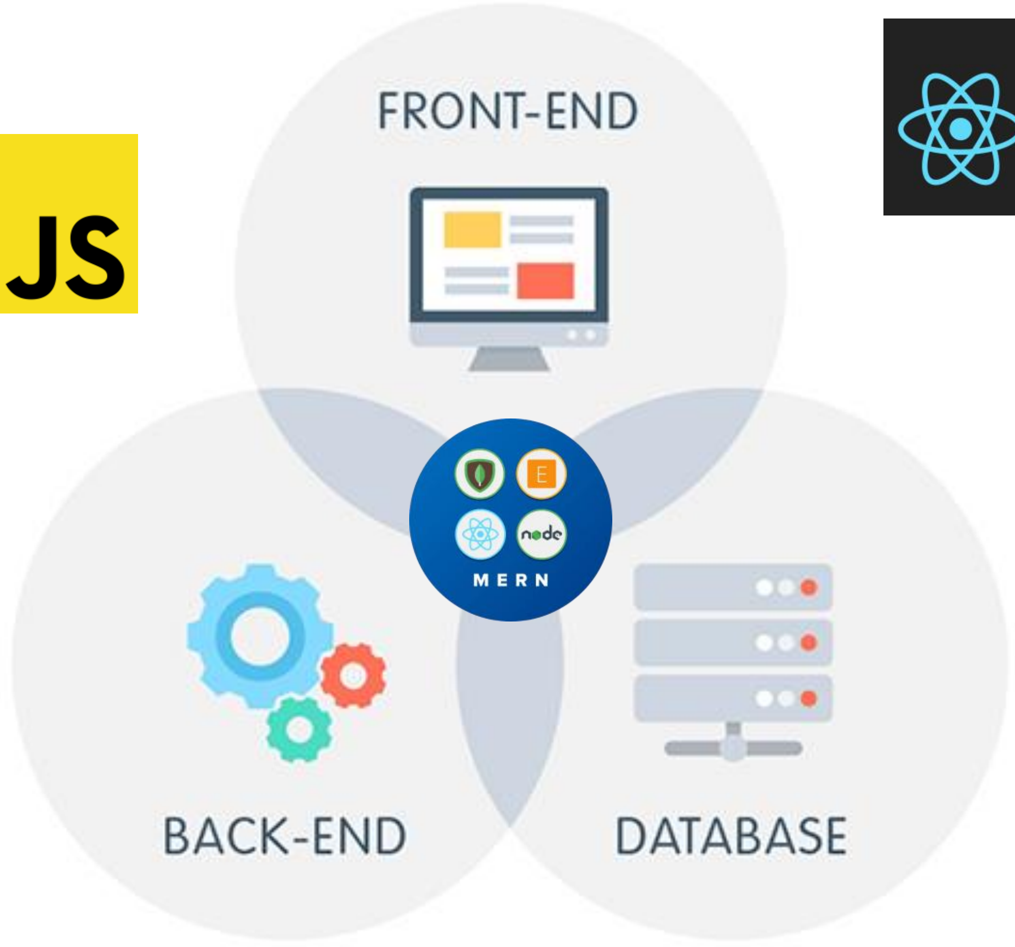
Working with Databases in Express.js

THAPA TECHNICAL YOUTUBE CHANNEL- NODEJS + DATABASE SERIES

Full-Stack Development

HTML CSS JS



node express

mongoDB MySQL PostgreSQL
redis cassandra

DataBase we will cover

MySQL

mongoDB

Drizzle ORM

mongoose

Prisma

THAPA TECHNICAL YOUTUBE CHANNEL- NODEJS + DATABASE SERIES

Introduction to Databases

- A database is a structured way to store and manage data.
- It helps organize data for easy access, retrieval, and updates.
- Databases store data like tables, documents, or key-value pairs.
- They are used in applications to handle user and system data.
- Databases ensure data is consistent and available when needed.
- Examples include MySQL, MongoDB, PostgreSQL, and Redis.

Why Do We Need Databases?

- **Data Integrity:** Ensures data consistency. For example, in banking, a transaction either happens fully or not at all.
- **Concurrency Management:** Discuss how multiple users can access the same data simultaneously without conflicts.
- **Backup & Recovery:** Automatic processes to save data and restore it after failures.
- **Scalability:** Mention how databases are designed to handle growth—from hundreds to millions of records.

