

HCIA Big Data Hive — Distributed Data Warehouse

Eng. Hagar Muhammed

Lecture 4



مجموعة جامعة
الجميع الذكية
للتعليم القابضة

Contents

- 1. Database Overview**
2. Components of DBMS
3. Use cases of DBMS
4. Data warehouse Overview
5. Data warehouse use cases
6. Hive

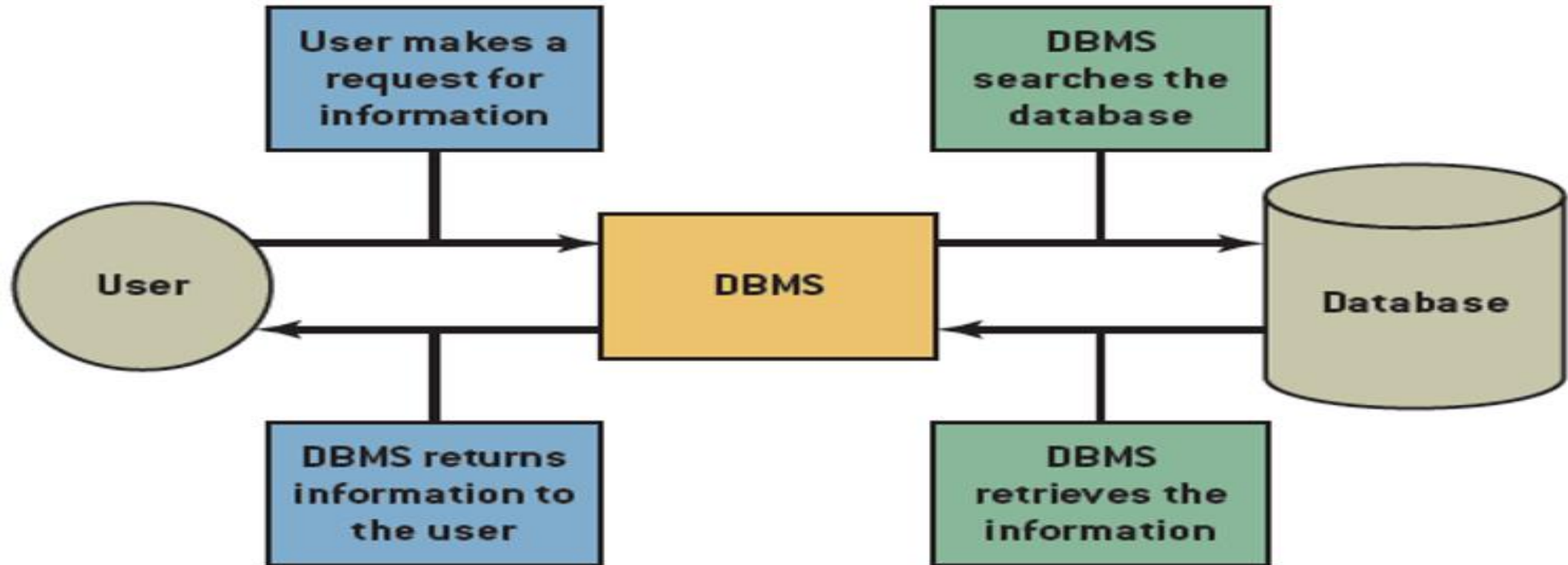
Database Overview

A **database** stores real-time information about one particular part of your business: its main job is to process the daily transactions that your company makes.

Database Overview

Databases use Online Transactional Processing (**OLTP**) to delete, insert, replace, and update large numbers of short online transactions quickly. This type of processing immediately responds to user requests, and so is used to process the day-to-day operations of a business in real-time. For example, if a user wants to reserve a hotel room using an online booking form, the process is executed with **OLTP**.

Database Management Systems



Contents

1. Database Overview
- 2. Components of DBMS**
3. Use cases of DBMS
4. Data warehouse Overview
5. Data warehouse use cases
6. Hive

Components of DBMS

1.Data Definition Language (DDL):

DDL is used to define the structure and schema of the database. It includes commands to create, modify, and delete database objects such as tables, indexes, and views.

2.Data Manipulation Language (DML):

DML allows users to interact with the data stored in the database. It includes commands like SELECT, INSERT, UPDATE, and DELETE for querying and modifying data.

Components of DBMS

3.Data Query Language (DQL):

DQL is a subset of DML and is specifically used for querying data. It primarily consists of the SELECT statement to retrieve data from the database.

4.Data Control Language (DCL):

DCL is used to control access to the database and its objects. It includes commands to grant and revoke permissions, ensuring data security and integrity.

Components of DBMS

5. Backup and Recovery Manager:

The backup and recovery manager is responsible for creating and managing database backups, allowing data restoration in case of failures or data loss.

