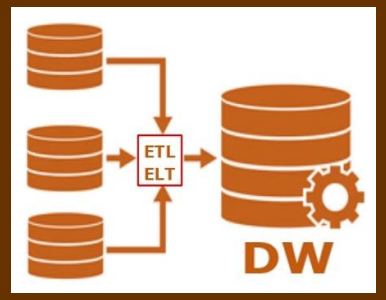


Data Warehousing

1. DATAWAREHOUSING

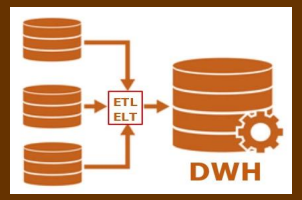
INTRODUCTION

Follow us on :

YouTube

Linked in

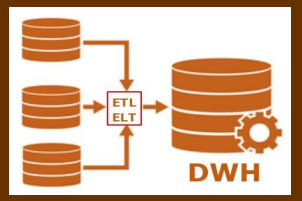
CDU
Cloud And Data Universe



What is a Data warehouse?

- Data warehouse is a centralized repository or place where data is stored.
- Abbreviations used are DW or DWH.
- Also known as Enterprise data warehouse (EDW)
- Data is loaded from one or more sources.
- DW stores historical and current data

Cloud And Data Universe

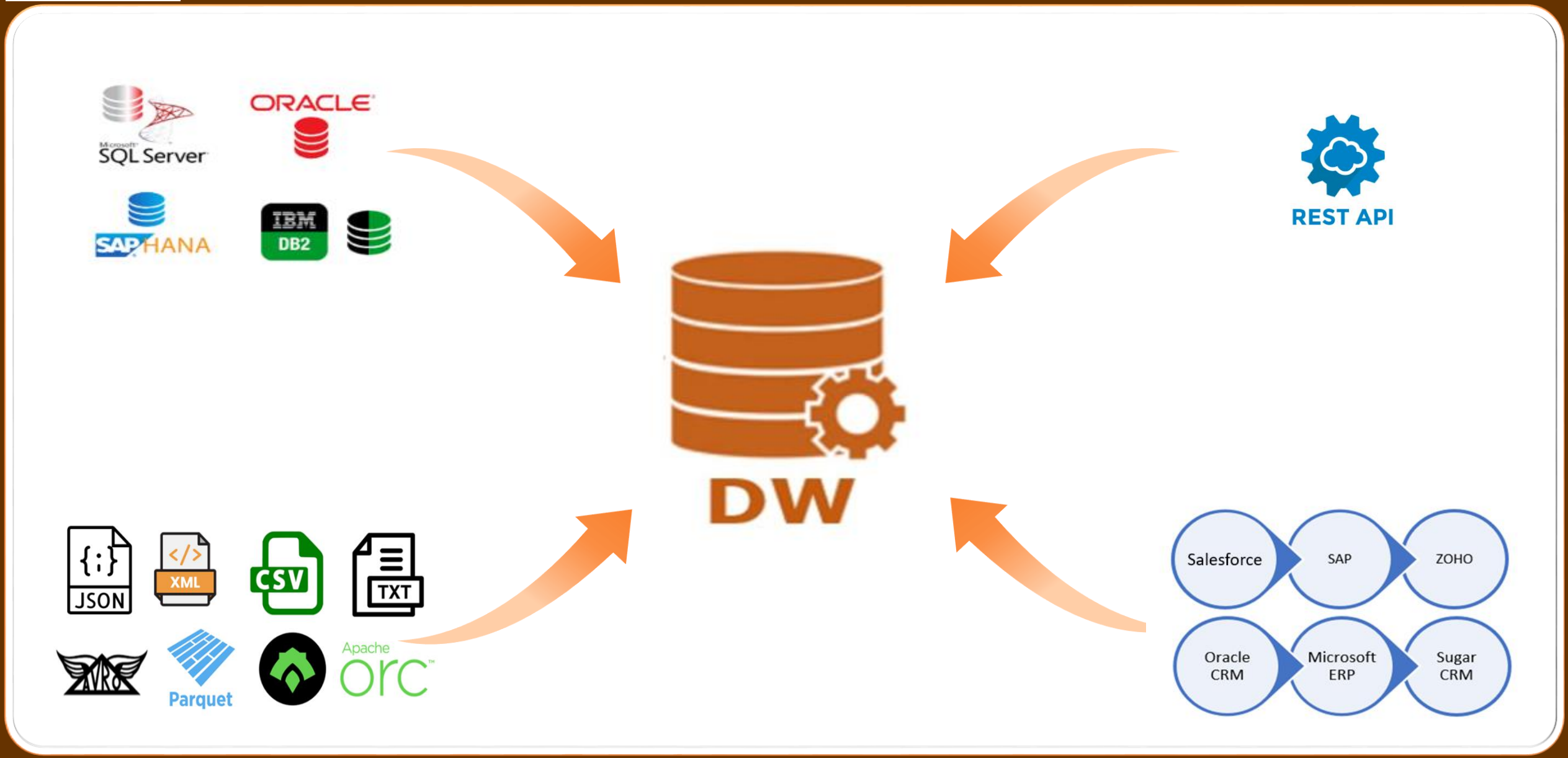


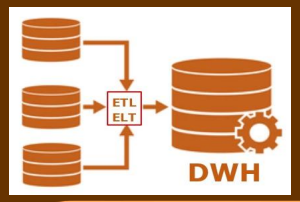
ETL/ELT

- ETL(Extract – Transform Load) / ELT(Extract – Load – Transform)
- Data load is performed by ETL process using different tools like SSIS, ADF, Informatica, etc.

