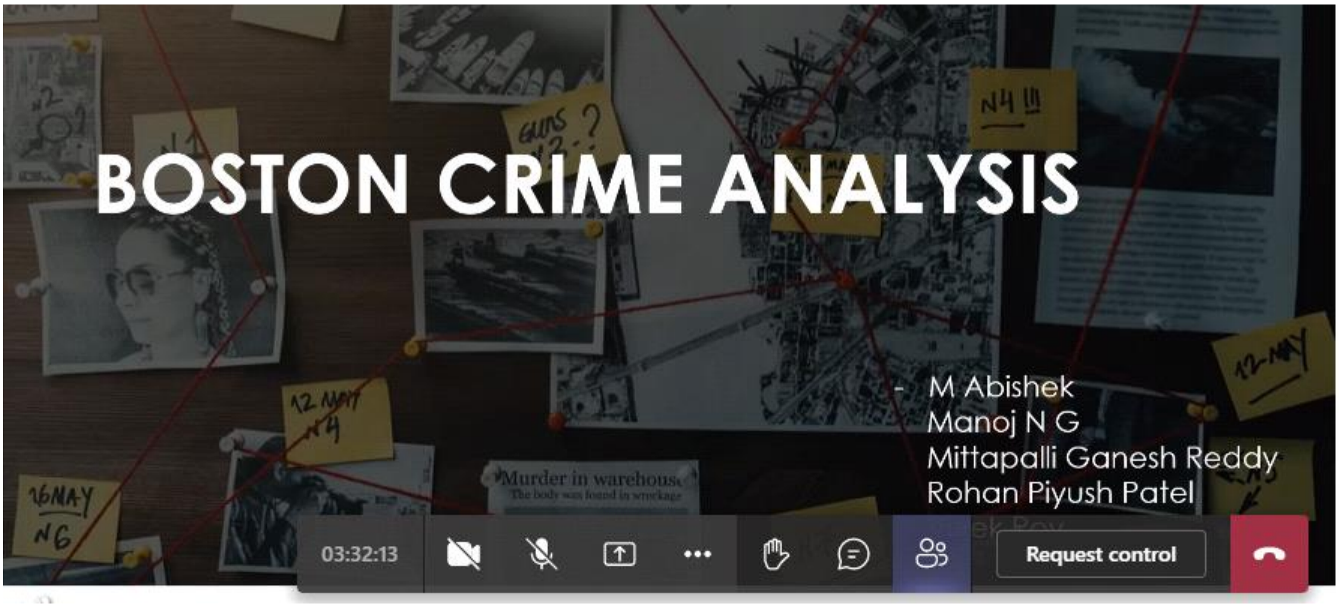
Team Members

1. Yashwanth Reddy Jonnala
2. Sourodeep Mitra
3. Akshata Ramesh Utekar
4. Aditya Singh
5. Siddhant Jogendra Mishra

Velkur Bharath

3. Boston Crime Analysis

4.



BOSTON CRIME ANALYSIS

- M Abishek
  Manoj N G
  Mittapalli Ganesh Reddy
  Rohan Piyush Patel

4. Airline Analysis

Case Study on Airlines Analysis

Batch: BI V7 with MS Azure PT Sep 6th Batch1

Group Members:

1. Harshit Mudgal
2. Abhaya Shankar
3. Armaandeep Sandhu
4. Ashitosh Joshi
5. Abhishek Baghel

Presentation Date: 20th October 2022

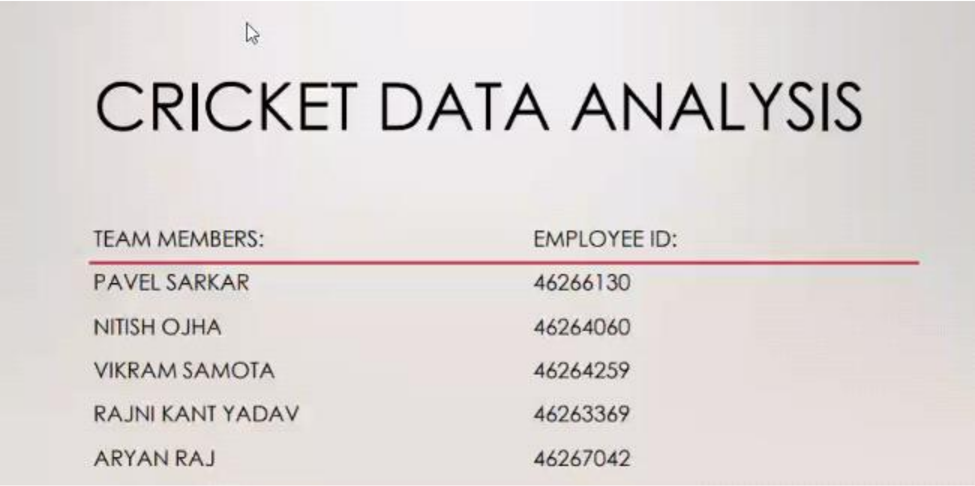
5. IPL Analysis

IPL ANALYSIS

Team :

Sahib Singh	- 46264890
Charan Reddy Thathykalva	- 46265397
Vivek kumar	- 46263355
konduru Daiva Preethi	- 46263570
Vatsal Raj	- 46263571

6. Cricket Data Analysis



CRICKET DATA ANALYSIS

TEAM MEMBERS:	EMPLOYEE ID:
PAVEL SARKAR	46266130
NITISH OJHA	46264060
VIKRAM SAMOTA	46264259
RAJNI KANT YADAV	46263369
ARYAN RAJ	46267042

Apache Spark

08 September 2022 18:48

Sprint 2 Topics

1. Apache Spark
2. Python
3. Azure Databricks
4. AWS and GCP overview

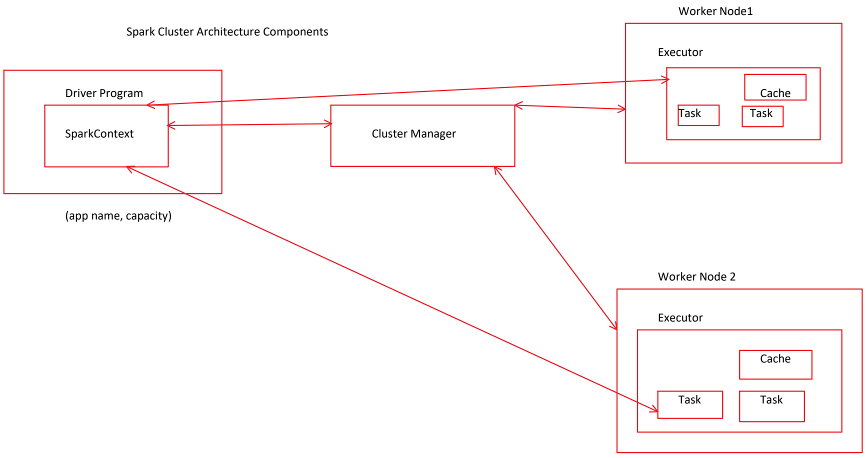
Difference between Hadoop and Spark

Features	Apache Hadoop	Apache Spark
Definition	Hadoop is the open-source distribution allows users to manage The big data sets over the nodes to solve the vast and intricate data problems.	Apache Spark is also an open source distributed stream processing engine for big data sets.
Core features	Highly scalable, cost-effective solution which stores and processes structured, unstructured data but requires good amount of infra capacity to execute the data pipeline faster	Spark splits up the large tasks across the different nodes. However, it tends to perform faster than hadoop and uses the RAM to cache and process data instead of the filesystem. Which enables Spark to handle use cases that hadoop can't do
Benefits	a) Data protection amid of hardware failure b) Vast scalability from a single server to thousands of machines c) Real time analytics for historical analyses and decision making processes	A unified processing engine which supports SQL queries, streaming data, ML and graph processing
Ecosystem	Hadoop supports advanced analytics for stored data (e.g. predictive, data mining, ML etc.). It enables big data processing tasks to be split into smaller tasks. The smaller tasks are performed in parallel by using an algorithm (e.g. MR) and then distributed across the hadoop cluster.	Apache Spark is the largest open source project in data processing. It combines the feature of both AI and ML. It enables the users to perform large-scale data transformations and analysis. Then it runs state-of art ML and AI algorithms
Components	a) HDFS - primary data storage system which manages the large datasets running on commodity hardware. b) It also provides high-throughput data across and high fault tolerance c) YARN - Cluster resource manager which schedules tasks and allocates resources (e.g. CPU and memory) to applications d) Hadoop MR - splits big data processing tasks into smaller ones, distributes the small tasks across the different nodes which runs each task e) Hadoop Core - set of common libraries and utilities on which the modules like HDFS, YARN, MR etc. depend on	a) Spark Core - underlying execution engine which schedules and dispatches tasks and coordinates I/O operations. b) Spark SQL - Gathers information about structured data to enable users to optimize the data processing c) Spark Streaming / Structured streaming - Both of streams could add data stream processing capability. Spark streaming takes data from different streaming sources and divides it into micro-batches for a continuous stream. Structured streaming is built on Spark SQL, reduces the latency and simplifies the programming. d) GraphX - user-friendly computation engine which enables interactive data building, modification and analysis of scalable and graph-structured data.
Core processing	Hadoop MR processes the data in disk	Spark is hadoop enhancement to MR. The primary difference between MR and Spark is spark processes and retains data in-memory for subsequent steps,
Benefits	MR is comparatively slower then Spark	Spark's data processing capacity is 100x times faster than MR
Job Execution	Data is being processed through two stages - Map - Reduce	Spark creates a directed-acyclic graph (DAG) to schedule tasks and orchestration of nodes across the hadoop cluster.
	Mapper and reducers are being executed including partitioners and combiners while processing the data	The task-tracking process enables fault-tolerance, which reapplies recorded operations to data from a previous state