6. Data Dictionary:

The data dictionary, also known as the metadata repository, stores information about the database's structure, data types, constraints, and relationships. It acts as a reference for the DBMS to understand the database's organization.

Contents

1. Database Overview
2. Components of DBMS
- 3. Use cases of DBMS**
4. Data warehouse Overview
5. Data warehouse use cases
6. Hive

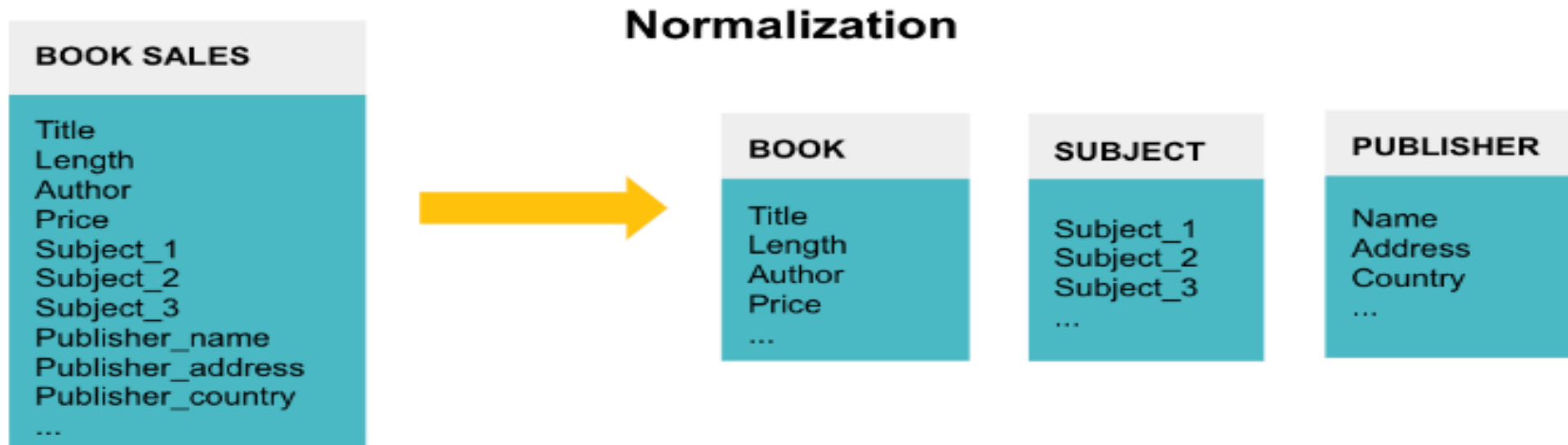
Use cases of DBMS

Databases process the day-to-day transactions in an organization. Some examples of database applications include:

- An ecommerce website creating an order for a product it has sold
- An airline using an online booking system
- A hospital registering a patient
- A bank adding an ATM withdrawal transaction to an account

Data Structure

The data in databases are normalized. The goal of normalization is to reduce and even eliminate data redundancy, i.e., storing the same piece of data more than once. This reduction of duplicate data leads to increased consistency and, thus, more accurate data as the database stores it in only one place



Contents

1. Database Overview
2. Components of DBMS
3. Use cases of DBMS
- 4. Data warehouse Overview**
5. Data warehouse use cases
6. Hive