CDU
Cloud And Data Universe

Data Warehouse Model





ETL/ELT Tools

CDU
Cloud And Data Universe

Azure Data Factory

Microsoft SQL Server Integration Services

HITACHI
Pentaho

talend
Talend Open Studio

Stitch
Stitch

Informatica
Informatica PowerCenter

ORACLE
Oracle Data Integrator

Fivetran
Fivetran

Airbyte
Airbyte

SINGER
Singer

integrate.io
Xplenty (Integrate.io)

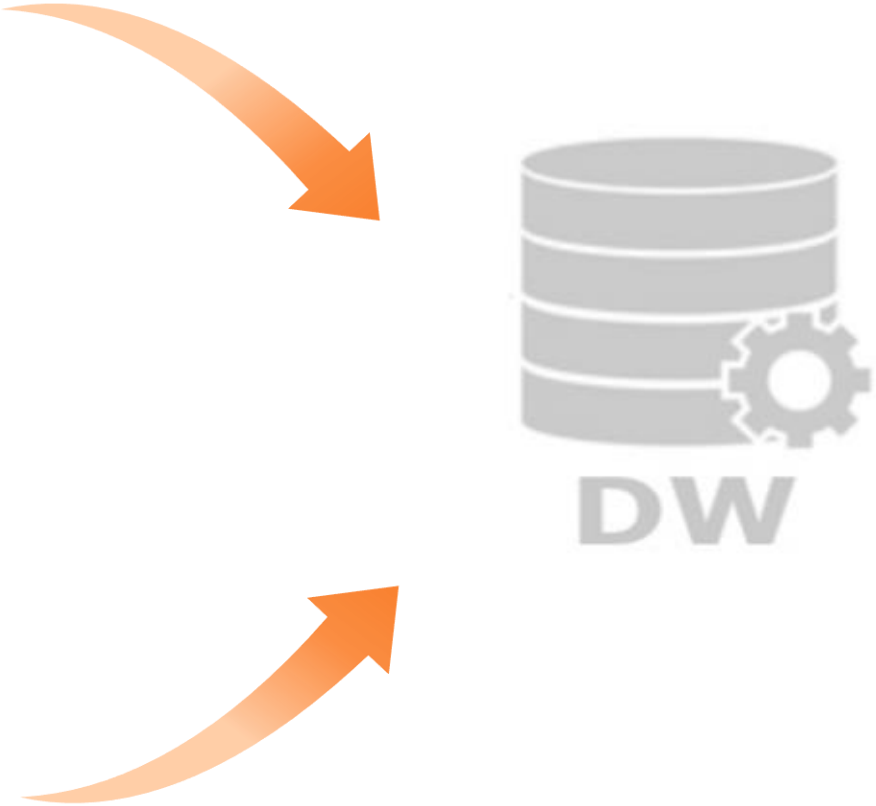
Dataddo

StreamSets

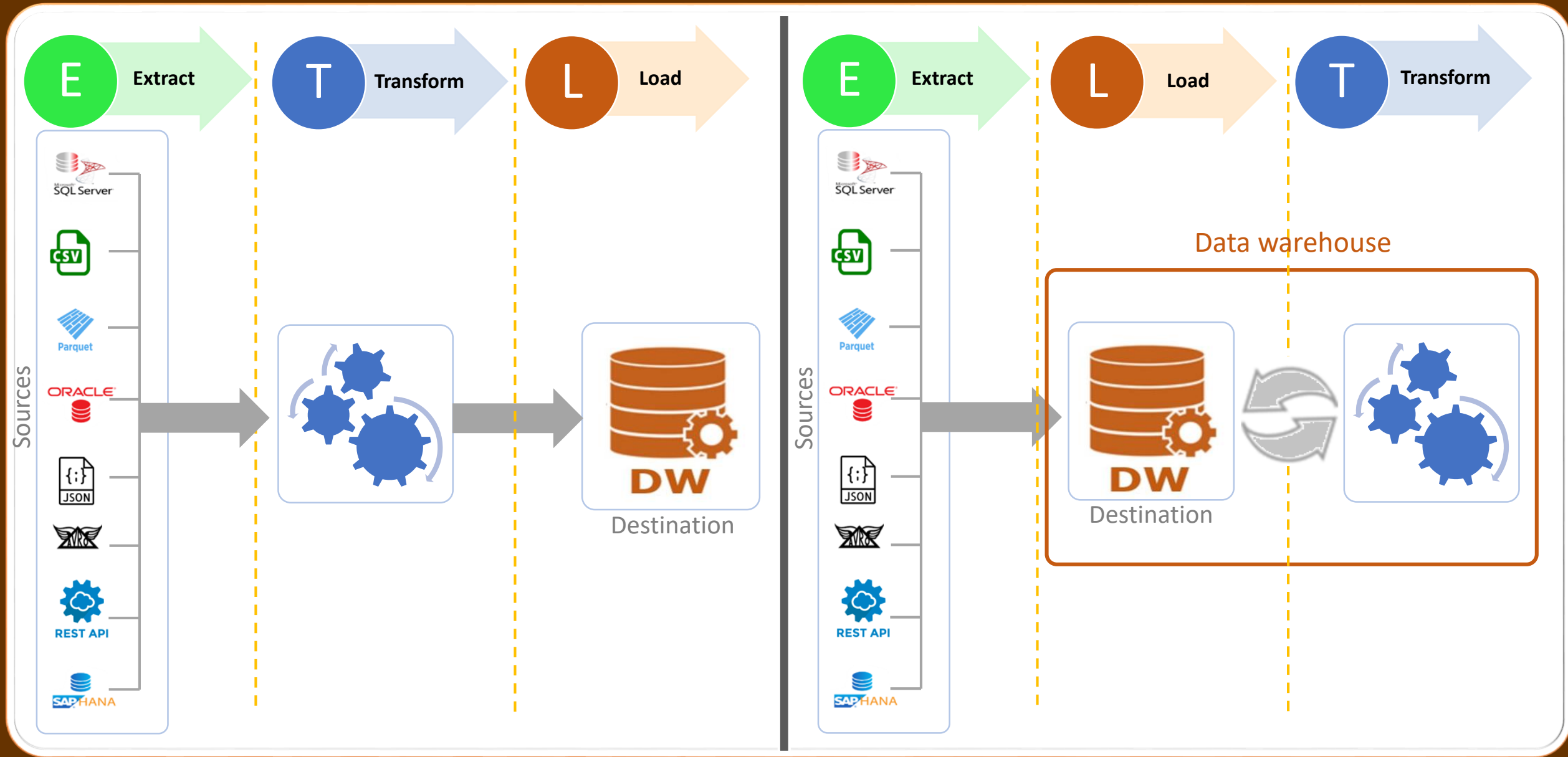
BLEND

IBM DataStage

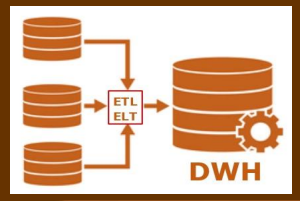
aws Glue



ETL/ELT Process



DW Tools



CDU
Cloud And Data Universe

Azure Data Factory

Microsoft SQL Server Integration Services

HITACHI Pentaho

talend Talend Open Studio

Stitch Stitch

Informatica Informatica PowerCenter

ORACLE Oracle Data Integrator

Fivetran Fivetran

Airbyte Airbyte

SINGER Singer

integrate.io Xplenty (Integrate.io)