Spark Components Overview

a) Spark applications run as independent set of processes on a cluster, co-ordinated by the SparkContext object in the main program(driver program).
b) Specifically, while running on a cluster, the SparkContext can connect to several types of cluster managers (either Spark's own standalone cluster manager, mesos, YARN or kubernetes) which allocates resources across the applications.
c) Once connected, Spark acquires executors on nodes on the cluster. Which are the processes that can run the computations on nodes on the cluster.
d) Which are processes that execute the computations and store data for your application.
e) Next, it can send the app code (JAR, python files passed to SparkContext) to the executors.
f) Finally, SparkContext sends tasks to the executors to run

Spark Cluster Architecture Components



1. Each Application in Spark gets its own executor processes. Which stay up to date during the whole applications execution duration and run tasks in multiple threads.
2. This has the benefits of isolating applications from each other, or both of the scheduling side(each driver schedules its own tasks) and executor side, (tasks from different applications run in different JVMs).
3. However, it also means data cant be shared across the different spark apps (instances of SparkContext) without writing to an external storage system
4. Spark is agnostic to the underlying cluster manager. As long as it can acquire the executor processes, these can communicate with each other , its relatively easy to run it even on a cluster manager which supports other apps (e.g. Mesos, kubernetes, YARN) .
5. The Driver program must listen for and accept incoming connections from its executors throughout its lifetime. As such the driver program must be network addressable from the worker nodes.
6. Because the driver schedules tasks over the cluster, it should be close the worker nodes, preferably on the same local area network, if you'd like to send requests to the cluster remotely, it's better to open an RPC (remote procedure call) to the driver and have it submit operations from the nearby then to the execution of driver far away from the worker nodes.

Driver Program	The process running the main() function for the application and responsible to create the SparkContext. Once the SparkContext is initialized then it can start interact with the executors for job processing
Cluster Manager	An external service for acquiring the resources on the Spark cluster (e.g. standalone mode, Mesos, Kubernetes)
Worker Node	Any node which can run the application code in the cluster
Executor	A process launched for an application on a worker node. It runs tasks and keeps the data on memory or disk storage across them. Each application has its own executors.
Tasks	Core unit of work which will be transferred to one executor
Job	A parallel computation unit consists of multiple tasks that gets spawned over in response to Spark action (e.g. save, collect etc.)
Stage	Each job passes over different stages. Its get divided into smaller set of tasks called the stages which depend on each other.
Application	User program built on Spark. It consists of a driver program and executor on the cluster

Cluster Manager types available in Spark

1. Standalone - a simple cluster manager included with Spark what makes it easy to setup a cluster
2. Apache Mesos - a general cluster manager which can also run Apache hadoop mapreduce and service apps
3. Hadoop YARN - the resource manager in hadoop 2 and hadoop 3
4. Kubernetes - an open source system for automating deployment, scaling and management of containerized applications.

Monitoring - Each driver program in Spark has an web UI. Typically available over the port 4040. Displays the information about the running tasks, executors and storage usage.

The first maps a line to an integer value while creating a new dataset.
Reduce is called on the dataset to find the largest word count.

The arguments are used here to map and reduce functions and Scala function literals and can use any language feature or scala/java library.

The flatMap function is used to transform a dataset of lines to a dataset of words.
Then combine groupByKey and count to compute the per word basis counts in the file as a dataset of (String, Long) pairs.
To collect the word counts in our shell, we can call it with collect() operator in Spark.

Identity is a pre-defined in-built function in Spark scala which helps to identify the grouping of values for each words / max words occurrence by key.

Resilient Distributed Dataset (RDD)

Spark revolves around the concept of resilient distributed dataset (RDD), it is a fault-tolerant collection of elements which can be operated on in parallel. There're two ways to create RDD.

a) Parallelizing - an existing collection of dataset can be executed in parallel in the Spark driver program
b) Referencing a dataset in an external storage system - HDFS, HBase or any other data source offering a Hadoop InputFormat.
c) Existing RDD

Operations on RDD:

1. Transformation
2. Action

Transformation:

In Spark, the role of transformation is to create a new dataset from an existing one. The transformation are being considered as lazy as they are only being computed when an action requires a result to be returned to the driver program.

(Lazy Transformation, Lazy Evaluation)

Transformation Function	Description	
map(func)	It returns a new distributed dataset formed by passing each element of the source through a function func	
Filter(func)	It returns a new dataset formed by selecting those elements of the source on which the func returns true	
flatMap(func)	Here, each input item can be mapped to zero or more output items, so func should return a sequence rather than a single item	
MapPartitions(func)	It's similar to Map, but it runs separately on each partition(block) of the RDD, so func must be the type of Iterator<T> => Iterator<U> when returning on an RDD of the datatype T	
Union(otherDataset)	It returns a new dataset which contains the union of elements in the source dataset and the argument	
Distinct([numPartitions])	It returns a new dataset which contains the distinct elements of the source dataset	
groupByKey([NumPartitions])	It returns a dataset (K, iterable) pairs when called on a dataset of (K,V) pairs.	
reduceByKey(func, [numPartitions])	When called on a dataset of (K,V) pairs, returns a dataset of (K,V) pairs, returns a key/value pairs, where the values of each key are aggregated using the given reduce function func, which must be of type (v,v) => v	
sortByKey([ascending], [numPartitions])	Returns the dataset of key/value pairs sorted over the keys in ascending/ descending order	

Narrow Transformation

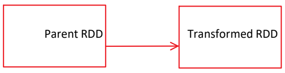
- Result of map, filter and such that data is from the single partition only. It's self -sufficient.

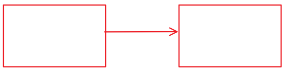
- An output RDD has the partitions with records which originates from a single partition in the parent RDD.

Examples

- Map
- FlatMap
- MapPartition
- Filter
- Sample
- Union

Spark groups the narrow transformations as a stage called as pipelining





Actions

In Spark, the role of Action is to return the value to the driver program after running the computation on the dataset.

Actions	Description
Reduce(func)	It aggregates the elements of the dataset using a function func(which takes two arguments and returns one argument). This function should be commutative and associative so that it can be computed correctly and in parallel.
Collect()	It returns all the elements of the dataset as an array at the driver program. This is usually useful after a filter or other operation which returns sufficiently small subset of data
Count()	It returns the number of elements in the dataset
First()	Returns the first element of the dataset(similar to take(1))
Take(n)	It returns an array with the n number of elements in the dataset
takeSample/takeOrdered	It returns an array with the random sample of num elements in the dataset, with or without replacement
saveAsTextFile(path)	It's used to write the elements of the dataset as a text file in a given directory of the local filesystem. HDFS/S3 , spark calls toString on each element to convert it to a line of text in the file
saveAsSequenceFile(path)	It's used to write the elements of the dataset in a simple format as Hadoop SequenceFile in a given path in the local filesystem.
CountByKey()	It's only available on RDDs of type (K,V). Thus, it can return a hashmap of (K,int) pairs with the count of each key.
Foreach(func)	It returns a function func on each element of the dataset for side transformations like updating the input dataset or changing the values of variables.

RDD is the fundamental data structure of Spark which consists of immutable collection of objects and which computes on the different nodes of the cluster.

Core Benefit of RDD

- Iterative algo
- Interactive data mining tools
- DSM (distributed shared memory) -the read operation on the distributed shared memory is fine-grained.

Each and every dataset in Spark RDD is logically partitioned across many servers so that they can be computed on different nodes of the cluster.

- Resilient - since fault-tolerant with the help of RDD, a lineage graph (DAG) and so able to recompute missing or damaged dataset partitions due to node failures.
- Distributed
- Dataset represents the records of the data being work with. The user can load the dataset externally which can either be JSON, csv, text or JDBC .

Caching in RDD

The Spark RDD can also be cached and manually partitioned. Caching is useful when we require to use RDD multiple times. Since, manual partitioning is important to correctly balance partitions. Generally, smaller partitions allow distributing RDD data more equally, among many executors. Hence, fewer partitions make the work easier.

RDD Persistence

Spark has the capability to provide a convenient way to work on the dataset by persisting it in memory across operations. While persisting an RDD, each node stores any partitions of it so that it can compute in memory. Now, we can reuse the RDDs on other tasks on that dataset.

We can use either persist() or cache() method to mark an RDD to be persisted. Spark's cache is fault tolerant.

Benefits

1. In case of missing partition, lost partition, the respective RDD partition will automatically be recomputed using the transformation which originally creates it.
2. There're different storage levels can be used to store persisted RDDs. These storageLevels can be used by passing StorageLevel object to persist().
3. The cache() method is used for the default storage level, for default storage level, RDD caching, the storage level type is StorageLevel.MEMORY_ONLY

StorageLevel in RDD for persistence and Caching

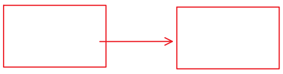
Memory_only	Stores the RDD as deserialized java objects on JVM. This is the default storage level of persisted RDD. If the RDD doesn't fit into the memory, some partitions will not be cached and recomputed each time when required
MEMORY_AND_DISK	Stores the RDD as deserialized java objects in JVM, if the RDD doesn't fit into the memory, store the partitions if doesn't fit in the disk, it will be read from there when they're needed.
MEMORY_ONLY_SER (Scala & Java)	Stores RDD as serialized java objects. This is generally more space -efficient than deserialized objects.
MEMORY_AND_DISK_SER (Scala & Java only)	Spill partitions which doesn't fit in memory to disk instead of recomputing them.
DISK_ONLY	Stores the RDD partitions only on disk
Memory_Only_2 Memory_AND_Disk_2	Replicate each RDD partition onto 2 cluster nodes.