Data Warehouse Overview

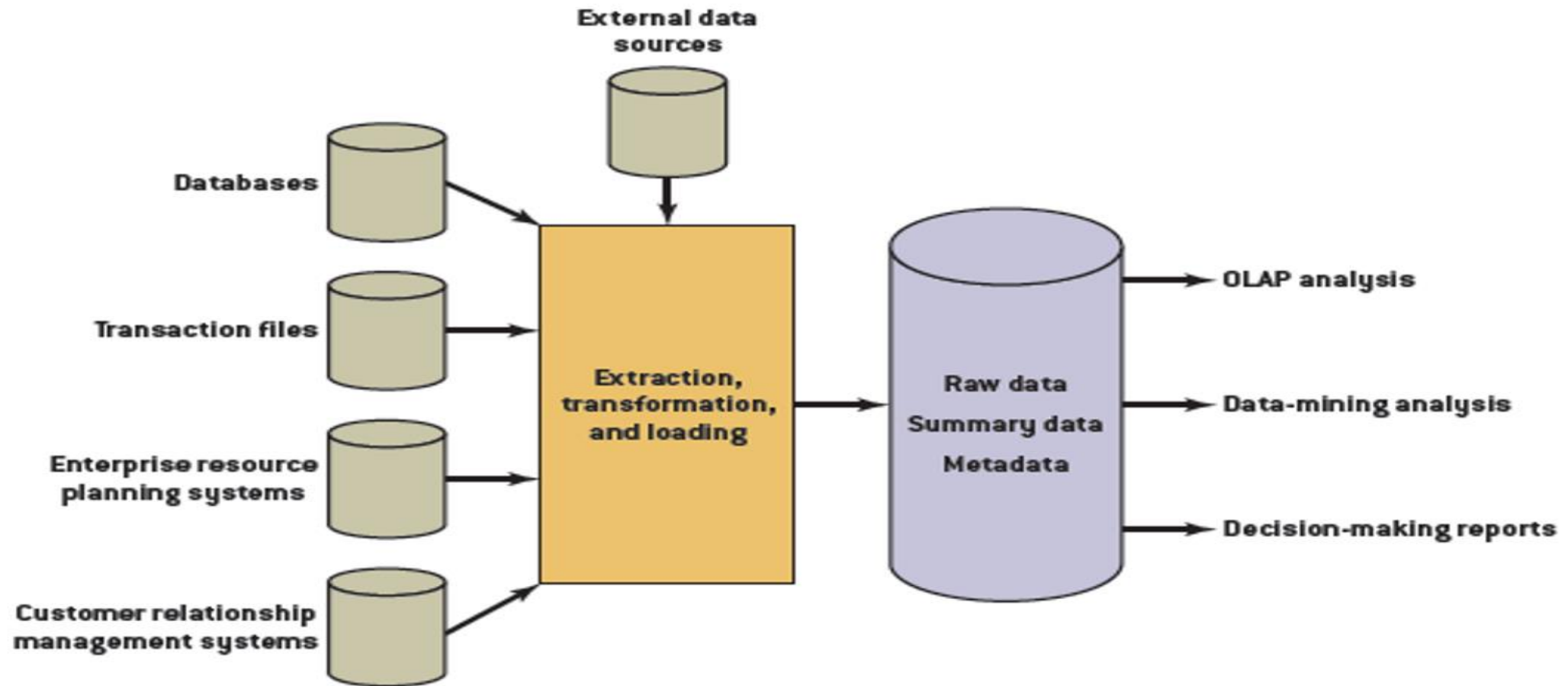
A data warehouse : is a system that pulls together data from many different sources within an organization for reporting and analysis. The reports created from complex queries within a data warehouse are used to make business decisions.

A data warehouse stores historical data about your business so that you can analyze and extract insights from it. It does not store current information, nor is it updated in real-time.

Data Warehouse Overview

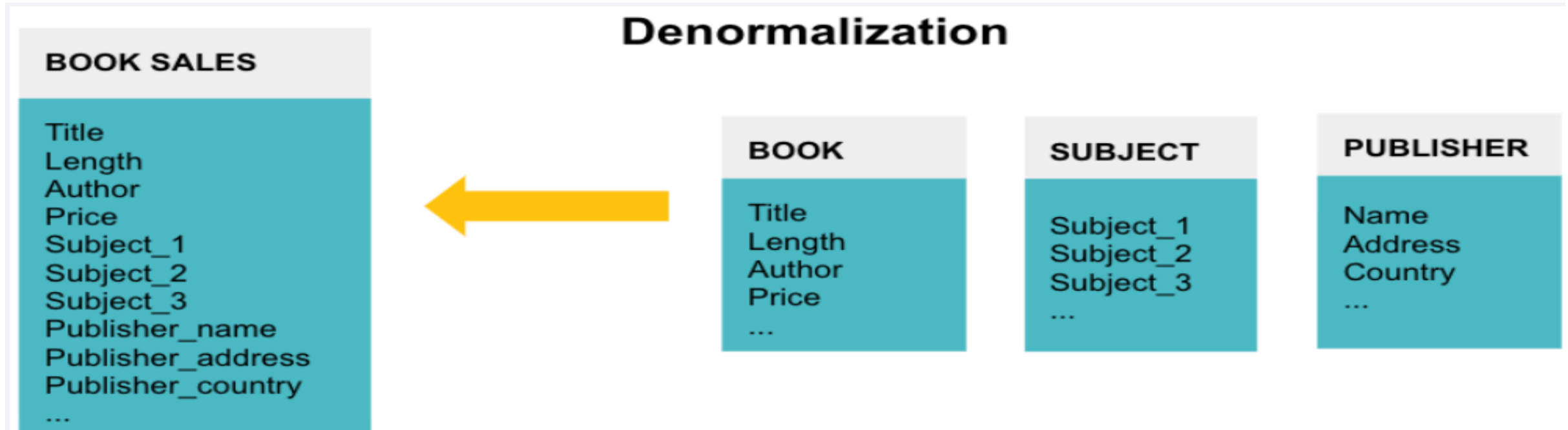
Data warehouses use Online Analytical Processing (**OLAP**) to analyze massive volumes of data rapidly. This process gives analysts the power to look at your data from different points of view. For example, even though your database records sales data for every minute of every day, you may just want to know the total amount sold each day. To do this, you need to collect and sum the sales data together for each day. **OLAP** is specifically designed to do this and using it for data warehousing 1000x faster than if you used OLTP to perform the same calculation.