Dataddo

StreamSets

BLEND

IBM DataStage

aws Glue

amazon REDSHIFT

PostgreSQL

Azure Synapse Analytics

IBM DB2 IBM Cloud Db2 Warehouse on Cloud

DW

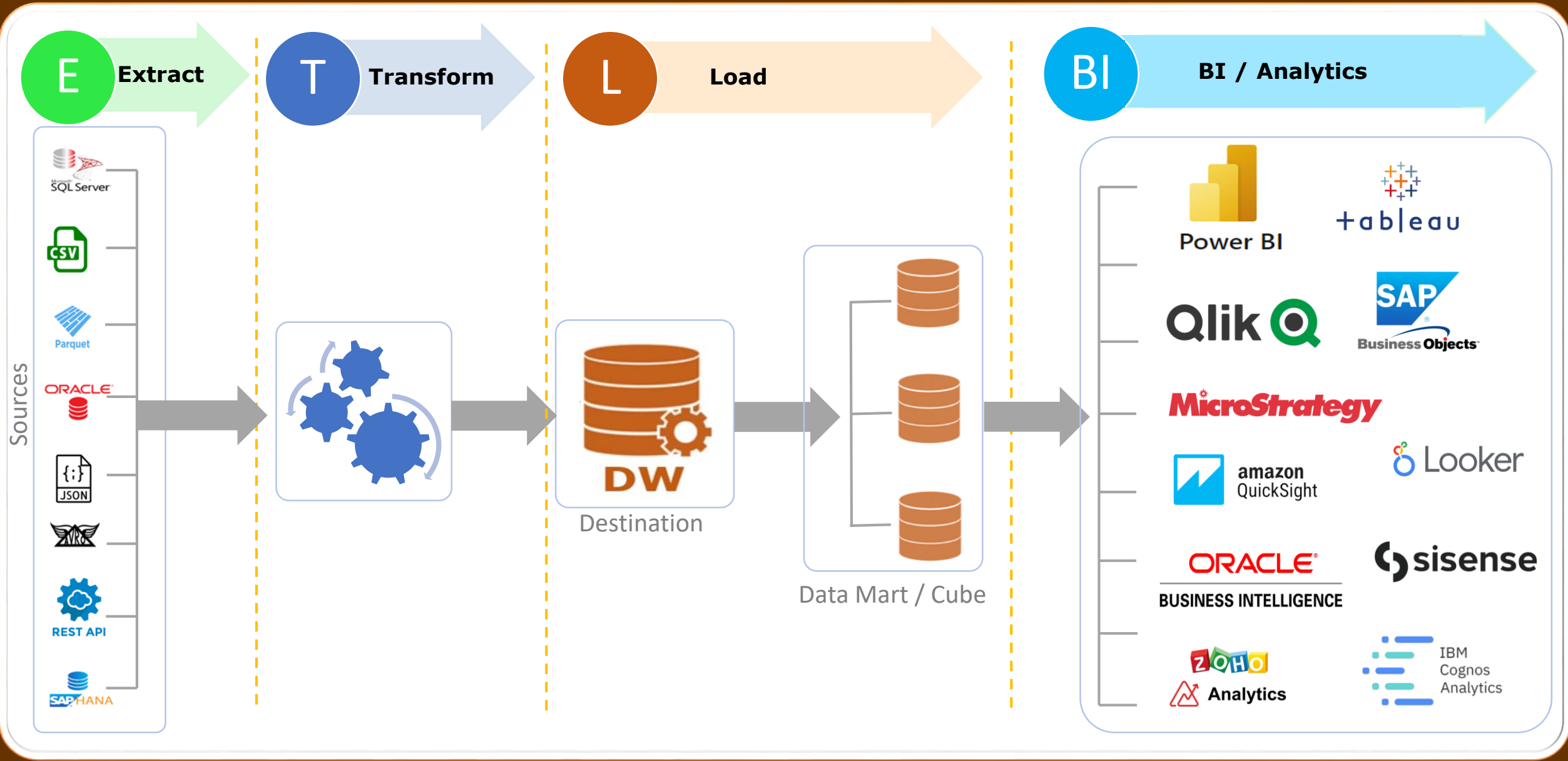
ORACLE DATA WAREHOUSE

snowflake

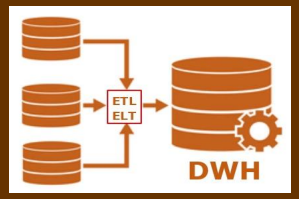
TERADATA

Google Big Query

Datawarehouse & Analytics Architecture

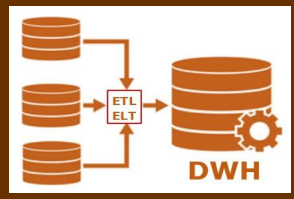


Data Model



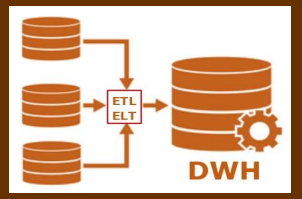
- Data Model typically consists of multiple tables.
- These tables are called Fact and Dimensions.
- These tables are related to each other by relationship between them.
- Mostly you will see more Dimension tables and fewer Fact tables.

