UNIT V – ADVANCED TOPICS

PART – A

1.State the need for distributed databases.

- **Scalability**
- **High Availability**
- **Fault Tolerance**
- **Performance**
- **Geographical Distribution**
- **Disaster Recovery**

2. Differentiate horizontal fragmentation from vertical fragmentation.

Aspect	Horizontal Fragmentation	Vertical Fragmentation
Definition	Data is divided based on rows or tuples of a relation.	Data is divided based on columns or attributes of a relation.
Example	Customer database table divided by customer locations.	Customer database table divided by type of customer information.
Scalability Impact	Improves scalability by distributing data across nodes, enabling parallel processing.	Improves scalability by reducing data size on each node, enhancing query performance.
Flexibility	Offers flexibility in managing data distribution based on criteria such as geographic location or customer segments.	Provides flexibility in optimizing data storage and access patterns based on application requirements.
Resource Utilization	Distributes data across nodes, enabling parallel processing of queries and transactions.	Reduces data size stored on each node, leading to more efficient resource utilization.

3.Define data replication.

Data replication is the process of creating and maintaining multiple copies of the same data in different locations as a way of ensuring data availability, reliability and resilience across an organization.

4.What is Object based database?

An object database stores complex data and relationships between data directly, without mapping to relational rows and columns, and this makes them suitable for applications dealing with very complex data.

Example: Address Book

5.Mention the advantages of XML databases.

- *Simplicity*
- *Openness*
- *Extensibility*
- *Self-description*
- *Contains machine-readable context information*
- *Separates content from presentation*
- *Supports multilingual documents and Unicode*.
- *Facilitates the comparison and aggregation of data.*

6. Define XML.

XML database is a data persistence software system that allows data to be stored in XML format. These data can then be queried, exported and serialized into the desired format. XML databases are usually associated with document-oriented databases.

7.List the differences between the SQL and NOSQL databases.

Aspect	SQL (Relational Databases)	NoSQL Databases
Data Model	Follows a structured, tabular format with predefined schemas.	Can follow various data models, including document, key-value, wide-column, or graph-based.

Aspect	SQL (Relational Databases)	NoSQL Databases
Schema	Requires a predefined schema with a fixed structure for tables.	Can have dynamic schemas, allowing for flexible data storage without predefined schema constraints.
Query Language	Primarily uses SQL (Structured Query Language) for querying data.	Uses a variety of query languages, often specific to the database type (e.g., MongoDB uses MongoDB Query Language).
Scalability	Vertical scalability (scaling up) by adding more powerful hardware.	Horizontal scalability (scaling out) by adding more servers or nodes to the database cluster.
ACID Transactions	Provides strong ACID (Atomicity, Consistency, Isolation, Durability) compliance for transactions.	May offer ACID properties but often prioritize high availability and partition tolerance (CAP theorem) over strict consistency.
Relationships	Emphasizes relationships between tables through foreign keys.	Relationships can be handled, but denormalization is often used to optimize performance.
Schema Changes	Schema changes require careful planning and can be complex, especially in production environments.	Schema changes can be more flexible and non-disruptive, allowing for easier adaptation to evolving data requirements.
Use Cases	Well-suited for applications with structured data and complex queries, such as banking systems or enterprise applications.	Ideal for applications with rapidly evolving data requirements, large-scale distributed systems, or unstructured/semi-structured data, such as social media platforms, IoT, or real-time analytics.
Examples	MySQL, PostgreSQL, Oracle Database	MongoDB, Cassandra, Redis, Amazon DynamoDB

8.List the categories of NOSQL database.

NoSQL databases are different from each other. There are four kinds of this database: document databases, key-value stores, column-oriented databases, and graph databases.

9. List the characteristics of NOSQL databases.

- Schema-free
- Simple API
- Distributed
- Complex-free working
- Better Scalability
- Flexible to accommodate
- Durable

10.Mention the applications of NOSQL database.

