No SQL Database

Dept of CSE

Module - 1

Dept of CSE

Why NoSQL? The Value of Relational Databases, Getting at Persistent Data, Concurrency, Integration, A (Mostly) Standard Model, Impedance Mismatch, Application and Integration Databases, Attack of the Clusters, The Emergence of NoSQL, Aggregate Data Models; Aggregates, Example of Relations and Aggregates, Consequences of Aggregate Orientation, Key-Value and Document Data Models, Column-Family Stores, Summarizing Aggregate Oriented Databases. More Details on Data Models; Relationships, Graph Databases, Schemaless Databases, Materialized Views, Modeling for Data Access

ACHARYA

Dept of CSE

Course Objectives

CLO 1. Recognize and Describe the four types of NoSQL Databases, the Document-oriented, Key-Value pairs
CLO 2. Column-oriented and Graph databases useful for diverse applications.
CLO 3. Apply performance tuning on Column-oriented NoSQL databases and Document-oriented NoSQL Databases.
CLO 4. Differentiate the detailed architecture of column oriented NoSQL database, Document database and Graph Database and relate usage of processor, memory, storage and file system commands.
CLO 5. Evaluate several applications for location based service and recommendation services. Devise an application using the components of NoSQL

Course Outcomes

CO1. Demonstrate an understanding of the detailed architecture of Column Oriented NoSQL databases, Document databases, Graph databases.
CO2. Use the concepts pertaining to all the types of databases.
CO3. Analyze the structural Models of NoSQL.
CO4. Develop various applications using NoSQL databases.

ACHARYA

Dept of CSE

Assessment Details (both CIE and SEE)

Continuous Internal Evaluation:

Three Unit Tests each of 20 Marks (duration 01 hour)	60M
Two assignments each of 10 Marks	20M
Group discussion/Seminar/quiz any one of three suitably planned	20M

CIE 100 M --> 50 M

SEE 100M ---> 50M

Dept of CSE

Textbooks

1. Sadalage, P. & Fowler, NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence, Pearson Addision Wesley, 2012

Reference Books

1. Dan Sullivan, "NoSQL For Mere Mortals", 1st Edition, Pearson Education India, 2015. (ISBN- 13: 978-9332557338)

2. Dan McCreary and Ann Kelly, "Making Sense of NoSQL: A guide for Managers and the Rest of us", 1st Edition, Manning Publication/Dreamtech Press, 2013. (ISBN-13: 978-9351192022)

3. Kristina Chodorow, "Mongodb: The Definitive Guide- Powerful and Scalable Data Storage", 2nd Edition, O'Reilly Publications, 2013. (ISBN-13: 978-9351102694)

Dept of CSE

Weblinks and Video Lectures (e-Resources):

1. https://www.geeksforgeeks.org/introduction-to-nosql/ (and related links in the page)

2. https://www.youtube.com/watch?v=0buKQHokLK8 (How do NoSQL databases work? Simply
explained)

3. https://www.techtarget.com/searchdatamanagement/definition/NoSQL-Not-Only-SQL (What is
NoSQL and How do NoSQL databases work)

4. https://www.mongodb.com/nosql-explained (What is NoSQL)

5. https://onlinecourses.nptel.ac.in/noc20-cs92/preview (preview of Bigdata course contains
NoSQL)

ACHARYA

Dept of CSE

Module - 1

Why NoSQL? The Value of Relational Databases, Getting at Persistent Data, Concurrency, Integration, A (Mostly) Standard Model, Impedance Mismatch, Application and Integration Databases, Attack of the Clusters, The Emergence of NoSQL.

Aggregate Data Models; Aggregates, Example of Relations and Aggregates, Consequences of Aggregate Orientation, Key-Value and Document Data Models, Column-Family Stores, Summarizing Aggregate Oriented Databases.

More Details on Data Models; Relationships, Graph Databases, Schemaless Databases, Materialized Views, Modeling for Data Access

ACHARYA

Dept of CSE

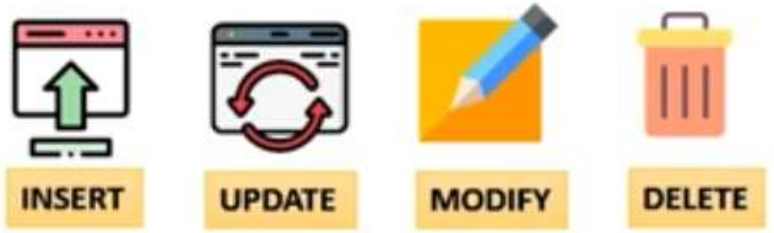
- WHAT IS SQL?
- WHY WE USE SQL?
- WHAT IS NOSQL?
- WHY WE USE NOSQL?
- HOW NOSQL WORKS?
- TYPES OF NOSQL DATABASES
- COMPARISON BETWEEN SQL AND NOSQL
- ADVANTAGES & DISADVANTAGES OF NOSQL

ACHARYA

Dept of CSE

1 WHAT IS SQL?

STRUCTURED QUERY LANGUAGE IS USED TO COMMUNICATE WITH DATABASE. ALLOWS THE USER TO ACCESS AND MANIPULATE DATA STORED IN THE DATABASE



ACHARYA

Dept of CSE

2 RELATIONAL DATABASE SYSTEMS

RELATIONAL DATABASE IS THE USED TO ORGANIZE STRUCTURED DATA INTO TABLES IN THE FORM OF ROWS AND COLUMNS

ACHARYA

Dept of CSE

3 POPULAR SQL DATABASES

Dept of CSE

Dept of CSE

Aggregate Data Model in NoSQL

We know, **NoSQL** are databases that store data in another format other than relational databases. Now a days, NoSQL deals in nearly every industry. For the people who interact with data in databases, the Aggregate Data model will help in that interaction.