Dimension Tables



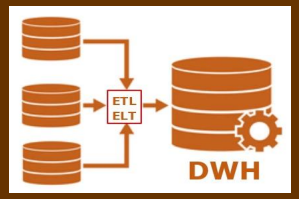
- Dimension tables are mostly designed per entity wise. For instance, you will see one dimension table each for products, customers, suppliers, date, vendors, employees, etc.
- In these tables you will have a primary key which acts as record identifier.
- Dimension tables can also be called as master tables.

When will be a new row/record added in dimension table?



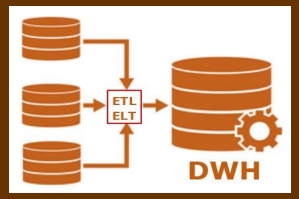
- Consider an Employees table: A new record will only be added when a new employee joins an organization.
- Same is the case with other dimension tables.
- Hence, you will see data in Dimension tables will not be appended/inserted frequently as compared to Fact tables

Fact Tables



- Fact tables are transaction tables.
- Consider a sales tables: A new row/record will be inserted for every single sales transaction. Hence, data will be appended/inserted very often as compared to Dimension table.

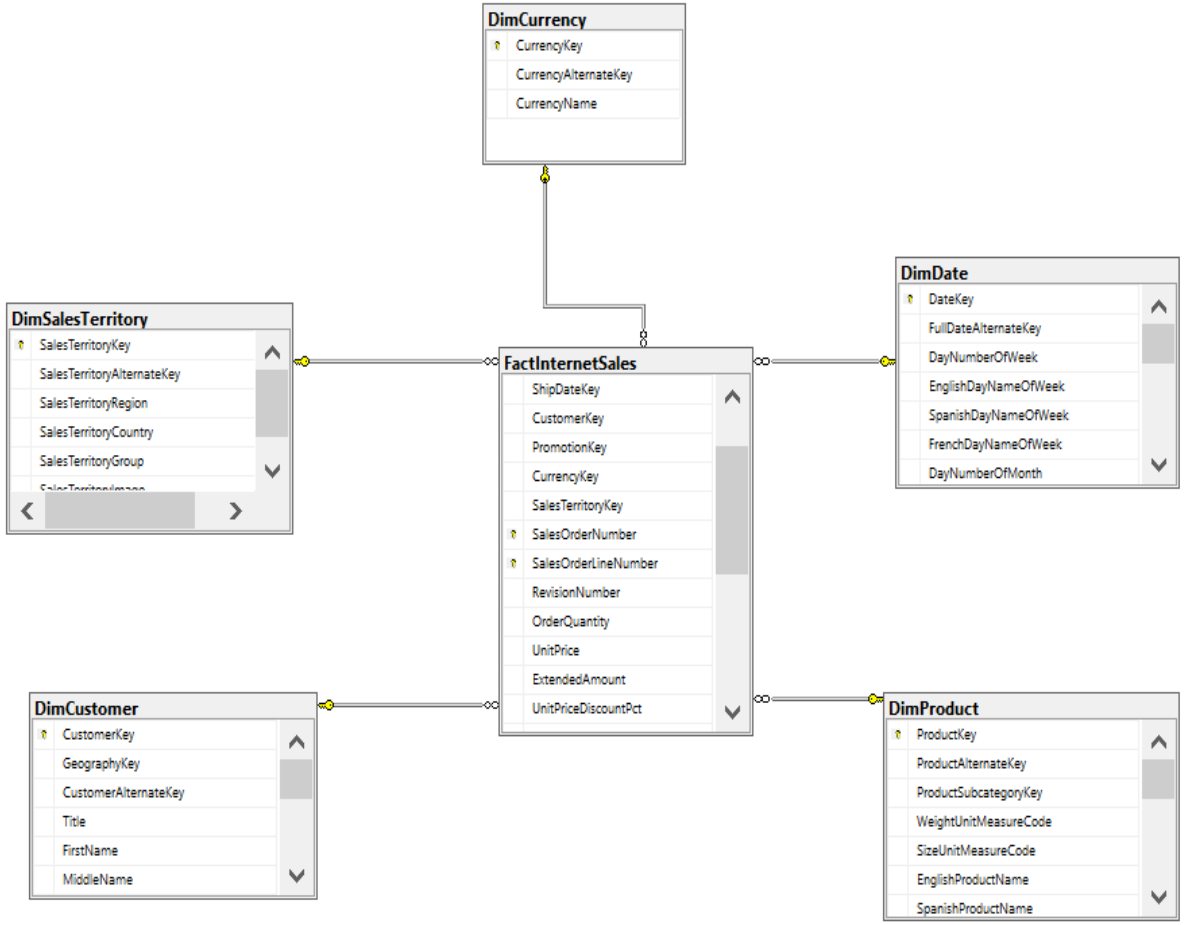
Fact Tables



- Fact tables contain foreign key columns which are coupled with primary key columns in Dimension tables to maintain referential integrity.

Star Schema

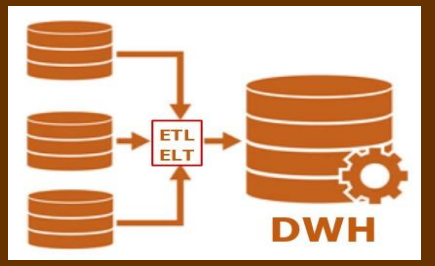
Star schema is a data model in which Dimension tables are directly connected to Fact table



Snowflake Schema

Snowflake schema is a data model in which Dimension tables may or may not be directly connected to Fact table