Data Warehouse Overview



Data Structure

Normalizing data ensures the database takes up minimal disk space and so it is memory efficient. However, it is not query efficient. Querying a normalized database can be slow and cumbersome. Since businesses want to perform complex queries on the data in their data warehouse, that data is often denormalized and contains repeated data for easier access.



Contents

1. Database Overview
2. Components of DBMS
3. Use cases of DBMS
4. Data warehouse Overview
- 5. Data warehouse use cases**
6. Hive

Data warehouse use cases:

- 1. Business Intelligence and Reporting:** This enables organizations to generate comprehensive and accurate reports, dashboards, and visualizations for better insights into business performance.
- 2. Data Analysis and Analytics:** With historical and current data readily available in a data warehouse, data analysts and data scientists can perform in-depth analysis, identify trends, and make data-driven predictions to support strategic decision-making.
- 3. Customer Analysis:** Data warehouses can help businesses gain a deeper understanding of their customers by consolidating data from different touchpoints. This allows for segmentation, profiling, and personalized marketing campaigns.

Data Mart

- Data mart
 - Smaller version of data warehouse
 - Used by single department or function
- More limited scope than data warehouses

Database vs Data Warehouse

Aspect	Database	Data Warehouse
Purpose	Store and manage operational data	Store and manage analytical data
Data Type	Real-time transactional data	Historical and aggregated data
Usage	Day-to-day operations and transactions	Decision-making and analysis
Schema Design	Normalized for data integrity	Denormalized for query efficiency
Data Volume	Smaller data sets	Large volumes of data
Performance	Optimized for read/write operations	Optimized for complex queries
Transaction Types	Small, frequent transactions	Fewer, longer analytical queries
Time Span	Short-term data storage	Long-term historical storage
Example Use Cases	Online banking, inventory tracking	Business intelligence, trends analysis