Core Features of RDD

1. In-memory computation
2. Lazy Evaluation
3. Immutability
4. Fault Tolerance
5. Persistence - users can choose which RDD will be reused and based on that choose the appropriate, storage level
6. Partitioning - fundamental unit of parallelism in Spark RDD. Each partition is one logical division of data which is mutable. One can create a partition through some transformations on existing partitions.
7. Location-stickiness - RDDs are capable of defining placement preference to compute partitions. Placement preference the location of RDD. The DAGScheduler places the partitions in such a way that tasks in close to the data as much as possible. Thus, can speed up the computation.
8. Coarse-grained operation

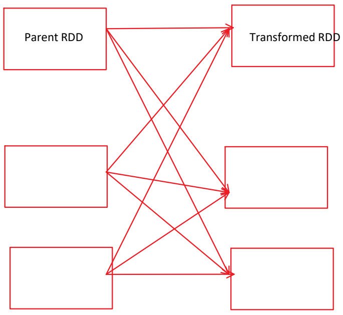
Paired RDD

RDDs can created in the form of key/value pairs. Key/value pairs actually contains two linked data item in it.



Wide Transformation

It's generated as a result of groupByKey() and reduceByKey() like functions. The data required to compute the records in a single parition can live in many different partitions of the parent RDD.
Wide transformations also are known shuffle transformation.



- Intersection
- Distinct
- ReduceByKey
- GroupByKey
- Join
- Cartesian
- Coalesce
- repartition

- Key (identifier)
- Value (data value corresponding to the key)

Benefits

- In Spark the paired RDDs ReduceByKey() can aggregate data separately for each key. Whereas, for join(), merges two RDDs altogether by grouping elements with the same key.
-
- transformation
-
- groupByKey
- reduceByKey
- combineByKey
- Keys()
- Values()
- sortByKey()

Actions

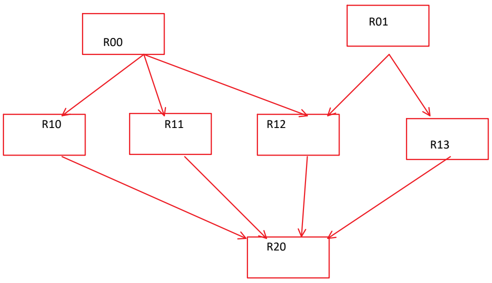
- countbyKey()
- collectAsMap
- Lookup(key)

RDD Lineage

When a new RDD is created from an existing Spark RDD, the new RDD carries a pointer to the parent RDD in Spark. This is the same as all the dependencies between the RDDs those are logged in a graph, rather than the actual data. It is called as lineage graph.

RDD lineage is graph of parent RDD.

- Physical execution plan or execution DAG is known as the DAG of stages
- Output of applying transformations to the spark. Then, it creates a logical execution plan.



Filter Transformation operator
Distinct
Count
groupByKey
sortByKey

- Create an RDD using Parellized collection
- Generate the result using collect()
- Apply filter transformation

groupByKey Function in RDD

It's used for transformation operation which performs the shuffling of data. It receives the key-value pairs (K,V) as input group the values based on the key and generates a dataset of (K, iterable) pairs as an output

Org.apache.spark.scala.util.collection -> CompactBuffer[V]

Shared Variables in RDD

When a function is passes to a Spark operation (map or reduce) is executed on a remote cluster node. It works on separate copies of all variables used in the function.
These variables are copied to each machine, and no updates to the variables on the more machine can be propagated back to the driver program.
Two kinds of Shared variables for RDD operations

a) Broadcast variable
b) Accumulators

a) Broadcast variables

Broadcast variables allows the developers to keep a read only variable cached on each machine rather than the shipping a copy of it with tasks. They can be reused. For example, to provide every node a copy of a large input dataset in an efficient manner.

1. Spark also attempts to distribute broadcast variables using efficient broadcast algorithms to reduce the communication cost.
2. Spark actions are executed through the set of stages, separated by distributed "shuffle" operations.
3. Spark automatically broadcasts the common data required by tasks within each stage. The data broadcasted this way cached in serialized form and deserialized before running each task.

4. Which means the explicitly creating the broadcast variables are only useful when the tasks across the multiple stages require the same data or when caching the data in deserialized form is important.

Once the broadcast variable is created, it should be used instead of the value v in any functions run on the cluster so that v is not shipped to the nodes more than once.

b) Accumulators

Accumulators are the variables which are only added to through an associative and commutative operation and can therefore be efficiently supported in parallel. They can used to implement counters or sums. Spark natively supports accumulators of numeric types where the developers can add the support for new types.

Feature	Broadcast Variable	Accumulator
	It can be used to cache a value in memory on all nodes	Accumulator which are variables that can be only added to such as counters and sums
	Allows the developers to keep a read-only variable cached on each machine/node rather than shipping a copy of it with tasks	Added to through an associate operation and can therefore be efficiently supported in parallel.
	Broadcast variable in Spark are cached on executor side.	Accumulators are variables which are used for aggregating info across the executors.

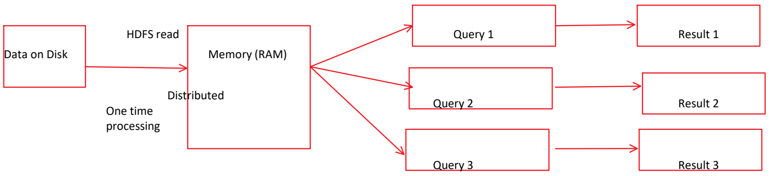
	e.g. if there're 10 executors are running on Spark worker nodes and in your app execute 100 tasks in total. The broadcast variables will be sent to the 10 executors as opposed to 100 times	

DataFrame in PySpark

The Dataframe is a distributed collection of data organized into named columns. It is conceptually equivalent to a table in RDBMS, similar like that, we create dataframe in Python. With more enhanced libraries support (numPy, SciPy, Pandas, Matplotlib)

Dataframes can be constructed from a wide array for sources such as structured array / data files, hive tables, external databases or existing RDDS.

Interactive Operations in RDD



Spark SQL

1. Spark SQL is a Spark module for structured data processing.
2. Interaction with Spark SQL can take place through SQL and Dataset API.

   Benefits

a) The major benefit of Spark SQL is to execute SQL queries. Spark SQL can also being used to read data from an existing hive table.
b) When running SQL query on Spark SQL from any programming language , the results will be returned as a Dataframe/Dataset.
c) We can interact with the SQL interface using the CLI and JDBC/ODBC.
d) Through Spark SQL, can build the relational data tables over RDD
e) Import relational data from Parquet, avro, csv/json and hive tables
f) Run SQL queries over imported data and existing RDDs.
g) Easily write RDDs out to Hive tables or parquet/avro/csv files.
h) Spark SQL includes a cost-based optimizer, columnar storage and code generation to make queries faster.
i) At the same time, it scales, to thousands of nodes to multi-hour queries using the Spark engine.
j) It provides full mid-query fault tolerance, without having to using any different engine for historical data.

DataSets and DataFrames

a) A dataset is a distributed collection of data.
b) The Dataset is a new interface added in Spark 1.6 onwards which provides the benefits of RDDs (strong typing, ability to use the lambda function) along with the benefits of Spark SQL's optimized execution engine.
c) A dataset can be constructed from JVM objects and then can be manipulated using functional transformations (map, flatmap, filter).
d) The Dataset API is available in Scala/ java. Python doesn't have support for dataset API.

DataFrame

a) A dataframe is a dataset organized into named columns. It's conceptually equivalent to a table in RDBMS or a dataframe in R/Python, but with richer optimization under the hood.
b) A dataframe can be constructed from a wide array of sources like structured data files, tables in hive , external databases or existing RDDs.
c) Dataframe API in spark is available in Scala, Python, Java, R
d) A dataframe is represented by a dataset of Rows.
e) Ability to process the data from KB to PB
f) Support for various file formats (Avro, Parquet, csv, HDFS, Cassandra)
g) State of the art optimization technique and code generation principles through the Spark SQL catalyst optimizer.
h) Can be easily integrated with all Big data tools and frameworks like Spark-Core
i) Provides API for Python, java, scala and R programming

SQLContext Class

SQLContext is a class and is used for initializing the functionalities of Spark SQL. SparkContext class object (sc) is required for initializing SQLContext class object.

Data Sources
Spark SQL supports operating on variety of data sources through the DataFrame interface. A DataFrame can be operated on using relational transformations and can also be used to create a temporary view.

Registering a Dataframe as a temporary view allows to run SQL queries over its data.

Caching / Performance Tuning

a) Spark SQL can cache tables using an in-memory columnar format by calling spark.catalog.cacheTable("tableName") or dataFrame.cache().

b) Spark SQL will scan only required columns and will automatically tune compression to minimize memory usage and GC pressure.
c) We can call spark.catalog.uncacheTable("tableName") or dataframe.unpersist() to remove the table from memory.

Spark SQL Functions

Spark SQL provides several built-in standard functions comes from org.apache.spark.sql.functions to work with DataFrame/Dataset and SQL queries. All these Spark SQL functions return org.apache.spark.sql.Column type.