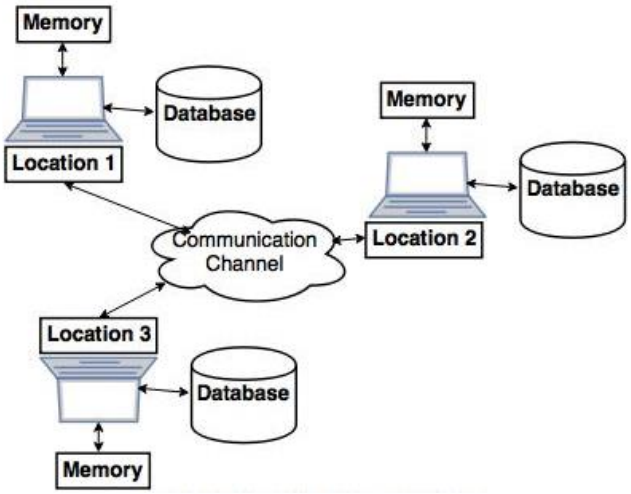
- Data Mining
- Social Media Networking Sites
- Software Development
- E-commerce
- Finance and Banking

PART-B

1.Detail the distributed database architecture, elaborating on its structure and functionality. also discuss how data replication and fragmentation are done in distributed environment with illustrative examples.

In a distributed database, there are a number of databases that may be geographically distributed all over the world. A distributed DBMS manages the distributed database in a manner so that it appears as one single database to users.

A **distributed database** is a collection of multiple interconnected databases, which are spread physically across various locations that communicate via a computer network. Distributed database is a system in which storage devices are not connected to a common processing unit. Database is controlled by Distributed Database Management System and data may be stored at the same location or spread over the interconnected network. It is a loosely coupled system. Shared nothing architecture is used in distributed databases.



Structure of Distributed Database Architecture:

1. **Data Distribution**: Data is distributed across multiple nodes or sites in the network. Each node may store a portion of the data, and data distribution strategies such as replication and fragmentation are used to ensure data availability and efficiency.
2. **Network Connectivity**: Nodes in the distributed database architecture are connected through a network infrastructure. This allows for communication and data exchange between nodes, facilitating distributed query processing, transaction management, and data synchronization.
3. **Database Management System (DBMS)**: Each node in the distributed database architecture runs a DBMS instance responsible for managing local data storage, query processing, and transaction management. These DBMS instances may be homogeneous or heterogeneous depending on the specific deployment.
4. **Distributed Query Processing**: Query processing involves distributing and executing queries across multiple nodes in parallel. Query optimization techniques are used to minimize data transfer and maximize parallelism, ensuring efficient query processing in distributed environments.
5. **Transaction Management**: Distributed transaction management ensures atomicity, consistency, isolation, and durability (ACID properties) across distributed data stores. Protocols such as the two-phase commit protocol are used to coordinate commit or rollback decisions across multiple nodes.
6. **Data Replication and Fragmentation**: Data replication involves maintaining multiple synchronized copies of the same data across different nodes to ensure redundancy and fault tolerance. Data fragmentation involves dividing data into smaller subsets or fragments and distributing them across multiple nodes to improve performance and scalability.

Features:

- Databases in the collection are logically interrelated with each other. Often they represent a single logical database.
- Data is physically stored across multiple sites. Data in each site can be managed by a DBMS independent of the other sites.
- The processors in the sites are connected via a network. They do not have any multiprocessor configuration.
- A distributed database is not a loosely connected file system.
- A distributed database incorporates transaction processing, but it is not synonymous with a transaction processing system.

FRAGMENTATION:

- Fragmentation is a process of dividing the whole or full database into various sub tables or sub relations so that data can be stored in different systems. The small pieces of sub relations or sub tables are called *fragments*. These fragments are called logical data units and are stored at various sites. It must be made sure that the fragments are such that they can be used to reconstruct the original relation (i.e, there isn't any loss of data).
-
- In the fragmentation process, let's say, If a table T is fragmented and is divided into a number of fragments say T1, T2, T3….TN. The fragments contain sufficient information to allow the restoration of the original table T. This restoration can be done by the use of UNION or JOIN operation on various fragments. This process is called ***data fragmentation***. All of these fragments are independent which means these fragments cannot be derived from others. The users needn't be logically concerned about fragmentation which means they should not concerned that the data is fragmented and this is called *fragmentation Independence* or we can say *fragmentation transparency*.

We have three methods for data fragmenting of a table:

- **Horizontal fragmentation**
- **Vertical fragmentation**
- **Mixed or Hybrid fragmentation**
-

DATA REPLICATION:

- **Data Replication** is the process of storing data in more than one site or node. It is useful in **improving the availability of data**. It is simply copying data from a database from one server to another server so that all the users can share the same data without any inconsistency. The result is a **distributed database** in which users can access data relevant to their tasks without interfering with the work of others.
-
- Data replication encompasses duplication of transactions on an ongoing basis, so that the **replicate is in a consistently updated state** and synchronized with the source. However in data replication data is available at different locations, but a particular relation has to reside at only one location.
-
- There can be full replication, in which the whole database is stored at every site. There can also be partial replication, in which some frequently used fragment of the database are replicated and others are not replicated.