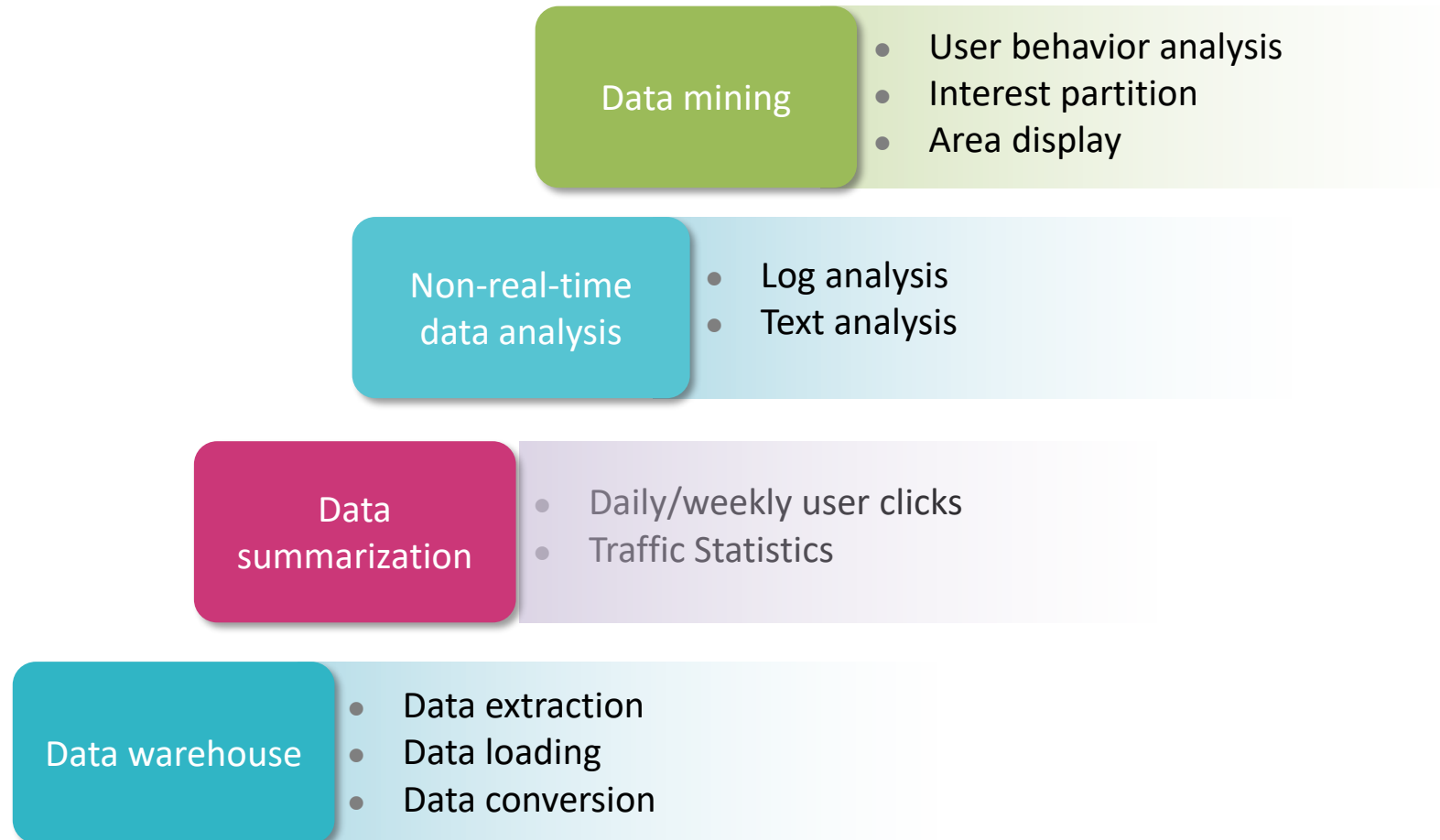
Contents

1. Database Overview
2. Components of DBMS
3. Use cases of DBMS
4. Data warehouse Overview
5. Data warehouse use cases
- 6. Hive — Distributed Data Warehouse**

Hive Overview

- Hive is a data warehouse tool that runs on Hadoop and supports PB-level distributed data query and management.
- **Hive features:**
 - Flexible extract, transform, load (ETL)
 - Support for multiple compute engines, such as Spark
 - Direct access to HDFS files and HBase
 - Easy to use and program