- String functions (concat, ltrim, upper, lower)
- Date and Time functions (current_date, to_date, add_months, date_add, year, quarter etc.)
- Collection functions
- Math functions (Pi, sin, cos, tan, log)
- Aggregate functions (avg, min, max, median, count, collect_list)
- Window functions (window, dense_rank, rank, ntile, rank, row_number, lag, lead)

Supported file formats

- Csv
- Text
- Avro
- Parquet
- Tsv
- Xml / many more

Windowing functions

In pyspark, Window functions operate over the group of rows,(like frame, partition) and return a single value for every input row. Pyspark SQL supports three kinds of window functions

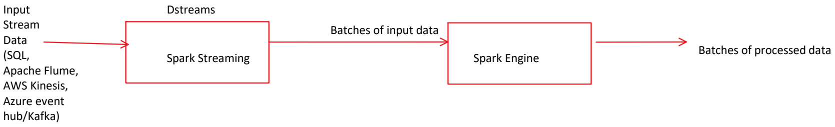
- Ranking functions
- Analytics functions
- Aggregate functions

Row_number - returns a sequential number starting from 1 within a window partition
Rank() : returns the rank of rows within a window partitions with gaps
Percent_rank: return the % of rank of rows within a window partition
Dense_rank(): returns the rank of rows within a window partition without any gaps.
Ntile(): returns the nth tile id in a window partition
Lag(): returns the value which is offset rows before the current row and null can be passed, If there is less than 'offset' rows are present before the current row.
Lead(): return the value which is offset rows after the current row, and null if there's less than 'offset' rows after the current row.

Array
Map
Date & timeStamp
JSON
Windowing
String

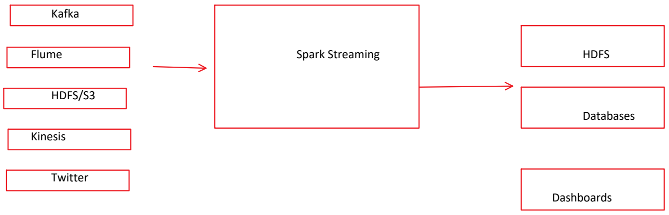
Spark Streaming

Spark Streaming works on the principles of micro-batches.


Input Stream Data (SQL, Apache Flume, AWS Kinesis, Azure event hub/Kafka)
Dstreams
Spark Streaming
Batches of input data
Spark Engine
Batches of processed data

Spark Streaming is able to process real-time from the various real time datasets as discretized streams (Dstreams) which are fundamental abstractions as they represent streams of data into small chunks.

a) Spark Streaming has gained rapid adoption because of its disparate data processing capabilities making it easy for developers to rely on a single framework to satisfy all processing requirements.
b) The data models can be trained offline using Mlib and then can be used for streaming data scoring using Spark Streaming.
c) Some models can learn and score continuously while streaming data is collected.
d) Spark SQL makes it possible to combine the streaming data with a wide range of static data sources.



- Spark Streaming provides a high-level abstraction called Dstreams or discretized streams, which represents a continuous stream of data.
- Dstreams can be created either from input data streams from sources like Kafka, Kinesis, or applying high-level operations on other Dstreams as represented as a sequence of RDDs.
- Spark Streaming starts with Dstreams. We can write the programs in Scala, java/python.

Data Processing steps in Spark Streaming

- Spark Streaming is a separate library in Spark which continuously flowing streaming data.
- It provides us the discretised Streams which called Dstreams empowered by Spark RDDs.
- Dstreams provide us data divided into chunks as RDDs received from the source of streaming to be processed.
- After processing, sends it to the destination.
-

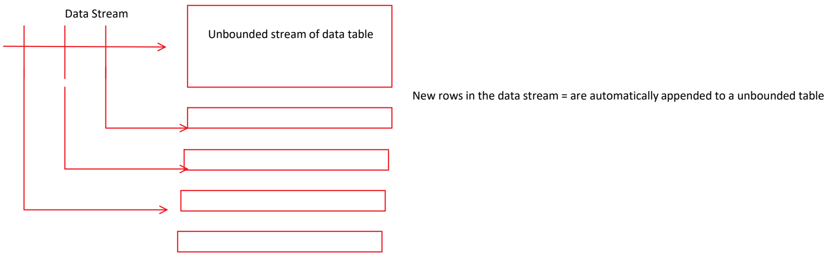
Spark Structured Streaming

Structured streaming is built on Spark SQL library. Structured Streaming is another way to handle streaming data with Spark. This model of streaming is based on Dataframe and Dataset API.

- We can apply any SQL query (using DataFrame API) or Scala (Dataset API) on streaming data,
- Started from Spark 2.x version onwards.

Spark Streaming	Spark Structured Streaming
Works on micro-batches. The stream pipeline is registered With some operations and Spark polls the source after every batch, duration. Then the batch is created of the received data. Each incoming record belongs to a batch of Dstream. Each batch represents an RDD.	Works also on the same architecture of polling the data after some duration, based on the trigger interval. There's no concept of batches. The received data in a trigger is appended into the continuously flowing data stream. Each row of data stream is processed and the result is updated into the unbounded result table. New data in the data stream , automatically, new rows are being appended to a unbounded streams of data table.
Spark Streaming is based on DStreams API	Uses dataFrame and Dataset API to perform streaming operations
Spark Streaming only works with the timestamp when the data is received by the Spark. Based on the ingestion timestamp, Spark Streaming puts the data in a batch even if the event is generated early and belonged to the earlier batch. It may cause into less accurate info leading to data loss.	Structured streaming provides the functionality to process the data on the basis of event-time when the timestamp of the event is included in the data received. Major feature - structured streaming provides a different way of processing the data according to the exact time of the data generation in real time. We can handle data coming on late events and get more accurate results.

Spark Structured Streaming


Data Stream
Unbounded stream of data table
New rows in the data stream = are automatically appended to a unbounded table

Spark Streaming also benefits of checkpointing. There's no guarantee of end to end exactly once semantics.	In Structured Streaming, though the utility of checkpointing, can provide end to end exactly once semantics. Avoiding the data loss. The destinations/sink must support idempotent operations
There's no restriction on the type of sink, RDD caching and performance tuning is possible. Streaming returns an RDD created by each batch one-by-one and can perform any actions over them, like saving or storage / performing some computations.	Structured streaming is now more flexible and gives an edge over the Spark streaming and over other flexible sinks.
	Structured Streaming, queries are processed using a micro-batch processing engine, which processes the data streams as a series of small batch jobs thereby achieving end to end latencies as low as 100 ms and exactly once fault tolerance guarantees through the checkpointing and Write ahead logs. Structured streaming provides fast, scalable, fault tolerant exactly-once stream processing without the user having to reason about streaming.

Data protection feature of Structured Streaming

- Structured streaming has data protection against node failure, a logs of journals are managed called as write ahead logs (WAL).
- Structure enforces fault-tolerance capacity by saving all data received by the receivers to logs file located in the checkpoint directory.
- It can be enabled through spark.streaming.receiver.writeAheadLog.enable property.
- WAL can help to prevent against data loss.
- Once the WAL is activated, cache level should make a replication.
- Additional condition is the reliability of receiver. It should acknowledge data reception only after be sure to save it into ahead logs.

Demo - 1

Spark Streaming class -
StreamingContext class explore and perform a count operation on the number of words in text data .

Demo - 2

Output modes for Spark Streaming

- Append mode
- Complete mode
- Update mode

a) Append OutputMode in which only the new rows in the streaming dataframe/dataset will be written to the sink.
b) Complete Output mode in which all the rows in the streaming dataframe/dataset will be written to the sink every time when there're some updates.

Use complete mode as output mode when we want to aggregate the data and the entire output results to the sink every time. This mode is used only when you have streaming aggregated data.
e.g. in the sample wordcount job, counting the words on streaming data and aggregating with previous data then provide the combined output to the sink. Works on aggregations and in the context of non support of streaming aggregation scenario, of streaming dataset/dataframe, exception will be thrown.

c) Update Output mode - in which only the rows that were updated in the streaming Dataframe/dataset will be written to the sink every time when there're some updates.

Similar to complete output mode, only exception - just outputs the updated aggregated results every time to data sink when new data arrives. But not the entire aggregated result set like complete mode. If the streaming data is not aggregated , then it'll act as append mode.

Demo - 3

Reading the streaming data from directory (HDFS, local disk, cloud based data store - s3/blob etc.)

- In Spark Structured Streaming, from Spark v2.3 onwards, introduced a new low-latency processing mode called as Continuous Processing which can achieve end-to-end latencies as low as 1 millisecond with at least once guarantees.

Without changing the dataset/dataframe operations in the queries, we can choose the mode based on the app requirement.