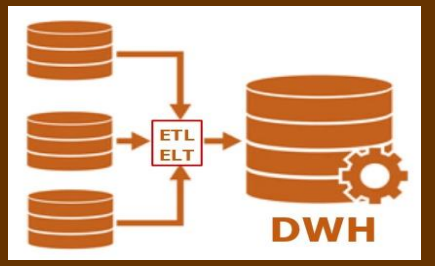
OLTP & OLAP



- OLTP stands for Online Transaction Processing

- OLAP stands for Online Analytical Processing

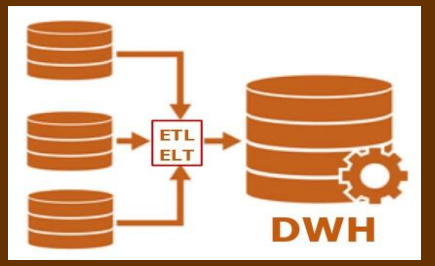
Differences between OLTP & OLAP



Period

OLTP	OLAP
Deals with most recent data	Deals with historic data

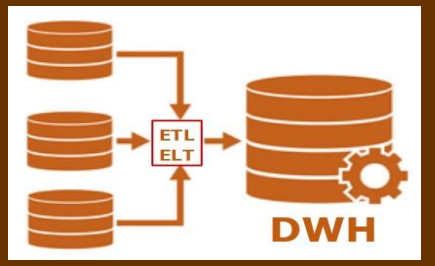
Differences between OLTP & OLAP



Processing

OLTP	OLAP
Processes small to large number of small transactions	Processes massive volume of data

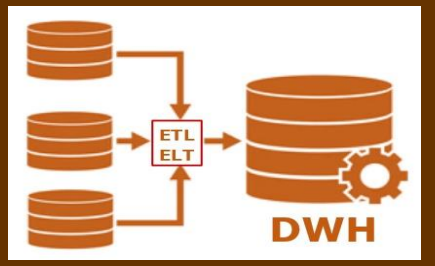
Differences between OLTP & OLAP



Queries

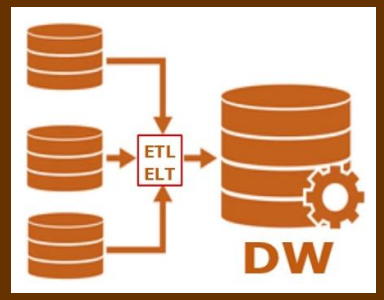
OLTP	OLAP
Simple queries are used	Complex queries are used
Uses INSERT, UPDATE & DELETE	Mostly SELECT only

Differences between OLTP & OLAP



Learning

OLTP	OLAP
Easy to learn	Need to invest time, doesn't come quickly



2. DATAWAREHOUSING

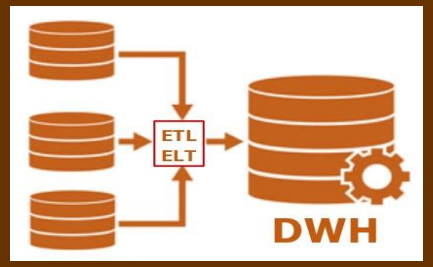
DATA LOADS

Follow us on :