Features of NoSQL Databases:

- **Schema Agnostic:** NoSQL Databases do not require any specific schema or storage structure than traditional RDBMS.
- **Scalability:** NoSQL databases scale horizontally as data grows rapidly certain commodity hardware could be added and scalability features could be preserved for NoSQL.
- **Performance:** To increase the performance of the NoSQL system *one can add a different commodity server than reliable and fast access of database transfer with minimum overhead*.
- **High Availability:** In traditional RDBMS it relies on primary and secondary nodes for fetching the data, Some NoSQL databases **use master place architecture**.
- **Global Availability:** As data is replicated among multiple servers and clouds **the data is accessible to anyone, this minimizes the latency period.**

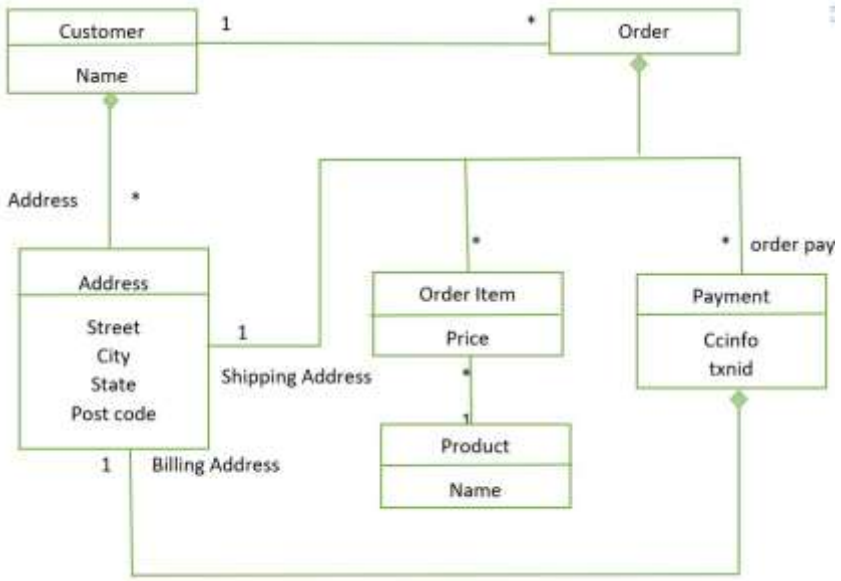
ACHARYA

Dept of CSE

Aggregate Data Models:

The term aggregate means a collection of objects that we use to treat as a unit. An aggregate is a collection of data that we interact with as a unit. These units of data or aggregates form the boundaries for ACID operation.

Example of Aggregate Data Model:



Here in the diagram have two Aggregate:

- **Customer** and **Orders link** between them represent an aggregate.
- The diamond shows how data fit into the aggregate structure.

- Customer contains a list of billing address
- Payment also contains the billing address
- The address appears three times and it is copied each time
- The domain is fit where we don't want to change shipping and billing address.

Dept of CSE

Consequences of Aggregate Orientation:

• Aggregation is not a logical data property, It is all about how the data is being used by applications.

• An aggregate structure may be an obstacle for others but help with some data interactions.

• It has an important consequence for transactions.

• NoSQL databases don’t support ACID transactions thus sacrificing consistency.

• aggregate-oriented databases support the atomic manipulation of a single aggregate at a time.

ACHARYA

Dept of CSE

Advantage:

- It can be used as a primary data source for online applications.
- Easy Replication.
- No single point Failure.
- It provides fast performance and horizontal Scalability.
- It can handle Structured semi-structured and unstructured data with equal effort.

Disadvantage:

- No standard rules.
- Limited query capabilities.
- Doesn't work well with relational data.
- Not so popular in the enterprise.
- When the value of data increases it is difficult to maintain unique values.

ACHARYA

Dept of CSE

Key-Value Data Model in NoSQL

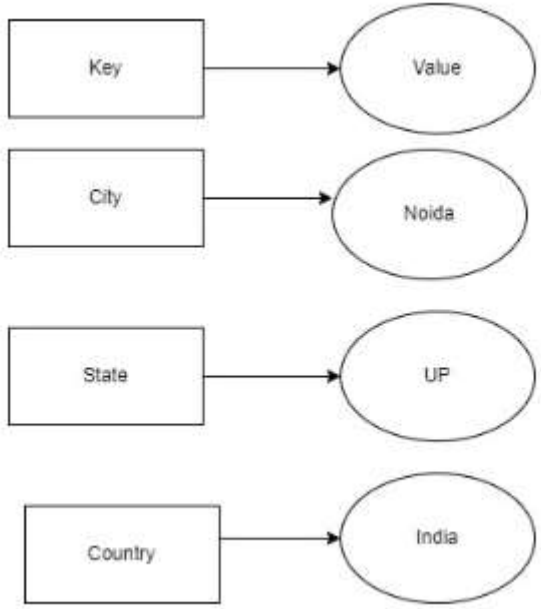
A key-value data model or database is also referred to as a ***key-value store***.