Python Programming

08 September 2022 18:48

IDE

1. VS code
2. IDLE
3. Spyder
4. PyCharm
5. Atom
6. IntelliJ
7. Jupyter notebook/ Anaconda (Linux/ Windows)

Features of Python

- High level programming language
- Allows programming in object-oriented and procedural paradigms
- Indentation is required to be maintained
- Huge collection of standard libraries can be used in the areas of
- Machine Learning
- GUI apps (Kivy, Tkinter, PyQt etc.)
- Web frameworks (flask and Django)
- Image processing (OpenCV etc.)
- Web scraping (Selenium etc.)
- Scientific research

Python 2.x and Python 3.x
Pyscript (run in browser like javascript)

Core Features of Python

- Dynamic , high-level and free open-source language
- Both object oriented and procedural - oriented programming
- Free and open source
- extensible feature
- Easy to debug
- Integrated language - interpreted language because code is executed line by line at a time. There's no need to compile python code which makes it easier to debug the code.
- The source code of Python is converted to an intermediate form called bytecode.
- regular expressions, unit-testing etc
- Dynamically typed language
- Allocate memory dynamically

New Features of Python 3.8

- Positional-only parameters
- "/"

- Assignment Expressions :=
- f-strings now supports "="
- Reversed works with a dictionary reversed()
- Pow() function
- Importlib_metadata

a) VS code
b) Python 3
c) VS extension for Python

- Activate Virtual Environment

1. Py -3 -m venv .venv
2. Select the interpreter in VS code for the new venv
3. Python -m pip install matplotlib (conda install matplotlib)

Linear equation

X + 3y = 8
2x + 4y = 16

Linear Regression

|| (W1 Xi + W2) - Yi || 2

N pairs of data suppose to be given , in linear regression can be defined (xi, yi) , the parameters can be defined are w1, w2 which can minimize the error

Connect SQL Server db with Python (pyodbc and pymssql)

Pre-requisites:

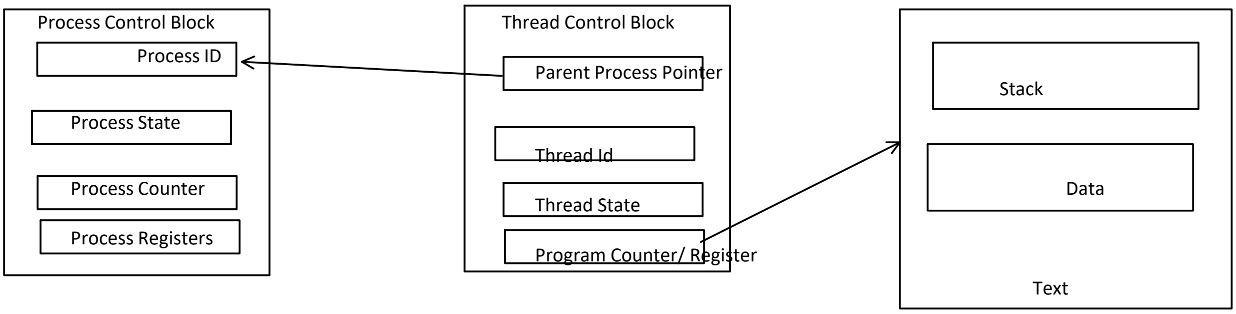
1. Install Python (3.9 >=)
2. Install Python SQL ODBC driver (https://learn.microsoft.com/en-us/sql/connect/python/pyodbc/python-sql-driver-pyodbc?view=sql-server-ver16)
3. Run the pyodbc from command prompt as admin mode
   pip install pyodbc
   Pip install pymssql (CRUD ops from py app)

# update your pip installer (test)

Pip install gensim / pyodbc (if getting failed)

# update the pip installer
1. Python -m pip install - upgrade pip
2. Pip -V or pip --version
3. Pip install - upgrade wheel
4. Pip install - upgrade setuptools

Threading Concepts in Python



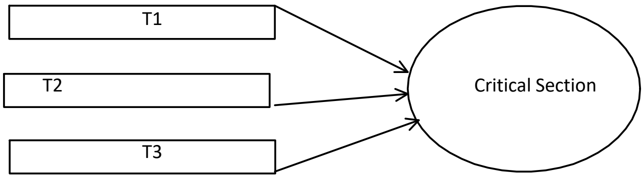
-Thread ID
-Stack Pointer
-Thread State(start, running, ready, waiting, done)
-Thread Register set
-Parent Process Pointer

Multithreading is the process / ability in which multiple threads can execute concurrently.

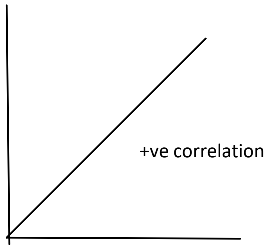
Thread Synchronization

It is a process in which two or more concurrent threads do not concurrently execute by critical section of program.

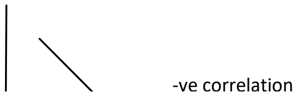
Mutual exclusive condition (deadlock)



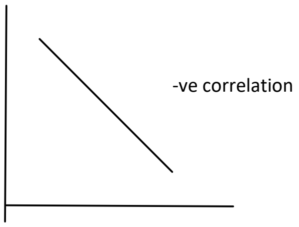
1. When two variables increase together and decrease together. They are called as positively correlated.



2. When one variable increases and other variable decreases together and vice-versa. They are negatively correlated. (-1 is there 's no correlation)



together and vice-versa. They are negatively correlated. (-1 is there 's no correlation)



3. Zero/no correlation - when there's no relationship exists between the Two variables at all. 0 is the perfect example of no correlation.

Demo -2
Analysis on Students graduation dataset

1. Different types of pandas plots
2. Overview of dataset through an histogram
3. Correlation with a scatter plot
4. Analyse different categories and their ratios.

Plotting of pandas plot

1. Median - median earnings of full-time, year round workers.
2. P25th - 25 percentile of earnings
3. P75th - 75th percentile of earnings
4. Rank - major rank by median earnings.

Azure Databricks

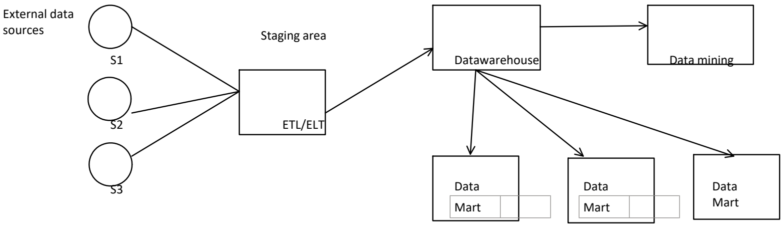
08 September 2022 18:49

Azure HDInsight	Azure Databricks
1. HDI is the Apache hadoop managed platform on Azure including other hadoop core components like Hive, Hive LLAP, Spark, Storm, Kafka, R etc.	1. The databricks platform provides around five times more performance than open-source apache spark.
2. Cost-effective and provides the benefits to optimize the data	2. Azure Databricks is the best choice for enterprise running Azure Cloud services as this is the spark-based analytics platform specially optimized for Azure.
3. Support private Link, VNET, logging and monitoring, Azure Log analytics integration etc.	3. Provides unified analytics platform (Lakehouse platform) makes the power of spark accessible. It's highly adaptable solutions for data engineering, data science and AI.
4. Integration with Azure AD, AD domain services are possible.	4. Works on premium Spark cluster. This one is faster than the open source Spark. Azure Databricks is a PaaS solution. It doesn't require a lot of admin efforts after the initial deployment. It's providing the core security through Azure AD without any requirement of the custom configuration.
5. Azure HDI provides the most popular OSS frameworks (Kafka, hadoop, hive, pig, hbase, oozie, sqoop etc.) which are easily accessible from the portal. If we need a combination of multiple clusters : Kafka on HDI for the streaming and Hive for interactive query purpose.	5. Databricks can offer the integration on the followings : - Delta lake - ML Flow - Lakehouse - Data engineering platform - Data streaming platform - Data Science and AI platform
6. Per cluster time based pricing	6. Per cluster time (VM cost + DBU processing time)
7. Apache Hive or Apache Spark	7. Apache Spark but optimized for Databricks, performance 5 times more than HDI
8. Ambari portal	8. Databricks notebooks, R studio
9. Hive QL, open source	9. R, python, scala, SQL, Julia etc.
10. Not scalable , requires cluster shutdown (premium tier) to resize.	10. Easy to change machines and allows auto-scaling

Azure Synapse And Databricks

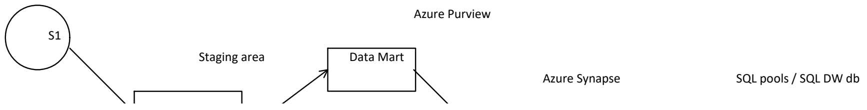
Azure Synapse Analytics	Azure Databricks
Azure Synapse is a limitless analytics service which brings together data warehousing and Big Data analytics. It gives the freedom to query data on your teams, either through dedicated or serverless SQL pools - at global scale. Azure Synapse brings together with a unified experience to ingest, prepare, manage and serve data for BI and ML engineers.	Azure Databricks is an Apache Spark based analytics platform which is used for data transformation, processing and end to end analytics purpose. Databricks can provide the benefits of integration of data engineering(transformation), data science models development and ML integration on managed spark env.
Synapse is based on analytical data warehouse - OLAP based models, interacts with data mines, data marts and DWs	Integrated with Azure can help to build optimized ML algorithms, (tensorflow, Pytorch, keras) , Mlflow can be executed on Databricks Spark cluster along with the benefits of pyspark interactivity, GPU based clusters Integration with git + CI/CD pipelines
No full git integration + CI/CD pipeline support, no multiuser notebook integration support available, As a data warehouse platform, we can ingest real-time data into Synapse using Stream analytics but doesn't support delta lake, ML flow, Synapse does not have full focus on real time data transformation yet.	Real time data transformation - preferred Databricks. - Spark structured streaming as part of databricks is proven to work seamlessly with delta lake, running optimized ML models. - Autoloader (allows the cluster to grow incrementally)
SQL analysis % data warehousing - Preferred Synapse A data warehouse with full capacity of SQL and DW capability with top-down approach. We can utilize PBI directly from Synapse studio The SQL pool (dedicated/ serverless) is the leader in the enterprise data warehousing	It's not possible with the full capacity of SQL and data warehousing capacity in a traditional data warehouse benefits in Databricks. Databricks doesn't have full t-SQL language support (Spark SQL)