YouTube

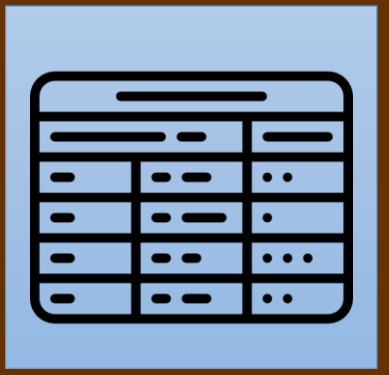
Linked in

CDU
Cloud And Data Universe

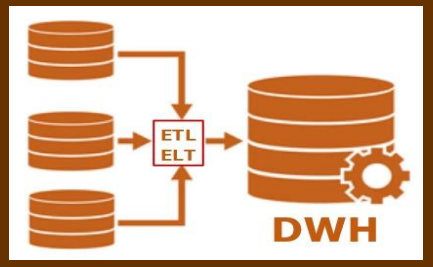


1. Full load

- Truncate & Load

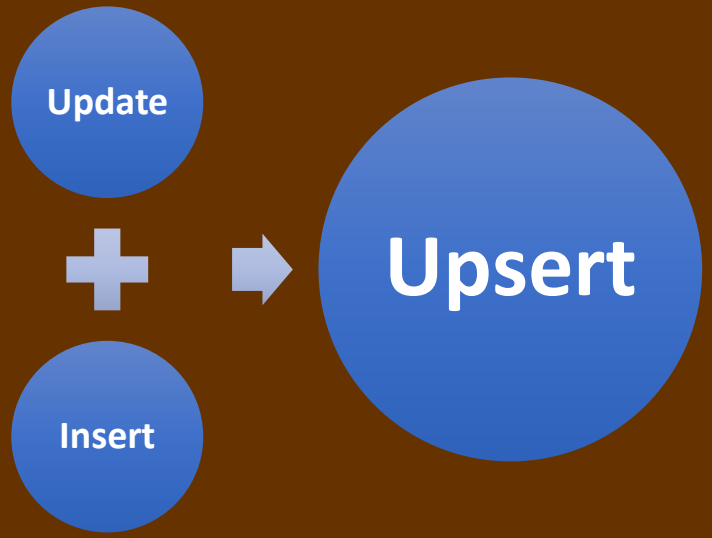


Cloud And Data Universe

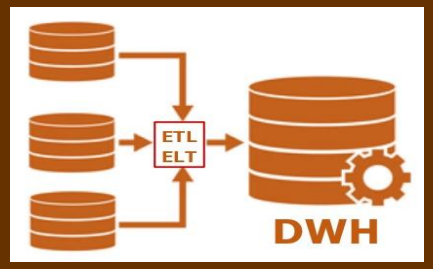


2. Upsert

- Combination of Update & Insert

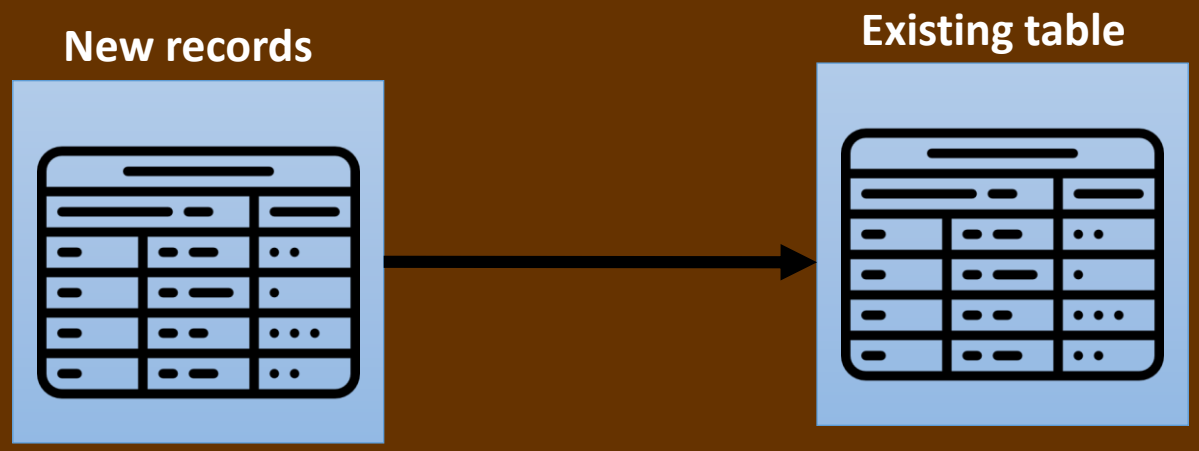


CDU
Cloud And Data Universe

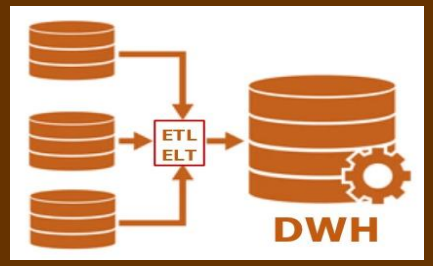


3. Incremental Load

- Identify & load new (delta) records mostly based on datetime

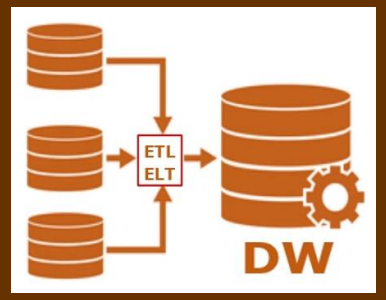


CDU
Cloud And Data Universe



4. Slowly changing dimensions (SCD)

- SCD1 - Update/overwrite changes
- SCD2 - Addition of new row
- SCD3 - Addition of new column



3. DATAWAREHOUSING

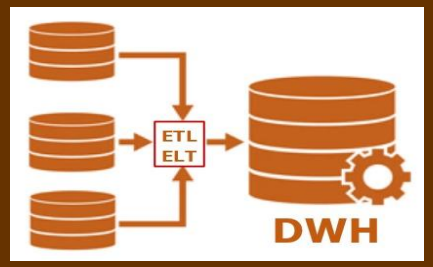
STORAGE LAYOUT MODELS

Follow us on :

YouTube

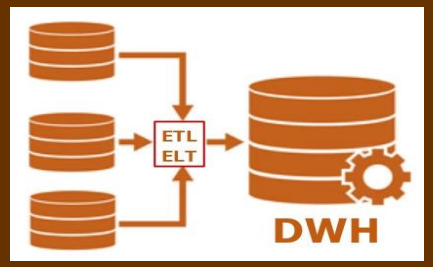
LinkedIn

CDU
Cloud And Data Universe



STORAGE LAYOUT MODELS

1. ROW-STORE
2. COLUMN-STORE
3. HYRBID-STORE



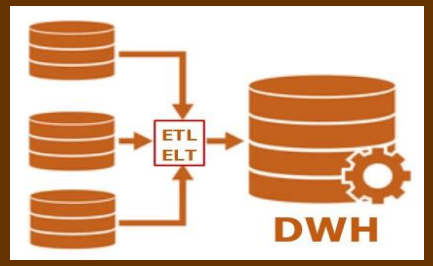
ROW-STORE

LOGICAL

EMPID	EMPNAME	SALARY
1	JOHN	4551
2	AMANDA	1105
3	SAM	4790
4	RAKESH	3720
5	AMAN	1223
6	RAHUL	3788

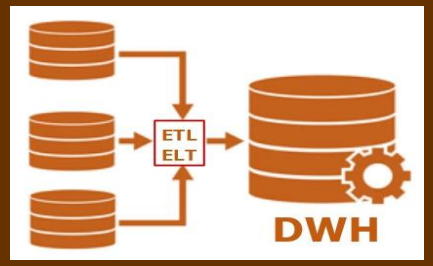
PHYSICAL

1	JOHN	4551	2	AMANDA	1105	3	SAM	4790
4	RAKESH	3720	5	AMAN	1223	6	RAHUL	3788



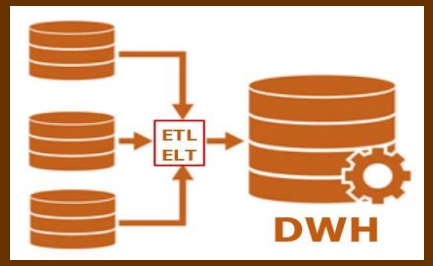
ROW-STORE

- Row-store / Row-Wise storage is horizontal partitioning.
- This is suitable when you need to insert or update a record.
- This model affects entire row where it scans through all the columns



ROW-STORE

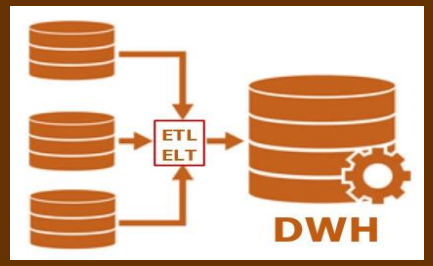
- Slower reads.
- Not optimized for querying.
- Used in case of frequent transactions.
- No efficient compression.



ROW-STORE

- Consider we need to sum up the salary column.
- In row store model it will scan through all the rows which is not performant.