It is a ***non-relational type of database***.

In this, an ***associative array*** is used as a basic database in which *an individual key is linked with just one value in a collection*.

For the values, keys are special identifiers. Any kind of entity can be valued.

The collection of key-value pairs stored on separate records is called ***key-value databases*** and they do not have an already defined structure.



ACHARYA

Dept of CSE

When to use a key-value database:

Here are a few situations in which you can use a key-value database:-

- User session attributes in an online app like finance or gaming, which is referred to as real-time random data access.
- Caching mechanism for repeatedly accessing data or key-based design.
- The application is developed on queries that are based on keys.

Features:

- One of the most un-complex kinds of NoSQL data models.
- For storing, getting, and removing data, key-value databases utilize simple functions.
- Querying language is not present in key-value databases.
- Built-in redundancy makes this database more reliable.

Advantages:

- It is very easy to use. Due to the simplicity of the database, data can accept any kind, or even different kinds when required.
- Its response time is fast due to its simplicity, given that the remaining environment near it is very much constructed and improved.
- Key-value store databases are scalable vertically as well as horizontally.
- Built-in redundancy makes this database more reliable.

Dept of CSE

Disadvantages:

- As querying language is not present in key-value databases, transportation of queries from one database to a different database cannot be done.
- The key-value store database is not refined. You cannot query the database without a key.

Here are some popular key-value databases which are widely used:

- **Couchbase:** It permits SQL-style querying and searching for text.
- **Amazon DynamoDB:** The key-value database which is mostly used is Amazon DynamoDB as it is a trusted database used by a large number of users. It can easily handle a large number of requests every day and it also provides various security options.
- **Riak:** It is the database used to develop applications.
- **Aerospike:** It is an open-source and real-time database working with billions of exchanges.
- **Berkeley DB:** It is a high-performance and open-source database providing scalability.

Document Data Model:

A Document Data Model is a lot different than other data models because it stores data in ***JSON, BSON, or XML documents***.

In this data model, ***we can move documents under one document and apart from this, any particular elements can be indexed to run queries faster***.

Often documents ***are stored and retrieved in such a way that it becomes close to the data objects which are used in many applications*** which means very less translations are required to use data in applications.

JSON is a native language that is often used to store and query data too.
So in the document data model, each document has a key-value pair below

```
{
"Name" : "Yashodhra",
"Address" : "Near Patel Nagar",
"Email" : "yahoo123@yahoo.com",
"Contact" : "12345"
}
```

Dept of CSE

Working of Document Data Model:

This is a data model which ***works as a semi-structured data model*** in which the records and data associated with them are stored in a single document which means this data model is not completely unstructured. The main thing is that *data here is stored in a document.*

Features:

- **Document Type Model:** As we all know data is stored in documents rather than tables or graphs, so it becomes easy to map things in many programming languages.
- **Flexible Schema:** Overall schema is very much flexible to support this statement one must know that not all documents in a collection need to have the same fields.
- **Distributed and Resilient:** Document data models are very much dispersed which is the reason behind horizontal scaling and distribution of data.
- **Manageable Query Language:** These data models are the ones in which query language allows the developers to perform **CRUD** (Create Read Update Destroy) operations on the data model.

ACHARYA

Dept of CSE

Examples of Document Data Models :

•Amazon DocumentDB
•MongoDB
•Cosmos DB
•ArangoDB
•Couchbase Server
•CouchDB

Advantages:

•**Schema-less:** There are absolutely no restrictions in the format and the structure of data storage.
•**Faster creation of document and maintenance:** It is very simple to create a document and apart from this maintenance requires is almost nothing.
•**Open formats:** It has a very simple build process that uses XML, JSON, and its other forms.
•**Built-in versioning:** It has built-in versioning which means *as the documents grow in size there might be a chance they can grow in complexity*. Versioning decreases conflicts.

Disadvantages:

•**Weak Atomicity:** It lacks in supporting multi-document ACID transactions.
•**Consistency Check Limitations:** One can search the collections and documents that are not connected to an author collection but doing this might create a problem in the performance of database performance.
•**Security:** Nowadays many web applications lack security which in turn results in the leakage of sensitive data. So it becomes a point of concern, one must pay attention to web app vulnerabilities.

ACHARYA

Dept of CSE

Column Family Stores Data Model of NoSQL :

Basically, the relational database stores data in rows and also reads the data row by row, column store is organized as a set of columns.

Columnar databases are those where the data is stored in **columns** instead of rows as is done in the traditional **row-based databases** as they offer impressive benefits in certain types of queries and **data manipulation** operations.

Therefore, databases that are organized in columns provide better performances for operations that require applications to read data such as **data analytics and data warehousing.**

This approach **enhances the rate of query processing and reduces disk I/O making it suitable for cases** that involve a large number of queries that are processed on large amounts of data.