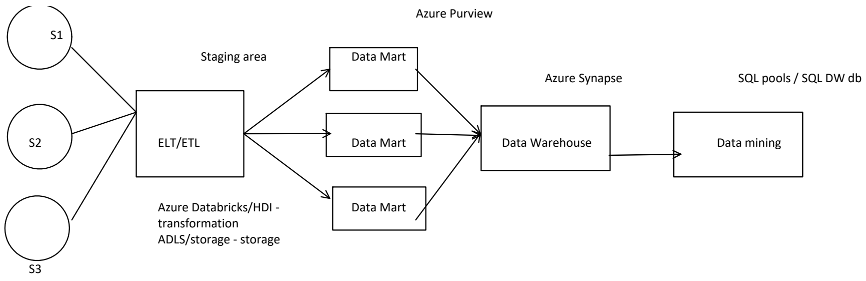
Top-down and Bottom-up approach



- Since the data marts are created from the data warehouse, provides consistent dimensional view of the data marts.
- Also, this model is considered as strongest model for business changes
- Creation of data mart from data warehouse is easier

Bottom - up approach (preferred)

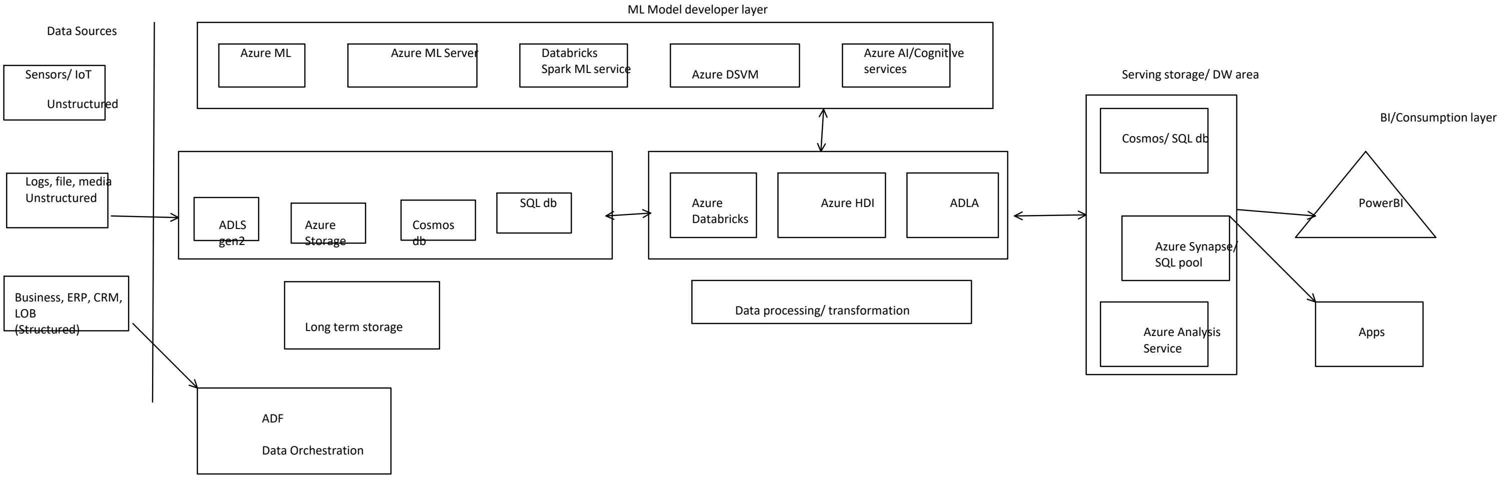




- First the data is extracted from the external data sources
- Then the data is moved to the staging area for data transformation and loading into the data marts , these data marts are created first and provide the reporting capability.
- Then these data marts are integrated into the data ware house.

Benefits
- As data marts are created first, so the reports can be generated quickly
- Accommodate more numbers of data marts and extension of DW is possible
- Cost and time effective.

Analytics solution architecture considerations



Databricks workspace

A workspace is an environment for accessing all of your databricks assets.

- A workspace organizes the objects (notebook, libraries, dashboards, experiments) into the folders and provides access to the data objects and computation resources.

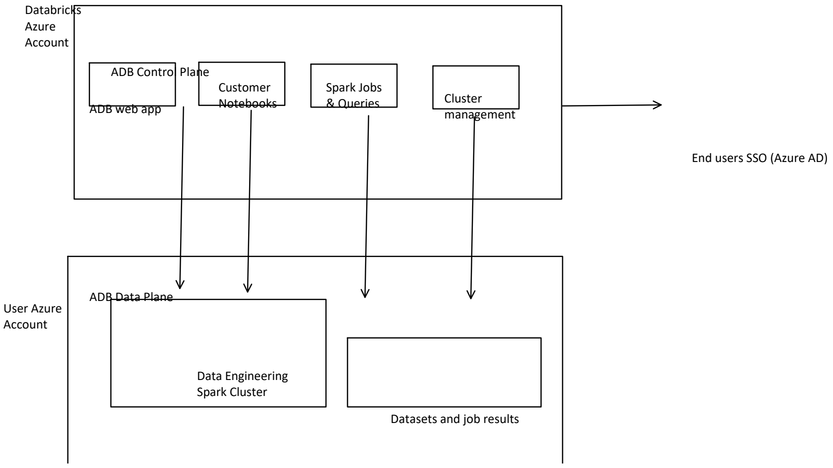
Resources of workspace

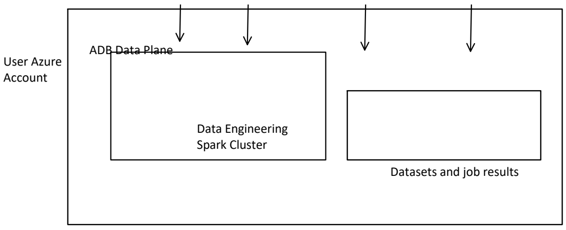
1. Notebook - a web-based interface to documents which can execute commands, visuals and text
2. Dashboard - an interface providing organized access to visuals
3. Library - package of code available to notebook or job running on the cluster. ADB runtimes can include many libraries and add customized libraries.
4. Repo - A folder whose contents are co-versioned together by syncing them to a remote Git repo.
5. experiment - a folder contains data sync to git repo and collection of ML flow models.

Demo 1 : Databricks

a) Create Spark Cluster in Databricks
b) Import the sample data from public blob storage account
c) Run the Spark SQL job in the Spark notebook

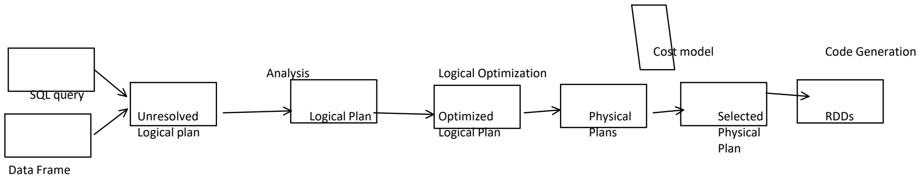
Azure Databricks Control Plane and Data Plane





- ADB control plane includes the backend services that Azure Databricks manages in own Azure account. All notebook commands and many other workspace configurations are stored in the control plane and encrypted at rest.
- The data plane is managed by the customer's azure account and the data also resides in the customer's azure account. This is the place where data is getting processed. We can use ADB connectors so that the cluster can connect to external data sources outside of Azure account to ingest the data or for storage.
- Ingest data from external streaming data sources, such as events data, streaming data / IoT data.

Spark Catalyst Optimizer



- At the core of Spark SQL is catalyst optimizer. It's responsible to leverage advanced features (scala's pattern matching) to build extensible query optimizer.
- Catalyst is based on functional programming constructs like in Scala.

Benefits

a) Easily add new optimization techniques and features to Spark SQL
b) Enable external developers to extend the optimizer (e.g. add new data source specific rules, support for new data types etc.)

c) Catalyst contains a general library for representing trees for applying rules to manipulate them.
d) It has the benefit specific to relational query processing (expressions, logical query plans) and several rules to handle different phases of query execution - analysis, logical optimization, physical planning and code generation to compile parts of queries to java bytecode.
e) Several public extension points including the internal data sources and user-defined types.

Azure Databricks - Demo 2:

Spark SQL - importing sample dashboards to ADB Spark SQL

Databricks Authentication and Authorization

1. Azure AD User - A unique individual who has access to the system. User identities are being represented by Azure AD groups, access policies and identities.
2. Service Principal - A service identity for use with jobs, automated tools, and systems such as scripts, apps and CI/CD platforms. Service principals are represented by the app Id.
3. Azure AD Group - A collection of identities in Azure AD. Groups can simplify identity management, making it easier assign access to workspaces, data and other scalable object. All Databricks identities can be assigned as members of groups.
4. Access Control List (ACL) - A list of permissions attached to the workspace, job, cluster, table and experiment. An ACL specifies which users or system processes are granted access to the objects, as well as what operations are allowed on the assets.
- Each entry in a typical ACL specifies a subject and an operation.
5. Personal Access Token (PAT) - An opaque string is used to authenticate to REST API tools for integration with Databricks workspace to connect to Azure SQL DW/ Synapse analytics.

Data Management Components in ADB

(For Data Science and Data Engineering interface)

- UI (The databricks UI provides an easy-to-use GUI to workspace folders and their contained objects, data objects and computation resoirces.
- REST API - There're three versions of REST API support available (REST API 2.1, REST API 2.0 and REST API 1.2) .