EMPID	EMPNAME	SALARY
1	JOHN	4551
2	AMANDA	1105
3	SAM	4790
4	RAKESH	3720
5	AMAN	1223
6	RAHUL	3788



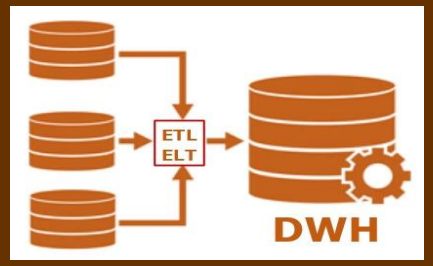
COLUMN-STORE

LOGICAL

EMPID	EMPNAME	SALARY
1	JOHN	4551
2	AMANDA	1105
3	SAM	4790
4	RAKESH	3720
5	AMAN	1223
6	RAHUL	3788

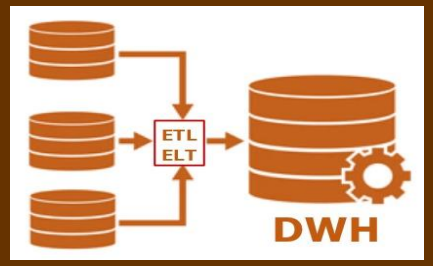
PHYSICAL

1	2	3	4	5	6	JOHN	AMANDA	SAM
RAKESH	AMAN	RAHUL	4551	1105	4790	3720	1223	3788



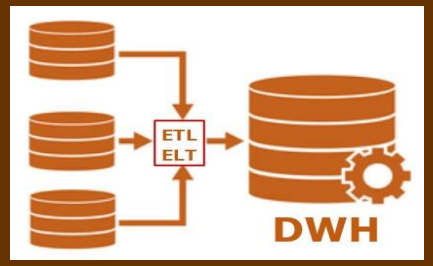
COLUMN-STORE

- Column-store model is vertical partitioning.
- In this model values of same column are stored continuously.
- In case where we need to extract specific column this model is preferred as it can easily extract those column/s without scanning entire data, which makes it extremely performant.



COLUMN-STORE

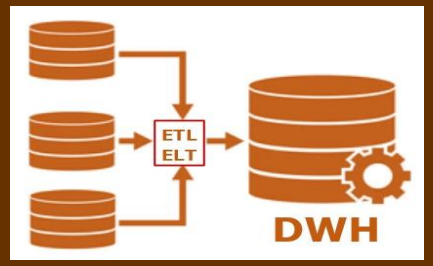
- Faster reads.
- Optimized for querying.
- Can’t be used in case of frequent transactions.
- Highly efficient compression.



COLUMN-STORE

- Consider we need to sum up up salary column.
- In row store model it will scan through all the rows which is not performant.

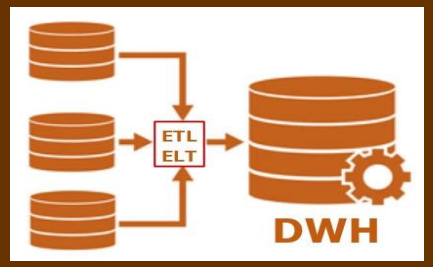
EMPID	EMPNAME	SALARY
1	JOHN	4551
2	AMANDA	1105
3	SAM	4790
4	RAKESH	3720
5	AMAN	1223
6	RAHUL	3788



COLUMN-STORE

1	2	3	4	5	6	JOHN	AMANDA	SAM
RAKESH	AMAN	RAHUL	4551	1105	4790	3720	1223	3788

- Consider we need to extract EMPNAME column.
- In this model it will easily extract the names as they are stored continuously, hence reducing the bottleneck vs row-store model.



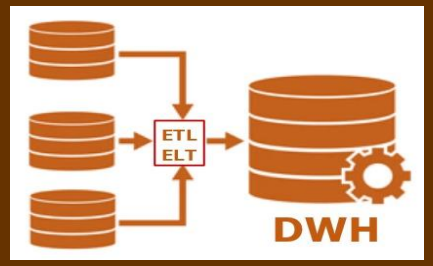
HYBRID-STORE

LOGICAL

EMPID	EMPNAME	SALARY
1	JOHN	4551
2	AMANDA	1105
3	SAM	4790
4	RAKESH	3720
5	AMAN	1223
6	RAHUL	3788

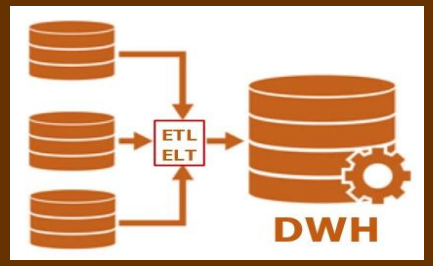
PHYSICAL

1	2	3	JOHN	AMANDA	SAM	4551	1105	4790
4	5	6	RAKESH	AMAN	RAHUL	3720	1223	3788



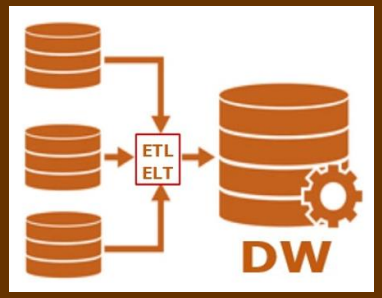
HYBRID-STORE

- Hybrid-store model combines both horizontal and vertical partitioning.



USE-CASES

- All these storage models are used in different SQL databases and file formats.



4.DATAWAREHOUSING

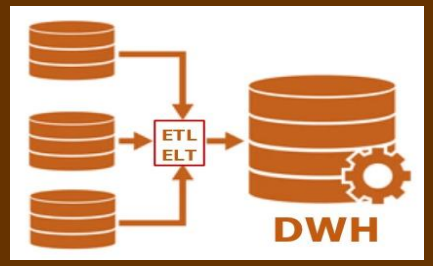
ROBIN ROUND DISTRIBUTION

Follow us on :