Types of Data Replication –

1. **Transactional Replication**.
2. **Snapshot Replication**.
3. **Merge Replication**

Example: Consider a multinational corporation with regional offices located in different countries. The company maintains a centralized database for managing employee information, payroll data, and financial records. To ensure fault tolerance and high availability, the company implements data replication between the primary data center (located in the headquarters) and regional data centers.

2. (i) Compare and contrast between Object Oriented databases and XML databases

(ii) Generate an XML representation for a bank management system and develop an XML schema to define its structure and constraints, ensuring proper organization and validation of data

i)Ans:

Aspect	Object-Oriented Databases (OODB)	XML Databases
Data Model	Store data as objects with attributes and methods.	Organize data as hierarchical structures using XML markup.
Structure	Objects are structured with properties and behaviours.	Data is organized hierarchically using XML tags and attributes.
Query Language	Typically support proprietary query languages or extensions of SQL.	XQuery is commonly used for querying XML data.
Schema	Dynamic schema evolution is supported, allowing changes to the object structure.	Schema definition is typically rigid and defined using XML Schema (XSD).
Flexibility	Offers flexibility in representing complex data structures and relationships.	Provides flexibility in representing semi-structured and hierarchical data.
Performance	Efficient for complex data models and object-oriented applications.	Suitable for managing semi-structured data but may have performance overhead for complex queries.
Integration with Programming Languages	Integrates well with object-oriented programming languages like Java or C++.	Can be used with various programming languages through APIs and libraries.
Transaction Support	Supports transaction management with ACID properties.	Transaction support varies, with some XML databases providing ACID compliance.
Data Storage Format	Stores data natively in object format.	Stores data in text-based XML format.
Use Cases	Suited for applications with complex data structures and relationships, such as CAD systems or multimedia	Ideal for managing semi-structured data like documents, configuration files, or web data.

Aspect	Object-Oriented Databases (OODB)	XML Databases
	databases.	
Examples	db4o, ObjectDB, Versant Object Database.	eXist-db, BaseX, MarkLogic.

ii)ans: will be send

3.Provide an overview of temporal databases and spatial databases, detailing their respective concepts and applications in managing time-dependent and spatial data effectively.

Temporal Database

A Temporal Database is a database with **built-in support for handling time sensitive data**. Usually, databases store information only about current state, and not about past states. For example in a employee database if the address or salary of a particular person changes, the database gets updated, the old value is no longer there. However for many applications, it is important to maintain the past or historical values and the time at which the data was updated. That is, the knowledge of evolution is required. That is where temporal databases are useful. It stores information about the past, present and future. Any data that is time dependent is called the temporal data and these are stored in temporal databases.
Temporal Databases store information about states of the real world across time. Temporal Database is a database with built-in support for handling data involving time. It stores information relating to past, present and future time of all events.

Examples Of Temporal Databases

- **Healthcare Systems**: Doctors need the patients' health history for proper diagnosis. Information like the time a vaccination was given or the exact time when fever goes high etc.
- **Insurance Systems**: Information about claims, accident history, time when policies are in effect needs to be maintained.
- **Reservation Systems**: Date and time of all reservations is important.

Temporal Aspects

There are two different aspects of time in temporal databases.

- **Valid Time**: Time period during which a fact is true in real world, provided to the system.
- **Transaction Time**: Time period during which a fact is stored in the database, based on transaction serialization order and is the timestamp generated automatically by the system.

Temporal Relation

Temporal Relation is one where each tuple has associated time; either valid time or transaction time or both associated with it.

- **Uni-Temporal Relations:** Has one axis of time, either Valid Time or Transaction Time.
- **Bi-Temporal Relations:** Has both axis of time – Valid time and Transaction time. It includes Valid Start Time, Valid End Time, Transaction Start Time, Transaction End Time.

Valid Time Example

Now let's see an example of a person, John:

- John was born on April 3, 1992 in Chennai.
- His father registered his birth after three days on April 6, 1992.
- John did his entire schooling and college in Chennai.
- He got a job in Mumbai and shifted to Mumbai on June 21, 2015.
- He registered his change of address only on Jan 10, 2016.