Examples of Columnar Data Model: **Cassandra and Apache Hadoop Hbase**

When to use the Columnar Database

1.Queries that involve only a few columns.
2.Compression but column-wise only.
3.Queries against a huge amount of data.

ACHARYA

Dept of CSE

Working of Columnar Data Model:

In Columnar Data Model instead of organizing information into rows, it does in columns. This makes them function the same way that tables work in relational databases. This type of data model is much more flexible obviously because it is a type of NoSQL database. The below example will help in understanding the Columnar data model:

Row-Oriented Table:

S.No.	Name	Course	Branch	ID
01.	Tanmay	B-Tech	Computer	2
02.	Abhishek	B-Tech	Electronics	5
03.	Samriddha	B-Tech	IT	7
04.	Aditi	B-Tech	E & TC	8

Column – Oriented Table:

S.No.	Name	ID
01.	Tanmay	2
02.	Abhishek	5
03.	Samriddha	7
04.	Aditi	8

S.No.	Course	ID
01.	B-Tech	2
02.	B-Tech	5
03.	B-Tech	7
04.	B-Tech	8

S.No.	Branch	ID
01.	Computer	2
02.	Electronics	5
03.	IT	7
04.	E & TC	8

Columnar Data Model uses the concept of keyspace, which is like a schema in relational models.

ACHARYA

Dept of CSE

Advantages of Columnar Database

1.Columnar databases can be used for different tasks such as when the **applications that are related to big data comes into play** then the column-oriented databases have greater attention in such case.

2.The data in the columnar database has a highly compressible nature and has different operations like (**AVG**), (**MIN**, **MAX**), which are permitted by the compression.

3.Efficiency and Speed: The speed of Analytical queries that are performed is faster in columnar databases.

4.Self-indexing: Another benefit of a column-based **DBMS** is self-indexing, which uses less disk space than a relational database management system containing the same data.

Dept of CSE

Applications of Columnar Data Model:

- Columnar Data Model is very much used in various **Blogging Platforms**.
- It is used in Content management systems like **WordPress, Joomla**, etc.
- It is used in Systems that maintain **counters**.
- It is used in Systems that require **heavy write requests**.
- It is used in Services that have **expiring usage**.

Limitation of Columnar Database

1.For loading incremental data, traditional databases are more relevant as compared to column-oriented databases.
2.For **Online transaction processing** (**OLTP**) applications, Row oriented databases are more appropriate than columnar databases.

ACHARYA

Dept of CSE

Graph Database on NoSQL

A graph database is a type of NoSQL database that **is designed to handle data with complex relationships and interconnections**. In a graph database, ***data is stored as nodes and edges, where nodes represent entities and edges represent the relationships between those entities.***

1.Graph databases are particularly well-suited for applications that require deep and complex queries, such as social networks, recommendation engines, and fraud detection systems. They can also be used for other types of applications, such as supply chain management, network and infrastructure management, and bioinformatics.

2.One of the main advantages of graph databases is their ability to handle and represent relationships between entities. This is because the relationships between entities are as important as the entities themselves, and often cannot be easily represented in a traditional relational database.

3.Another advantage of graph databases is their flexibility. Graph databases can handle data with changing structures. This makes them particularly useful for applications with rapidly changing data structures or complex data requirements.

Some popular graph databases include **Neo4j, OrientDB, and ArangoDB**. Out of which **Neo4j** is the most popular one.

Dept of CSE

As we all know the graph is a pictorial representation of data in the form of nodes and relationships which are represented by edges.

A graph database is a type of database used to represent the data in the form of a graph.