Data Management Components:

a) Databricks File System (DBFS) - A filesystem abstraction layer over the blob storage/ ADLS gen2 storage. It contains directories, can contain files (data files, libraries and images) and other directories. DBFS is automatically populated with some datasets which can be explored.
b) Database - A collection of info which can organized so that it can easily be accessed, managed and updated.
c) Table - A representation of structured data. We can query the tables with Spark SQL and Spark APIs.
d) Hive Metastore - The component which stores all the structures of information like various tables, and their partitions information in the data warehouse including columns, column type information, the serializers and deseralizers necessary to read and write the data, the corresponding files where the data is stored. Each ADB deployment has a central hive metastore repo accessible by all Spark clusters to persist the table metadata.

Computation Management components in ADB:

1. Cluster - A set of computation resources and configurations on which we can execute notebooks and jobs. There're two types of clusters
a) All-purpose cluster - create it via UI, CLI, REST API. We can manually terminate and restart the all-purpose cluster. Multiple users can share such clusters to do collaborative interactive analysis.

b) Job cluster - In Databricks, the job scheduler creates a job cluster where we can run a new job cluster and terminate the cluster while the job is completed. A job cluster cant be restarted.

2. Cluster Pool - A set of idle, ready to use instances what induce the cluster start and autoscaling times. When it's attached to a pool, a cluster allocates its driver, worker nodes from the pool. If

the pool does not have sufficient idle resources to accommodate the cluster's request, the pool expands by allocating to new instances from the instance provider. When an attached cluster is terminated, the instances are used being returned to the pool and can be reused by another different cluster.

3. Databricks runtime - The set of core components which can execute on the cluster managed by Databricks. It offers several types of runtimes.

- Databricks Runtime - includes Apache Spark but also adds a number of components and updates which substantially improve the usability, performance and security of data analytics.

- Databricks Runtime for Machine Learning - is built on Databricks Runtime and provides a ready-to-go environment for ML and Data science. It contains multiple popular libraries , including tensorflow, Keras, Pytorch, XGBoost.

4. Workflows - Frameworks which can develop and execute the data processing pipelines.

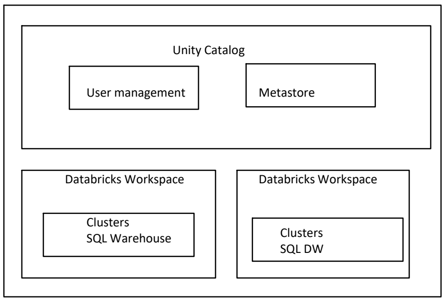
- Create, run and manage ADB jobs - A non-interactive mechanism for running a notebook or library either immediately or on a scheduled basis.

- Delta Live table introduction - A framework for building reliable, maintainable and testable data processing pipelines.

5. Workload - Databricks identifies two kinds of workload subjected to pricing schemas.
- Data Engineering - An automated workload runs on a job cluster which the ADB job scheduler creates for each workload.
- Data Analytics - An interactive workload runs on all-purpose cluster. Interactive workloads typically run commands within a ADB notebook.
- Running a job on an existing all-purpose cluster is also treated as interactive workload.

6. Execution Context - The State of execution environment for each supported programming, Python, Scala, R and Java.

Unity Catalog in Azure Databricks



In unity catalog, admins and data engineers can manage users
And access to the data centrally across all of the workspaces in an Azure Databricks account.

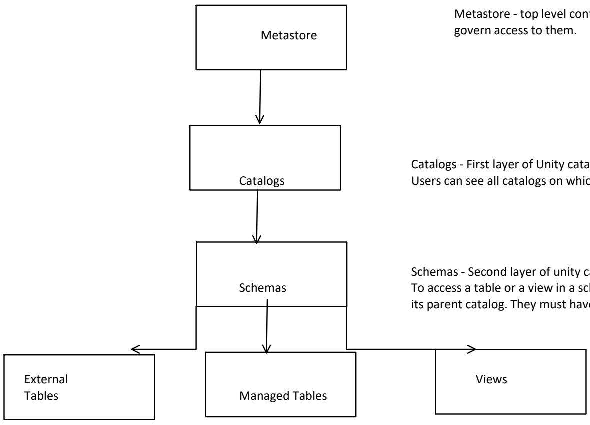
Users in different workspaces can share access to the same data, depending on the privileges granted centrally in Unity catalog.

- Define once and secure everywhere - Unity catalog offers a single place to administer data access policies which apply across all workspaces and personas

- Standards - compliant security model - Unity catalog security model is based on standard ANSI SQL allows administrators to grant permissions based on their existing data lake, at the level of catalogs, databases, schemas and views.

- Built-in auditing - Unity catalog automatically captures db, catalogs, tables and views. It can capture user level audit logs which record access to the data.

The Unity Catalog Object Model



Metastore - top level container of objects in Unity Catalog. Stores the data assets and permissions which govern access to them.

Catalogs - First layer of Unity catalog's three level namespace. It is used to organize the data assets. Users can see all catalogs on which they have been assigned to the USAGE data permission.

Schemas - Second layer of unity catalog's three level namespace. A schema organizes tables and views. To access a table or a view in a schema, users must have the USAGE data permission on the schema and its parent catalog. They must have the SELECT permission on the table/view.

A table resides on the third layer of unity catalog. Rows and attributes of the data.

- Managed table - stored in the root storage location("/DBFS") when we create the metastore. We can use the Delta table format.
- External table - External tables are tables whose data can be stored after the deletion, data is stored outside of root storage location. We require direct access to the data using other tools.

- Delta
- CSV
- JSON
- AVRO
- PARQUET
- ORC
- TEXT

Configure Unity Catalog of ADB

- Configure storage container and Azure managed identity that unity catalog use it to store and access data in the Azure account.
- Create a metastore for each region in which each org operates, attach workspaces to the metastore. Each workspace will have the same view of the data can managed in the Unity catalog.
- If there's a new account, users, groups and service principals can be added to the Azure Databricks account.

- Demo 1 :

a) Import sample dashboards to Databricks SQL

1. ADB workspace is required to be created in premium plan
2. Explore the Databricks Spark SQL environment
3. Explore Databricks SQL dashboards

- Lab 2:

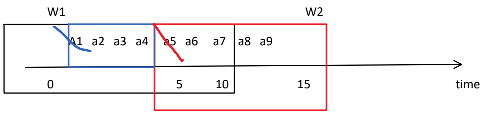
1. Data Transformation with Databricks Spark SQL

- Advanced Dataframe options

maxFilesPerTrigger - maximum number of new files to be considered in every spark streaming trigger.

Sliding Window - data streams are grouped within a window that slides across the data stream according to the time interval. A time based sliding window with a length of 10 seconds contains the streams within a 10th sec window. The set of streams / data within the window are evaluated in every five seconds.

Sliding window can contain overlapping data.



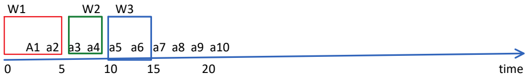
The common events are a5, a6 , a7 are common between both of these Window W1 and W2.
A5, a6, a7

W2 is evaluated in t=15 sec,

The events i.e. A1, a2, a3, a4 are dropped from the event queue.

Moving average of a stock price across last 5 sec
Collecting tweets over a # over last 10 sec

Tumbling Window - events are grouped in a single window based on time or count. An event belongs to one window.



None of windows can never overlap
Each segment represents a distinct time segment

e.g. compute the average price of a stock over last 5 min which is computed in every 5 min

Watermark - it's a feature which allows users to specify the threshold of the threshold of late arrived data. Accordingly it allows the streaming engine to clean up the old data streams.

Personal Access Tokens (PAT)

In order to authenticate to the ADB REST API, a user can create a personal access token and use it in their REST API request. Tokens have an optional expiration date and can be revoked.

- The number of PAT tokens can be created per user is limited to 600 / workspace
- The workspace administrator can enable to disable the ability to create the PAT token access from any workspace, regardless of when the workspace was created.
- With the ability to generate PAT tokens enabled for the workspace, all users in the ADB workspace can generate personal access token to access the ADB REST API and they can generate these tokens with any expiration date they like including indefinite lifetime.

Conditional Access

Azure Databricks supports Azure AD feature of conditional access which allows administrators to control where and when users are permitted to sign to databricks.

- Have Azure AD premium plan P1/ P2 plan enabled.
- Must be a role of global administrator / conditional access admin

DBFS Root

The DBFS root is the default storage location for a Databricks workspace, provisioned as part of a workspace creation in the cloud / Azure account containing the workspace.

Best Practices

- ADB uses the DBFS root directory as a default location for some workspace actions.
- It recommends against storing any production data or sensitive info in the DBFS root.
- DBFS root is not accessible for all users in the workspace
- The default location for the managed tables in hive metastore is the DBFS root, to prevent the end users who create the managed tables from writing to the DBFS root, declare a location on external storage when creating databases in the hive metastore.
- encrypt the DBFS root data with a customer - managed key.

Delta Lake & Delta Table in Databricks

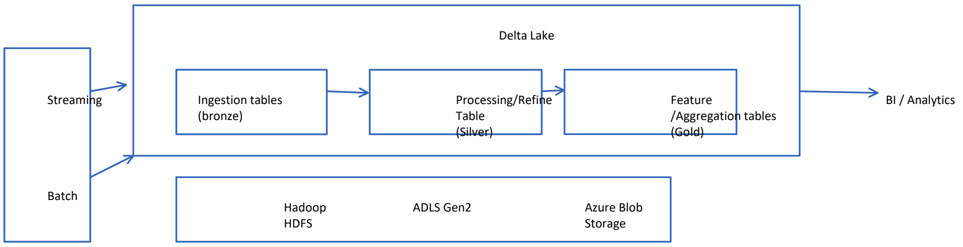
1. Delta Lake is the optimized storage layer which provides the foundation for storing data and tables in the Databricks lakehouse platform. Delta Lake is a OSS software which extends parquet data files with a file based transaction log for ACID transactions and scalable metadata handling.