Use Cases of Hive



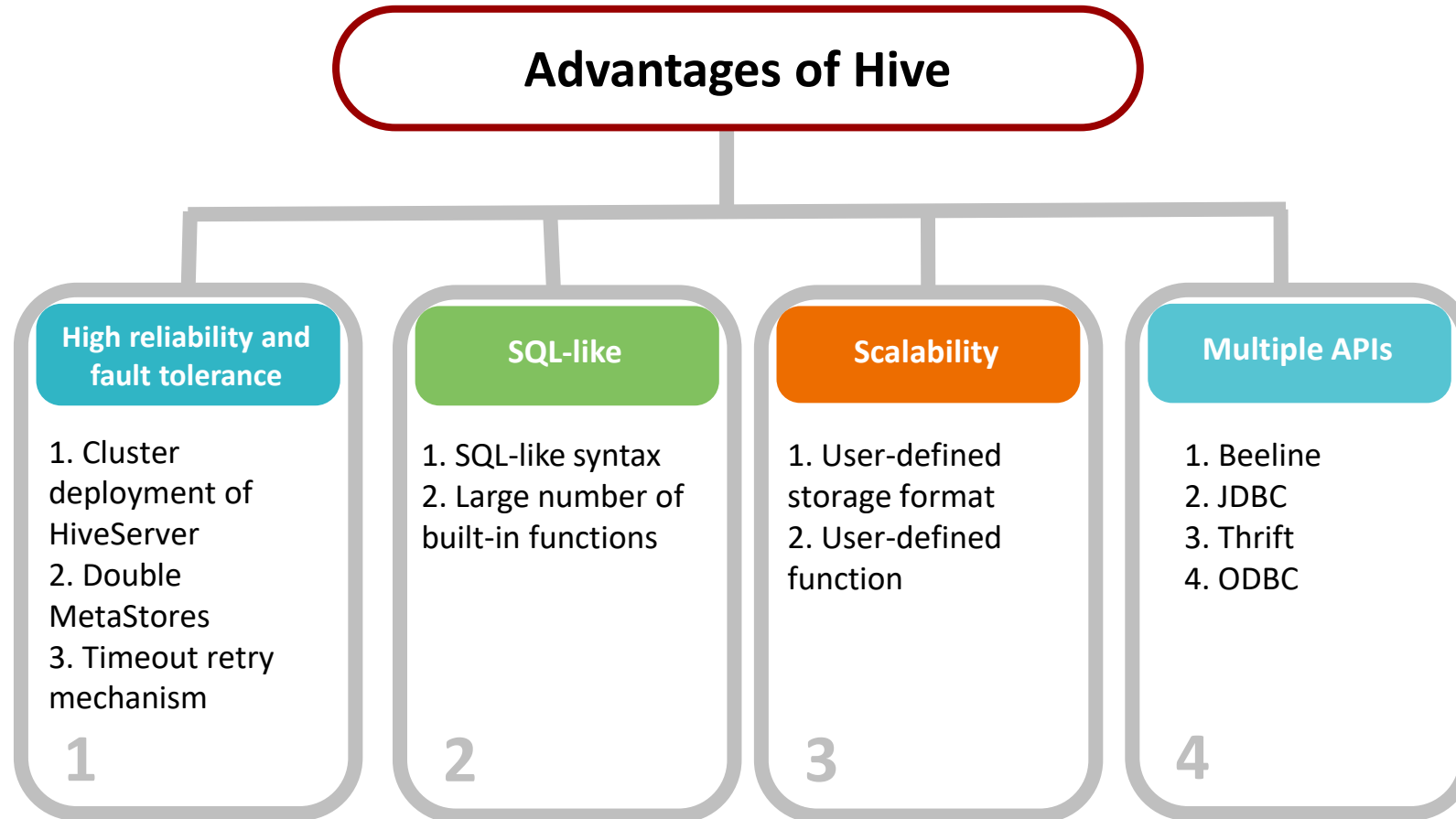
Hive vs. Traditional Data Warehouses

	Hive	Traditional Data Warehouse
Storage	HDFS is used to store data. Theoretically, infinite expansion is possible.	Clusters are used to store data, which has a capacity upper limit. With the increase of capacity, the computing speed decreases sharply. Therefore, data warehouses are applicable only to commercial applications with small data volumes.
Execution engine	The default execution engine is Tez.	More efficient algorithms can be selected to perform queries and more optimization measures can be taken to speed up the queries.
Usage	HQL (SQL-like)	SQL
Flexibility	Metadata storage is independent of data storage, decoupling metadata and data.	Low flexibility. Data can be used for limited purposes.
Analysis speed	Computing depends on the cluster scale and the cluster expands easily. In the case of a large amount of data, computing is much faster than that of a conventional data warehouse.	When the data volume is small, the data processing speed is high. When the data volume is large, the speed decreases sharply.