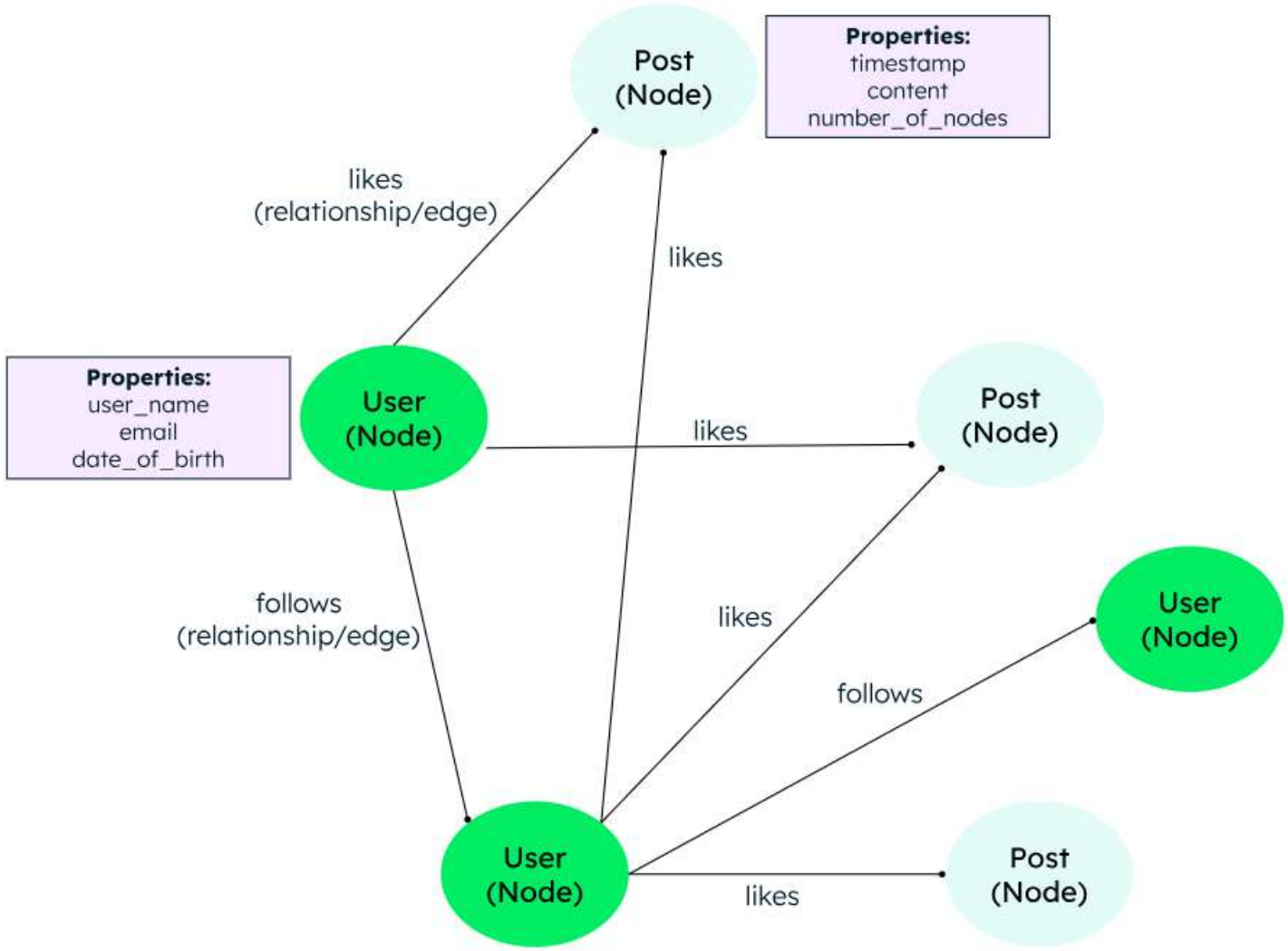
It has three components: **nodes, relationships, and properties**. These components are used to model the data.

The concept of a Graph Database is based on the **theory of graphs**. It was introduced in the year 2000. They are commonly referred to NoSql databases as data is stored using nodes, relationships and properties instead of traditional databases.

A graph database is very useful for **heavily interconnected data**. Here relationships between data are given priority and therefore the relationships can be easily visualized. They are useful in the fields of social networking, fraud detection

ACHARYA

Dept of CSE



A graph database stores data in the form of nodes and edges. Nodes typically store information about people, places, and things (like nouns), while edges store information about the relationships between the nodes.

Dept of CSE

The description of components are as follows:

•**Nodes:** represent the objects or instances. They are equivalent to a row in database. The node basically acts as a vertex in a graph. The nodes are grouped by applying a label to each member.
•**Relationships:** They are basically the edges in the graph. They have a specific direction, type and form patterns of the data. They basically establish relationship between nodes.
•**Properties:** They are the information associated with the nodes.

Types of Graph Databases:

•**Property Graphs:** These graphs are **used for querying and analyzing data by modelling the relationships among the data**. It comprises of vertices that has information about the particular subject and edges that denote the relationship. The vertices and edges have additional attributes called ***properties***.

•**RDF Graphs:** It stands for Resource Description Framework. It focuses **more on data integration**. They are used to represent complex data with well defined **semantics**. It is represented by three elements: two vertices, an edge that reflect the subject. Every vertex and edge is represented by URI(Uniform Resource Identifier).

ACHARYA

Dept of CSE

When to Use Graph Database?

•Graph databases should be used for heavily interconnected data.
•It should be used when amount of data is larger and relationships are present.
•It can be used to represent the cohesive picture of the data.

Example of Graph Database:

•Recommendation engines in E commerce use graph databases to provide customers with accurate recommendations, updates about new products thus increasing sales and satisfying the customer's desires.

•Social media companies use graph databases to find the "friends of friends" or products that the user's friends like and send suggestions accordingly to user.

•To detect fraud Graph databases play a major role. Users can create graph from the transactions between entities and store other important information. Once created, running a simple query will help to identify the fraud.

ACHARYA

Dept of CSE

Advantages of Graph Database:

•Potential advantage of Graph Database is establishing the relationships with external sources as well

•No joins are required since relationships is already specified.

•Query is dependent on concrete relationships and not on the amount of data.

•It is easy to manage the data in terms of graph.

•Flexible relationships: Graph databases are designed to handle complex relationships and interconnections between data elements. This makes them well-suited for applications that require deep and complex queries, such as social networks, recommendation engines, and fraud detection systems.

•High performance: Graph databases are optimized for handling large and complex datasets, making them well-suited for applications that require high levels of performance and scalability.