Advantages

The main advantages of this bi-temporal relations is that it provides historical and roll back information. For example, you can get the result for a query on John's history, like: Where did John live in the year 2001?. The result for this query can be got with the valid time entry. The transaction time entry is important to get the rollback information.

- Historical Information – Valid Time.
- Rollback Information – Transaction Time.

Products Using Temporal Databases

The popular products that use temporal databases include:

- Oracle.
- Microsoft SQL Server.
- IBM DB2.

Spatial Databases

A spatial database is a database that is enhanced to store and access spatial data or data that defines a geometric space. These data are often associated with geographic locations and features or constructed features like cities. Data on spatial databases are stored as coordinates, points, lines, polygons, and topology. Some spatial databases handle more complex data like three-dimensional objects, topological coverage, and linear networks.

The **Open Geospatial Consortium (OGC)** developed the **Simple Features** specification (first released in 1997) and sets standards for adding spatial functionality to database systems. The SQL/MM Spatial ISO/IEC standard is a part of the SQL/MM multimedia standard and extends the Simple Features standard with data types that support circular interpolations.

Spatial Databases:

A database that needs to store and query spatial objects, e.g.

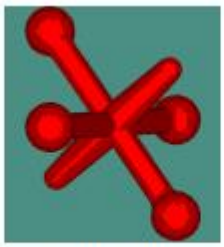
• Point: a house, a monument

• Line: a road segment, a road network

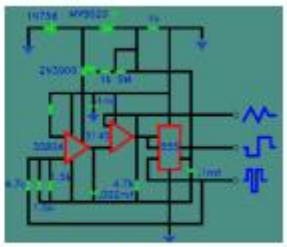
• Polygon: a county, a voting area

Types of spatial data

GIS Data



CAD Data



CAM Data

Spatial Database Management Systems:

- A Database Management System that manages data existing in some space
- 2D or 2.5D
    - Integrated circuits: VLSI design
    - Geographic space (surface of the Earth): GIS, urban planning _
- 2.5 D: Elevation
- 3D
    - Medicine: Brain models
    - Biological research: Molecule structures
    - Architecture: CAD
    - Ground models: Geology
- Supporting technology able to manage large collections of geometric objects
- Major commercial and open-source DBMSs provide spatial support

What makes it different from other database systems?

Common database systems use **indexes** for a faster and more efficient search and access of data. This index, however, is not fit for **spatial queries**. Instead, spatial databases use something like a unique index called a **spatial index** to speed up **database performance**. Spatial indexing is very much required because a system should be able to retrieve data from a large collection of objects without really searching the whole bunch. It should also **support relationships** between connecting objects from different classes in a better manner than just filtering.

Aside from the indexes, spatial databases also offer **spatial data types** in their **data model** and **query language**. These databases require special kinds of data types to provide a fundamental abstraction and model the structure of the geometric objects with their corresponding relationships and operations in the spatial environment. Without these kinds of data types, the system would not be able to support the kind of modeling a spatial database offers.

Why Spatial Databases?

- Queries to databases are posed in high level declarative manner (usually using SQL)
- SQL is popular in the commercial database world
- Standard SQL operates on relatively simple data types
- Additional spatial data types and operations can be defined in spatial database
- SQL was extended to support spatial data types and operations, e.g., OGC Simple Features for SQL
- A DBMS is a way of storing information in a manner that
    - Enforces consistency
    - Facilitates access
    - Allows users to relate data from multiple tables together.

Characteristics of Spatial Database

A spatial database system has the following characteristics

- It is a database system
- It offers spatial data types (SDTs) in its data model and query language.

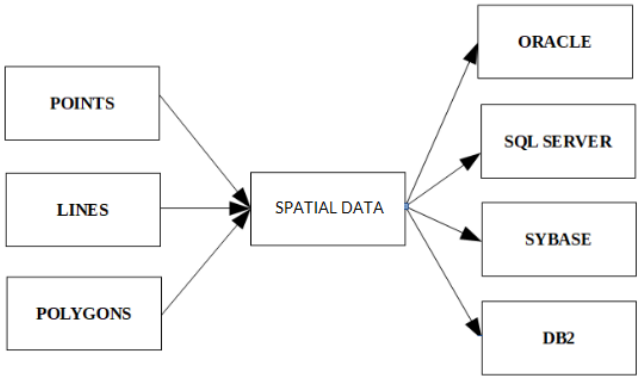
- It supports spatial data types in its implementation, providing at least spatial indexing and efficient algorithms for spatial join.