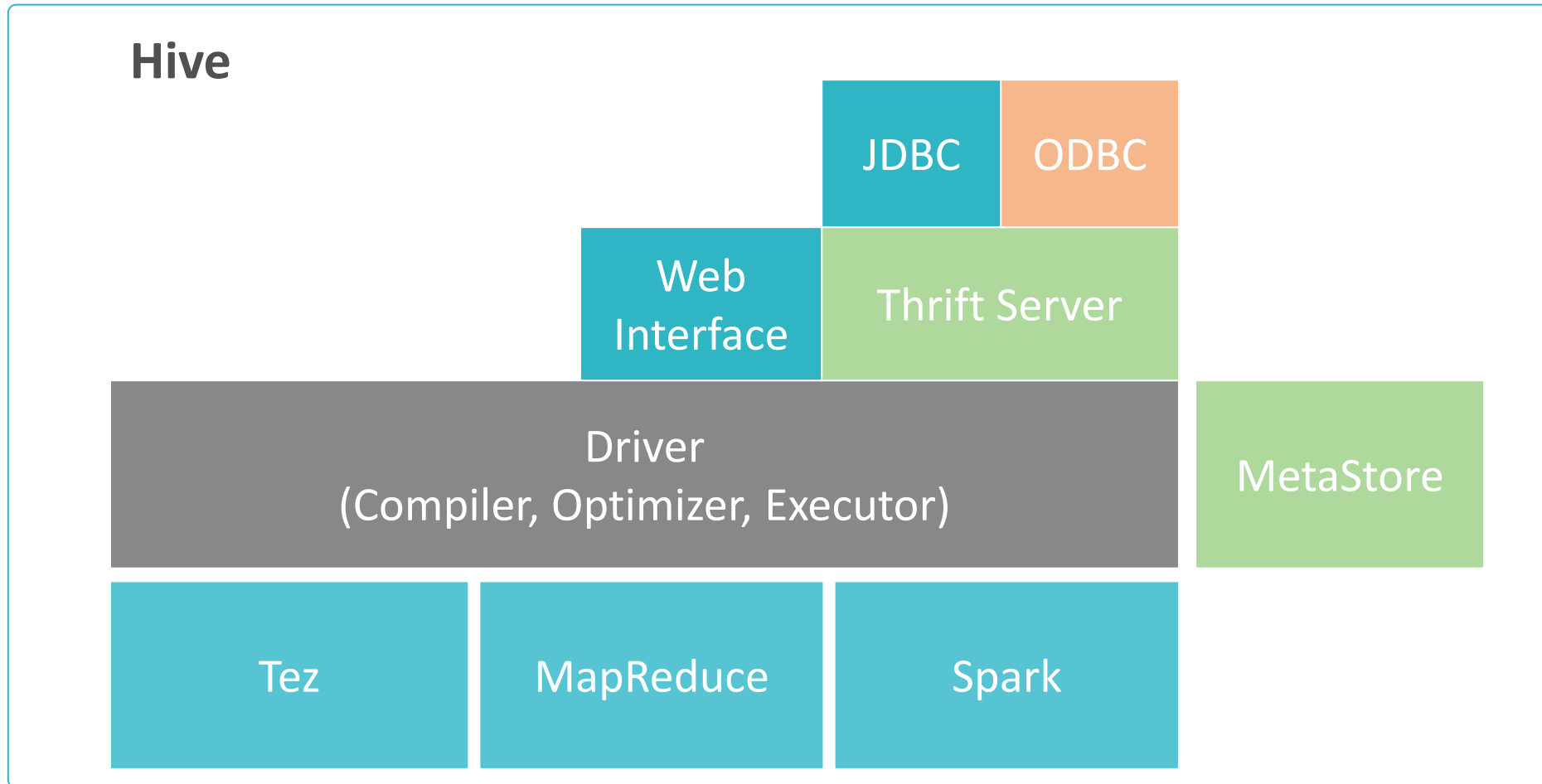
Hive vs. Traditional Data Warehouses

	Hive	Traditional Data Warehouse
Usability	Application models need to be self-developed. The flexibility is high, but the usability is low.	Integrates a set of mature report solutions to facilitate data analysis.
Reliability	Data is stored in HDFS, implementing high data reliability and fault tolerance.	Reliability is low. If a query fails, the task needs to be started again. The data fault tolerance depends on hardware.
Environment dependency	Low dependency on hardware, applicable to common machines	Highly dependent on high-performance business servers
Price	Open source	Expensive