•Scalability: Graph databases can be easily scaled horizontally, allowing additional servers to be added to the cluster to handle increased data volume or traffic.

•Easy to use: Graph databases are typically easier to use than traditional relational databases. They often have a simpler data model and query language, and can be easier to maintain and scale.

ACHARYA

Dept of CSE

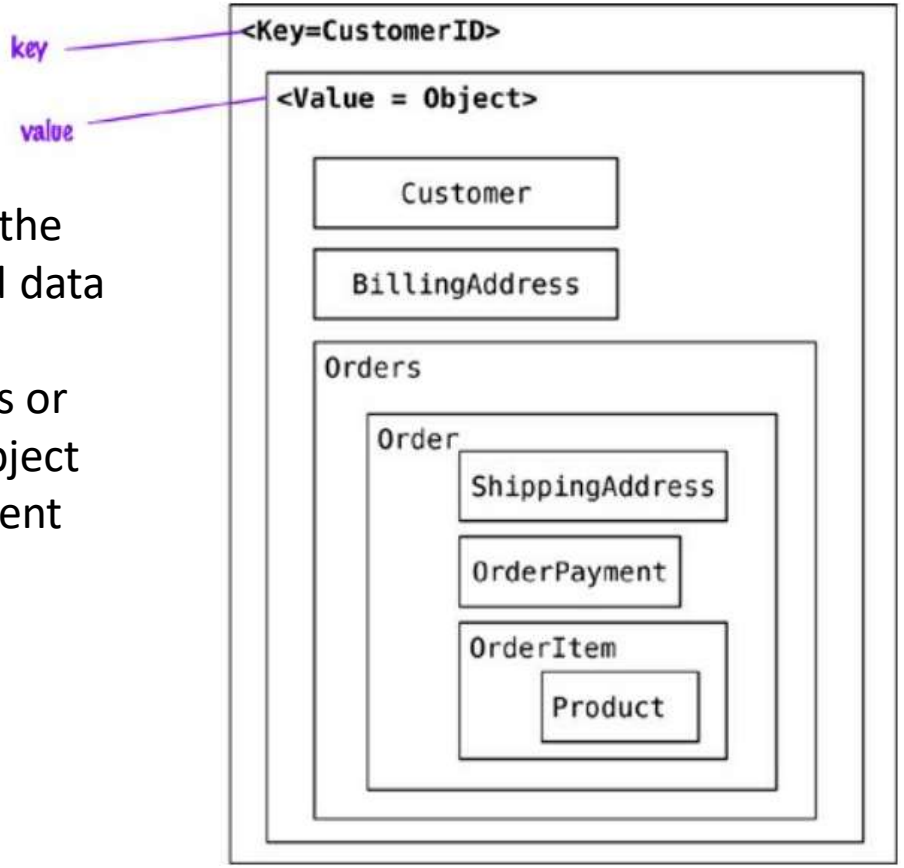
Disadvantages of Graph Database:

•Often for complex relationships speed becomes slower in searching.
•Limited use cases: Graph databases are not suitable for all applications. They may not be the best choice for applications that require simple queries or that deal primarily with data that can be easily represented in a traditional relational database.
•Specialized knowledge: Graph databases may require specialized knowledge and expertise to use effectively, including knowledge of graph theory and algorithms.

Overall, graph databases on NoSQL offer many advantages for applications that require complex and deep relationships between data elements. They are highly flexible, scalable, and performant, and can handle large and complex datasets. However, they may not be suitable for all applications, and may require specialized knowledge and expertise to use effectively.

Data Modeling

- When modeling data aggregates we need to consider how the data is going to be read as well as what are the side effects on the data related to those chosen aggregates.
- **Example:** Let's start with the model where all the data for the customer is embedded using a key-value store.



- In this scenario, the application can read the customers information and all the related data by using the key.
- If the requirements are to read the orders or products sold in each other, the whole object has to be read and then parsed on the client side to build the results.

Embed all the objects for customer and their orders.

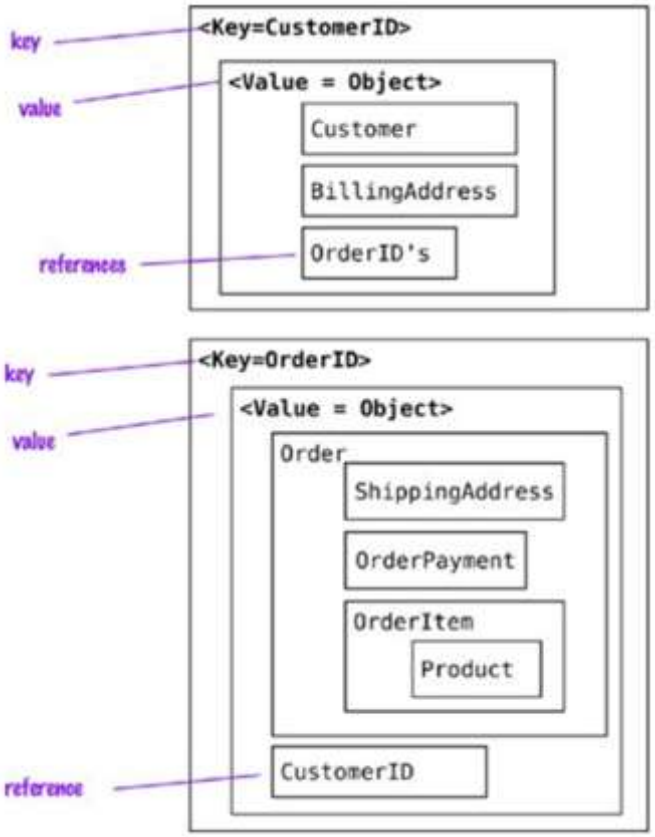
ACHARYA

Dept of CSE

When references are needed, we should change the data for the key- value store to document based store to split the value object into Customer and Order objects and then maintain these objects references each other.