Example

A road map is a visualization of geographic information. A road map is a 2-dimensional object which contains points, lines, and polygons that can represent cities, roads, and political boundaries such as states or provinces.

In general, spatial data can be of two types −

- Vector data: This data is represented as discrete points, lines and polygons
- Rastor data: This data is represented as a matrix of square cells.



The spatial data in the form of points, lines, polygons etc. is used by many different databases as shown above.

Application Areas

- Street network-based
  - Vehicle routing and scheduling (cars, planes, trains)
  - Location analysis
- Natural resource-based
  - Management of areas: agricultural lands, forests, recreation resources, wildlife habitat analysis, migration routes planning...
  - Environmental impact analysis
  - Toxic facility siting
  - Groundwater modeling,
- Land parcel-based
  - Zoning, subdivision plan review

• Environmental impact statements
• Water quality management
• Facility management: electricity, gaz, clean water, used water, network

4. Compose notes on multimedia databases, covering their structure, management,and applications, while emphasizing their handling of various media types and retrieval techniques.

Multimedia database is the collection of interrelated multimedia data that includes text, graphics (sketches, drawings), images, animations, video, audio etc and have vast amounts of multisource multimedia data. The framework that manages different types of multimedia data which can be stored, delivered and utilized in different ways is known as multimedia database management system. There are three classes of the multimedia database which includes static media, dynamic media and dimensional media.

Content of Multimedia Database management system :

1. **Media data –** The actual data representing an object.
2. **Media format data –** Information such as sampling rate, resolution, encoding scheme etc. about the format of the media data after it goes through the acquisition, processing and encoding phase.
3. **Media keyword data –** Keywords description relating to the generation of data. It is also known as content descriptive data. Example: date, time and place of recording.
4. **Media feature data –** Content dependent data such as the distribution of colors, kinds of texture and different shapes present in data.

Types of multimedia applications based on data management characteristic are :

1. **Repository applications –** A Large amount of multimedia data as well as meta-data(Media format date, Media keyword data, Media feature data) that is stored for retrieval purpose, e.g., Repository of satellite images, engineering drawings, radiology scanned pictures.
2. **Presentation applications –** They involve delivery of multimedia data subject to temporal constraint. Optimal viewing or listening requires DBMS to deliver data at certain rate offering the quality of service above a certain threshold. Here data is processed as it is delivered. Example: Annotating of video and audio data, real-time editing analysis.
3. **Collaborative work using multimedia information –** It involves executing a complex task by merging drawings, changing notifications. Example: Intelligent healthcare network.

There are still many challenges to multimedia databases, some of which are :

1. **Modelling –** Working in this area can improve database versus information retrieval techniques thus, documents constitute a specialized area and deserve special consideration.
2. **Design –** The conceptual, logical and physical design of multimedia databases has not yet been addressed fully as performance and tuning issues at each level are far more complex as they consist of a variety of formats like JPEG, GIF, PNG, MPEG which is not easy to convert from one form to another.
3. **Storage –** Storage of multimedia database on any standard disk presents the problem of representation, compression, mapping to device hierarchies, archiving and buffering during input-output operation. In DBMS, a "BLOB"(Binary Large Object) facility allows un typed bitmaps to be stored and retrieved.
4. **Performance –** For an application involving video playback or audio-video synchronization, physical limitations dominate. The use of parallel processing may alleviate some problems but such techniques are not yet fully developed. Apart from this multimedia database consume a lot of processing time as well as bandwidth.
5. **Queries and retrieval –**For multimedia data like images, video, audio accessing data through query opens up many issues like efficient query formulation, query execution and optimization which need to be worked upon.

Areas where multimedia database is applied are :

- **Documents and record management :** Industries and businesses that keep detailed records and variety of documents. Example: Insurance claim record.
- **Knowledge dissemination :** Multimedia database is a very effective tool for knowledge dissemination in terms of providing several resources. Example: Electronic books.
- **Education and training :** Computer-aided learning materials can be designed using multimedia sources which are nowadays very popular sources of learning. Example: Digital libraries.
- Marketing, advertising, retailing, entertainment and travel. Example: a virtual tour of cities.
- **Real-time control and monitoring :** Coupled with active database technology, multimedia presentation of information can be very effective means for monitoring and controlling complex tasks Example: Manufacturing operation control.
- Marketing
- Advertisement
- Retailing
- Entertainment
- Travel

5. Summarize the following categories of NOSQL databases, providing an example for each to illustrate their respective structures and functionalities:

(i) Key Value Store

(ii) Document Store

(iii) Graph Database

What is NoSQL?