2. Delta lake is fully compatible with Apache Spark APIs with integration support of Structured streaming, allows to easily use a single copy of data at both batch and streaming analytics and provides the incremental processing scale.

3. All of the tables on Azure Databricks are Delta tables. Databricks originally developed the Delta lake protocol and continues to actively contribute,

4. Atomic transactions with Delta lake can provide many options for updating data and metadata.
5. Delta lake supports upserts using the merge operation
6. Delta lake provides numerous option for selective overwrites based on filters and partitions.
7. We can manually or automatically update the table schema without rewriting the data
8. Column mapping enables columns to be renamed or deleted without rewriting data.

- Delta lake create Delta live tables
- COPY into (copy data from delta lake to tables)
- Incrementally and efficiently processes new data files as they arrive in cloud storage (S3, s3://, abfss://, gs://, wasbs:// , dbfs://)
- Autoloader can ingest the files like JSON, CSV, Parquet, binary, ORC, TEXT, AVRO etc.
- Schema enhancements (delta lake schema validation, constraints on ADB
- Manage files and indexing data with Delta lake


Delta Lake

Streaming

Batch

Ingestion tables (bronze)

Processing/Refine Table (Silver)

Feature /Aggregation tables (Gold)

BI / Analytics

Hadoop HDFS

ADLS Gen2

Azure Blob Storage

The existing data lake

- Storage abstraction layer on top of existing data lake
- Designed to work with DBFS and Spark
- Unified streaming and batch processing, ACID transactions, and scalable metadata handling support.
- Stores the data into parquet file formats (throughput, optimization, compression) and maintains a transaction logs which tracks changes to the tables.

Demo - 1

1. ETL job execution on Databricks (Configure the auto loader to ingest data into Delta lake)
2. Delta table , read the stream and transformation
3. Schedule ETL job in Databricks Spark streaming

Lakehouse

The Databricks Lakehouse combines the ACID transactions and data governance of data warehouses with the flexibility and cost-efficiency of data lakes to enable BI and ML on all data.

The Databricks Lakehouse keeps the data in the massively scalable cloud object storage in open source data standards allowing us to use data however and wherever it's required.

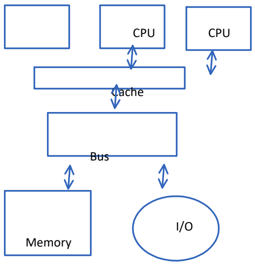
- Delta tables
  -ACID transaction
- Data versioning
- ETL
- Indexing

Delta table

Table which is created on ADB to use the Delta lake protocol by default. When we create a new Delta table,

- Metadata used to reference the table is added to the metastore in the declared schema or db
- Data and table metadata are saved to the directory in cloud object storage (S3/ abfs / wasb/ dbfs)
- A directory containing table data in the parquet file format
- A sub-directory /_delta_log which contains metadata about table versions in JSON and parquet format.

Difference Between SMP and MPP


CPU

CPU

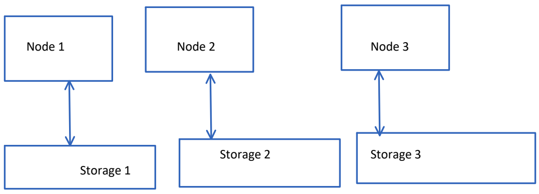
Cache

Bus

Memory

I/O

Symmetric Multi-processing	Massive Parallel Processing
Tightly coupled multi-processor system Where processors can share resources - single instances of OS, memory, I/O devices and connected using a common bus.	MPP is the co-ordinated processing of a single task by multiple processors. Each processor where using its own OS, memory and communicate with each other by using messaging interface. It's a loosely - coupled architecture
Primary parallel architecture employed client - server connection	MPP can be setup with a shared nothing or shared disk architecture
	Data is horizontally partitioned across nodes, each node has a subset of rows from each table in the db
	Each node then processes only the rows on its own disks.
	Systems based on this architecture can achieve massive scale as there's no single bottleneck to slow down the system


Node 1

Node 2

Node 3

Storage 1

Storage 2

Storage 3

MPP - Azure SQL DW , Synapse

Overview of AWS

08 September 2022 18:49

AWS Cloud Adoption Framework :

Well - architected framework:

1. **Operational excellence pillar** - focuses into running and monitoring systems, continuously improving processes and procedures. Automated changes , responding to events, and defining the standards to manage the daily operations.
   a) Perform operations as code - app development, platform engineering and policy development, infra deployment and devops

b) Make small changes/ reversible -
c) Fail fast -

2. **Cost-effective pillar** - focuses on avoiding unnecessary costs where we can utilize spot instances, pay per second basis, right type of instance and quantity

3. **Reliability Pillar** - focuses on the workload performing their intended functions. Recover from failure to meet demands.
4. **Performance efficient pillar** - focuses on the efficient use of computing resources to meet requirements, how to maintain efficiency as demand changes and tech stack evolve
5. **Sustainability pillar** - optimizing workloads and resource utilization, minimizes the total resource required to be deployed for the workload.
6. **Security pillar** - protecting the information and systems

Infrastructure Side :

a) **Compute**

1. EC2 (Elastic Compute Cloud) - (Azure VM)
2. EBS (Elastic Block Storage) - (Azure Disk)
3. EFS (Elastic file systems) - (Azure File storage)
4. Lambda (Serverless compute) - (Azure Function)

Application side:

1. AWS EBS - (Elastic Bean Stock) - This is the managed PaaS service for provisioning apps developed any standard programming with support of automated deployment, autoscaling , failover etc. - (Azure App Service)

b) Storage

1.S3 (Simple Storage Service) - (Azure Blob storage/ ADLS)

2. CDN (content delivery network) - (Azure CDN)

c) Database

1. Dynamodb (NOSQL db offering) - (Azure Cosmos db)
2. RDS (Managed relational db) - (Azure SQL db , sql managed instance, sql on azure vm)
3. RDS Aurora (managed high performance mysql db) - Azure database for mysql

d) Big Data Analytics

1.EMR - Elastic Map Reduce - (Azure HDI)

2. Redshift - (DW & MPP) - (Azure SQL DW / Synapse)
3. AWS Glue - (data orchestration and analytics pipeline) - Azure Data Factory
4. AWS Athena (real time streaming platform) - Azure Stream Analytics
5. AWS Kinesis (real time streaming data ingestion and partitioning service) - Azure Event Hub

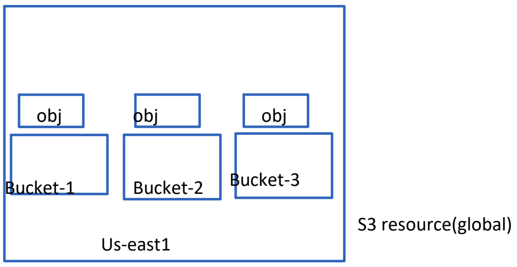
e) Network

1.AWS VPC (virtual private cloud) - Azure Virtual Network (Vnet)
2.AWS Subnet , route table , security groups - Azure subnet, route table and network security groups (NSG)

S3 buckets

Object storage built into store and retrieve any amount of data from anywhere.

- Create bucket, specify the region, access controls, management options, upload any amount of data.
- Control access to data
- Optimize cost with storage classes
- Replicate the data to any other region on S3 bucket
- Protect and secure the data



ACL permissions are being defined with granular access policies Into individual objects within the buckets of S3.

AWS RDS (Relational Database Service)

DB instances for AWS RDS can be provisioned for MySQL, PostgreSQL, Oracle, MSSQL using AWS elastic block store(EBS) volume for database and log storage.

RDS uses a logical database server on which the choosen db type is being provisioned. There's a VM instance is running behind the RDS database instance.

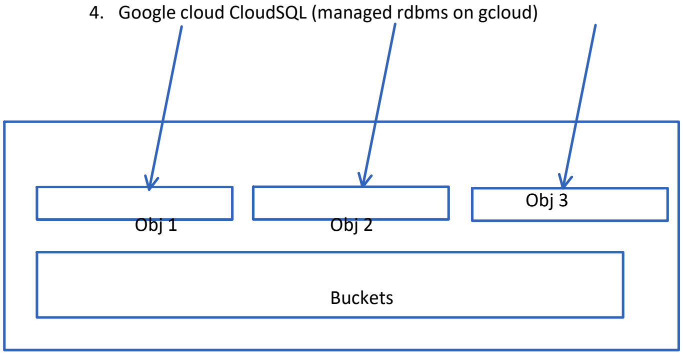
RDS is equivalent to SQL Server on Azure VM.

RDS is deployed on EC2 instance and uses the EBS volume for database and log storage.

Overview of GCP

08 September 2022 18:49

1. Google Cloud Compute Engine VM instance
2. Google Cloud Storage bucket
3. Google cloud VPC , subnet (IPv4 ranges)
4. Google cloud CloudSQL (managed rdbms on gcloud)



Obj 1

Obj 2

Obj 3

Buckets

Google Cloud Storage

Uniform

Fine-grained

User 1 - obj 1 - permission (r- rw - rwx)
User 1 - obj 3 - permission (r)

User 2 - obj 1 - permission (r - rw)
User 2 - obj 2 - permission (rw - r -rwx)