We can now find the orders independently from the Customer, and with OrderId reference in the Customer, we can find all Orders for the Customer.

Using aggregates, this way allows for read optimization, but we have to push the OrderId reference into Customer for every new Order




```
# Customer object
{
"customerId": 1,
"customer": {
  "name": "Martin",
  "billingAddress": [{"city": "Chicago"}],
  "payment": [{"type": "debit","ccinfo": "1000-1000-1000-1000"}],
  "orders":[{"orderId":99}]
 }
}

# Order object
{
"customerId": 1,
"orderId": 99,
"order":{
  "orderDate":"Nov-20-2011",
  "orderItems":[{"productId":27, "price": 32.45}],
   "orderPayment":[{"ccinfo":"1000-1000-1000-1000",
           "txnId":"abelif879rft"}],
   "shippingAddress":{"city":"Chicago"}
   }
}
```

Customer is stored separately from Order

Dept of CSE

In document stores, since we can query inside documents, we can find all Orders for the Customer even removing references to Orders from the Customer object. This change allows us to not update the Customer object when orders are placed by the Customer

```
# Customer object
{
"customerId": 1,
"name": "Martin",
"billingAddress": [{"city": "Chicago"}],
"payment": [
  {"type": "debit",
  "ccinfo": "1000-1000-1000-1000"}
  ]
}

# Order object
{
"orderId": 99,
"customerId": 1,
"orderDate":"Nov-20-2011",
"orderItems":[{"productId":27, "price": 32.45}],
"orderPayment":[{"ccinfo":"1000-1000-1000-1000",
        "txnId":"abelif879rft"}],
"shippingAddress":{"city":"Chicago"}
}
```

- Aggregates can also be used to obtain analytics; for example, an aggregate update may fill in information on which Orders have a given Product in them.
- This denormalization of the data allows for fast access to the data we are interested in and is the basis for **Real Time Businees Intelligence** or **Real Time Analytics**: enterprises do not have to rely on end-of-the-day batch to populate data warehouses tables and generate analytics.
- Of course, only pre-packed analyses are possible through this approach

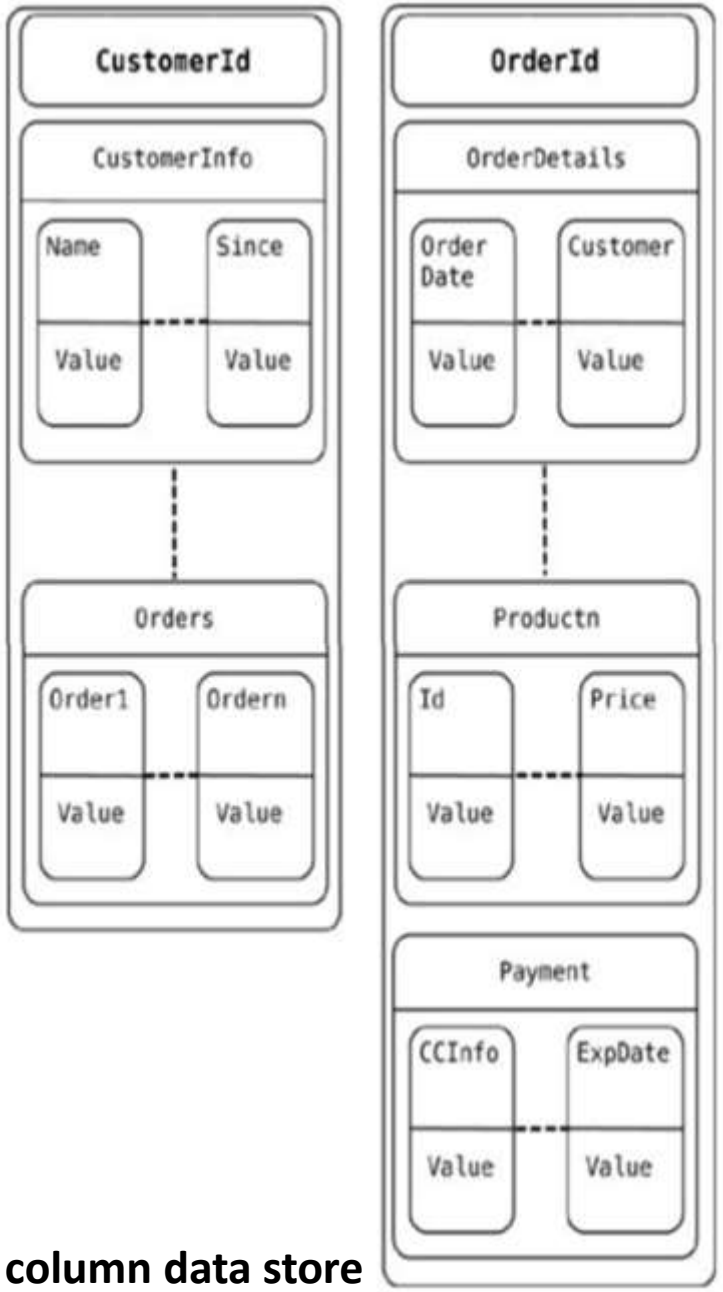
document store modeling

```
{
"itemid":27,
"orders":{99,545,897,678}
}
{
"itemid":29,
"orders":{199,545,704,819}
}
```