NoSQL Database is a non-relational Data Management System, that does not require a fixed schema. It avoids joins, and is easy to scale. The major purpose of using a NoSQL database is for distributed data stores with humongous data storage needs. NoSQL is used for Big data and real-time web apps. For example, companies like Twitter, Facebook and Google collect terabytes of user data every single day.

Types of NoSQL Databases

NoSQL Databases are mainly categorized into four types: Key-value pair, Column-oriented, Graph-based and Document-oriented. Every category has its unique attributes and limitations. None of the above-specified database is better to solve all the problems. Users should select the database based on their product needs.
Types of NoSQL Databases:

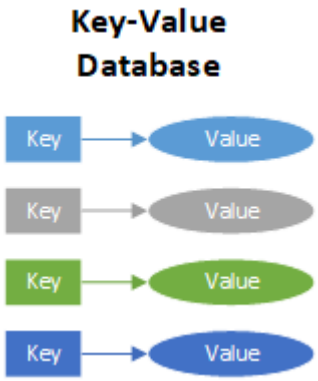
- Key-value Pair Based
- Graphs based
- Document-oriented

Key Value Pair Based

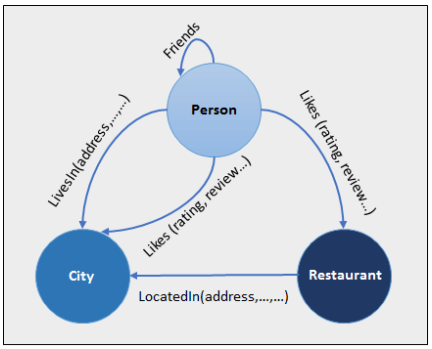
Data is stored in key/value pairs. It is designed in such a way to handle lots of data and heavy load.
Key-value pair storage databases store data as a hash table where each key is unique, and the value can be a JSON, BLOB(Binary Large Objects), string, etc.
For example, a key-value pair may contain a key like “Website” associated with a value like “Guru99”.

Key	Value
Name	Joe Bloggs
Age	42
Occupation	Stunt Double
Height	175cm
Weight	77kg

It is one of the most basic NoSQL database example. This kind of NoSQL database is used as a collection, dictionaries, associative arrays, etc. Key value stores help the developer to store schema-less data. They work best for shopping cart contents.
Redis, Dynamo, Riak are some NoSQL examples of key-value store DataBases. They are all based on Amazon's Dynamo paper.

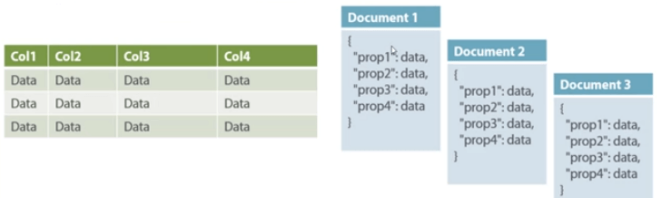


Graph-Based:
A graph type database stores entities as well the relations amongst those entities. The entity is stored as a node with the relationship as edges. An edge gives a relationship between nodes. Every node and edge has a unique identifier.



Compared to a relational database where tables are loosely connected, a Graph database is a multi-relational in nature. Traversing relationship is fast as they are already captured into the DB, and there is no need to calculate them.
Graph base database mostly used for social networks, logistics, spatial data.
Neo4J, Infinite Graph, OrientDB, FlockDB are some popular graph-based databases.

Document-Oriented:
Document-Oriented NoSQL DB stores and retrieves data as a key value pair but the value part is stored as a document. The document is stored in JSON or XML formats. The value is understood by the DB and can be queried.



Relational Vs. Document

In this diagram on your left you can see we have rows and columns, and in the right, we have a document database which has a similar structure to JSON. Now for the relational database, you have to know what columns you have and so on. However, for a document database, you have data store like JSON object. You do not require to define which make it flexible. The document type is mostly used for CMS systems, blogging platforms, real-time analytics & e-commerce applications. It should not use for complex transactions which

require multiple operations or queries against varying aggregate structures. Amazon SimpleDB, CouchDB, MongoDB, Riak, Lotus Notes, MongoDB, are popular Document originated DBMS systems.