Advantages of Hive

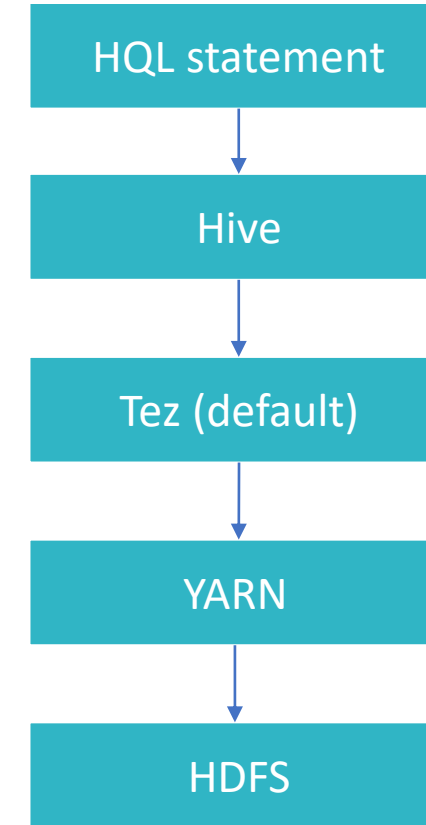


Hive Architecture

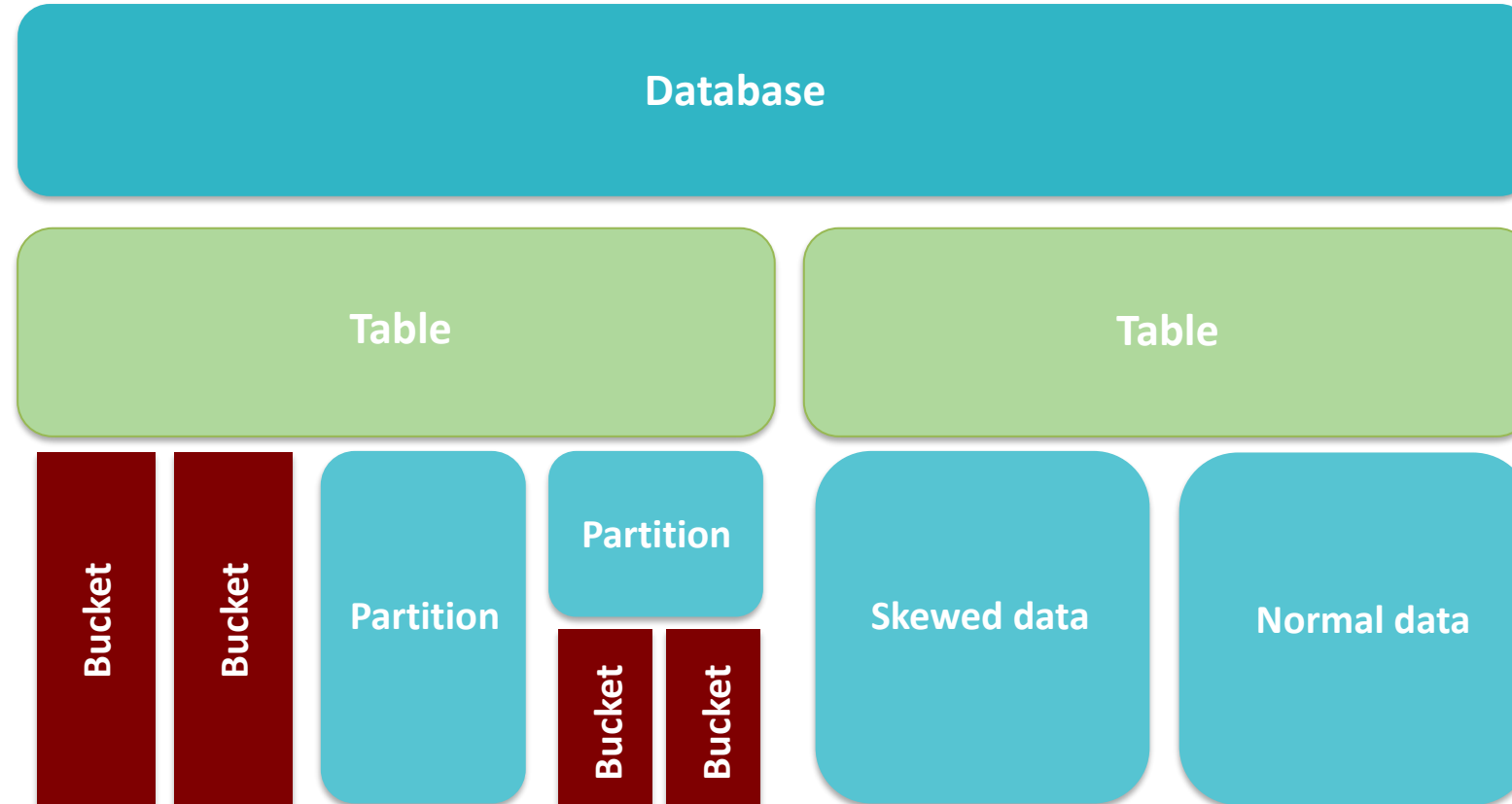


Hive Running Process

- A client submits HQL commands.
- Tez executes queries.
- YARN allocates resources to applications in the cluster and authorizes Hive jobs in the YARN queue.
- Hive updates data in HDFS or the Hive data warehouse based on table type.
- Hive returns the query result.



Data Storage Model of Hive



Hive Partition and Bucket

- Partition: Data tables can be partitioned based on the value of a certain field.
 - Each partition is a directory.
 - The number of partitions is not fixed.
 - Partitions or buckets can be created within partitions.

Hive Partition and Bucket

- Bucket: Data can be stored in different buckets.
 - Each bucket is a file.
 - The number of buckets is specified when creating a table. The buckets can be sorted.

Hive Data Storage Model: Managed and External Tables

- Hive can create managed and external tables.
 - After a Hive table is created, Hive moves data to the data warehouse directory.
 - If all processing is performed by Hive, you are advised to use managed tables.
 - If you want to use Hive and other tools to process the same data set, you are advised to use external tables.

	Managed Table	External Table
DROP	Metadata and data are deleted together.	Only metadata is deleted.

Functions Supported by Hive

- Built-in Hive functions
 - Mathematical functions, such as **round()**, **floor()**, **abs()**, and **rand()**.
 - Date functions, such as **to_date()**, **month()**, and **day()**.
 - String functions, such as **trim()**, **length()**, and **substr()**.
- User-defined functions (UDFs)

Hive SQL Overview

- DDL-Data Definition Language:
 - Creates tables, modifies tables, deletes tables, partitions, and data types.
- DML-Data Management Language:
 - Imports and exports data.
- DQL-Data Query Language:
 - Performs simple queries.
 - Performs complex queries such as **Group by**, **Order by**, and **Join**.

DDL Operations

-- Create a table:

```
hive> CREATE TABLE table_name (col1 data type, col2 data type);
```

-- Describe a table:

```
hive> DESCRIBE table_name;
```

-- Modify a table:

```
hive> ALTER TABLE table_name RENAME TO new_table_name;  
hive> ALTER TABLE table_name ADD COLUMNS (new_col data type);
```

DML Operations

-- update data to a table:

```
hive> update Table_name  
Set new_value  
Where condition;
```

-- delete data:

```
hive> Delete from Table_name  
Where condition;
```

DQL Operations

--SELECTS and FILTERS

```
hive> SELECT column_name FROM table_name WHERE condition;
```

--GROUP BY

```
hive> SELECT column_name, count(*) FROM table_name GROUP BY column_name;
```

Summary

Hive is a data warehousing and SQL-like query language system that facilitates the management and querying of large datasets in a distributed computing environment, commonly associated with the Apache Hadoop ecosystem. It was initially developed by Facebook and later open-sourced as part of the Apache Software Foundation. The primary goal of Hive is to provide a high-level, familiar interface for analysts and data scientists to interact with big data stored in distributed file systems, like Hadoop's HDFS

Thank You
For Your Attention