ACHARYA

Dept of CSE

- When using the column families to model the data, we can model the schema a little more. Obviously, there are multiple ways to model the data.
- In our example, one way is to store the Customer and Order in different column families.
- The reference to all the orders placed by the customer are in the Customer column family. Similar other denormalizations are generally done so that query (read) performance is improved



Conceptual view into a column data store

Dept of CSE

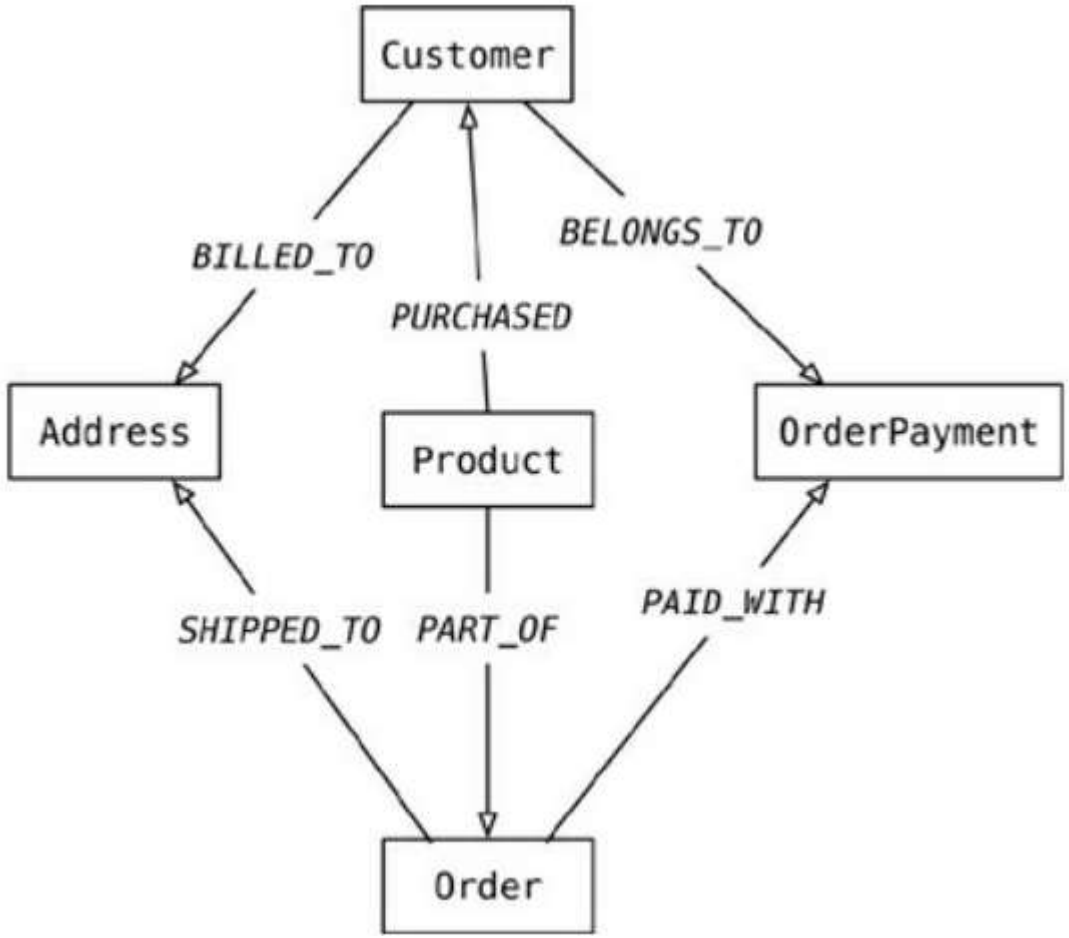
When using graph databases to model the same data, we model all objects as nodes and relations within them as relationships; these relationships have types and directional significance.

Each node has independent relationships with other nodes. These relationships have names like PURCHASED, PAID_WITH, or BELONGS_TO; these relationship names let you traverse the graph. Let's say you want to find all the Customers who PURCHASED a product with the name Refactoring Database. All we need to do is query for the product node Refactoring Databases and look for all the Customers with the incoming PURCHASED relationship

ACHARYA

Dept of CSE

This type of relationship traversal is very easy with graph databases. It is especially convenient when you need to use the data to recommend products to users or to find patterns in actions taken by users.



Graph model of e-commerce data