YouTube

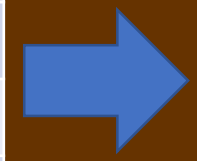
LinkedIn

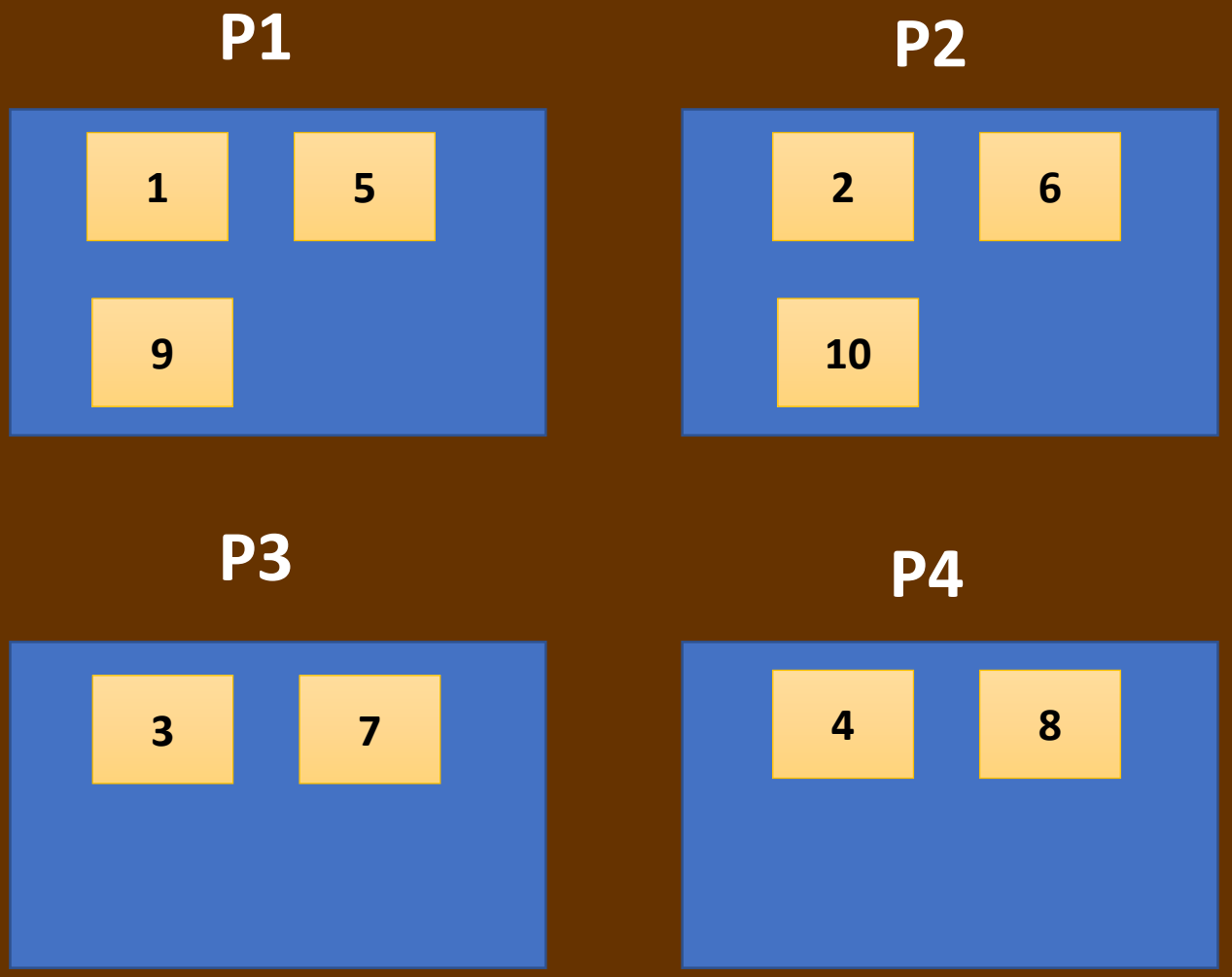
CDU
Cloud And Data Universe

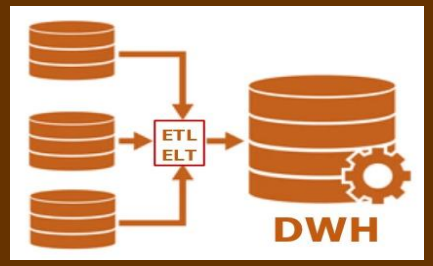


Robin round distribution

ID	QTY	PRICE	AMOUNT
1	10	10	100
2	5	3	15
3	1	2	2
4	20	10	200
..	..	..	..
..	..	..	..
10	100	3	300

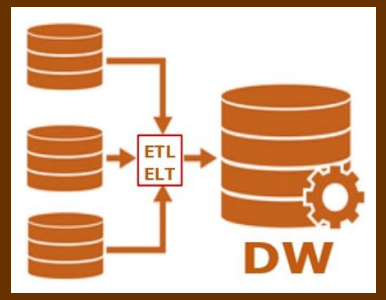






Important points

- Robin round distributes data evenly across partitions in a sequential way.
- It is fastest way to load a data in table/partition.
- Useful in case of staging.
- Not performant when using joins as data needs to be shuffled.



5. DATAWAREHOUSING

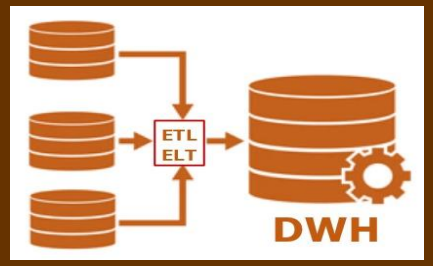
HASH DISTRIBUTION

Follow us on :

YouTube

LinkedIn

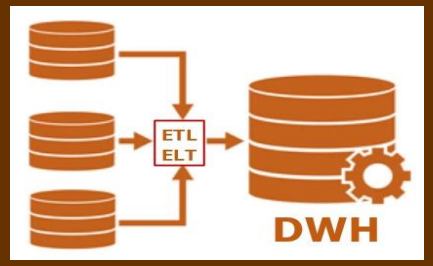
CDU
Cloud And Data Universe



HASH DISTRIBUTION

- Hash distribution uses a hash function to decide to pre determine which partition the row/value is to be assigned.

- Hash function returns a value called as hash value which is derived by using a calculation.



HASH DISTRIBUTION

ID	QTY	PRICE	AMOUNT
1	10	10	100
2	5	3	15
3	1	2	2
4	20	10	200
..	..	..	..
..	..	..	..
10	100	3	300