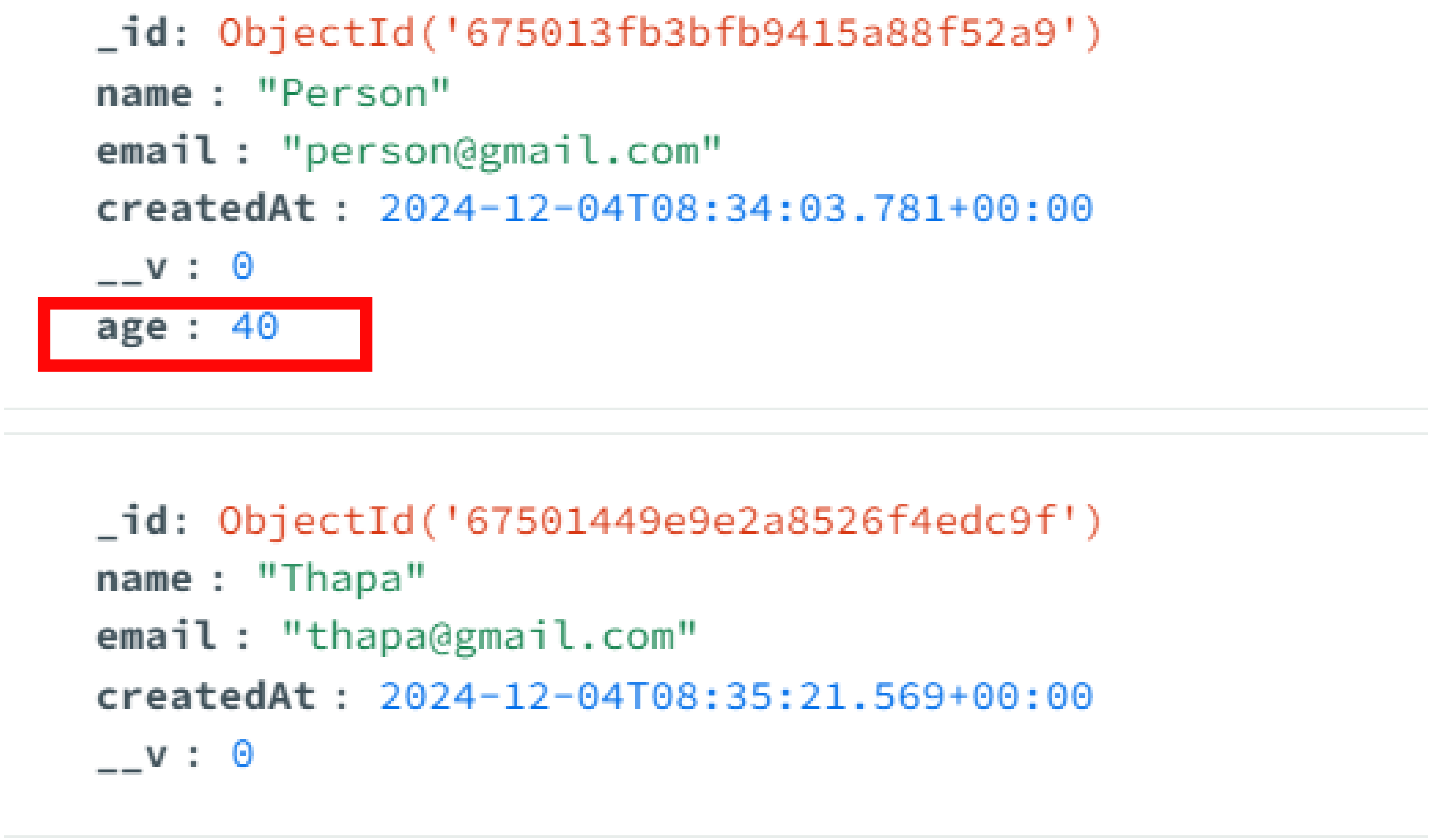
Types of Databases

SQL Databases	NoSQL Databases
Store data in **structured tables** with **rows and columns**.	Designed for **flexibility** with **unstructured, semi-structured**, or large-scale data.
Use **predefined schemas**, where the structure is fixed before data entry.	**Do not require a fixed schema**, making them ideal for dynamic data models.
Support complex queries using **SQL (Structured Query Language).**	Support complex queries using query languages or APIs specific to each NoSQL database, such as MongoDB's JSON-like queries, Cassandra's CQL (Cassandra Query Language), or Neo4j's Cypher for graph traversal.
Examples: MySQL, PostgreSQL, SQL Server, etc.	Examples based on format in which data is stored: • **Key-Value**: Redis, DynamoDB (e.g., caching or session data). • **Document**: MongoDB, CouchDB (e.g., JSON-like documents for user profiles). • **Graph**: Neo4j, ArangoDB (e.g., social networks or recommendation engines).

Introduction to MongoDB

- MongoDB is a NoSQL database that stores data in a flexible, JSON-like format called BSON (Binary JSON).
- It is document-oriented, meaning data is stored as documents in collections instead of rows and tables.
- MongoDB supports dynamic schemas, allowing fields to vary between documents in the same collection.
- MongoDB provides features like indexing, aggregation pipelines, and replication for high performance and reliability.
- Commonly used for applications requiring flexible data models

MongoDB Documents Example



```
_id: ObjectId('675013fb3bfb9415a88f52a9')
name : "Person"
email : "person@gmail.com"
createdAt : 2024-12-04T08:34:03.781+00:00
__v : 0
age : 40
```

```
_id: ObjectId('67501449e9e2a8526f4edc9f')
name : "Thapa"
email : "thapa@gmail.com"
createdAt : 2024-12-04T08:35:21.569+00:00
__v : 0
```

Key Concepts of MongoDB

- **Collections**
  - A collection is a group of documents in MongoDB, similar to a table in relational databases.
  - Collections do not enforce a fixed schema, allowing documents to have different fields.
  - Example: A "users" collection can store data about different users.
- **Documents**
  - A document is the basic unit of data in MongoDB, stored in a JSON-like format.
  - Documents contain key-value pairs, where each key is a field, and the value is the data.

  -
    ```
    {
      "name": "John Doe",
      "email": "john@example.com",
      "age": 25
    }
    ```

THAPA TECHNICAL YOUTUBE CHANNEL- NODEJS + DATABASE SERIES

Key Concepts of MongoDB

- **BSON (Binary JSON)**
  - BSON is the binary-encoded format MongoDB uses to store data.
  - It extends JSON by adding data types like Date, Binary, and others.
  - BSON is efficient for data storage and supports faster parsing compared to JSON.
  - Example: A BSON document might store a Date field that is not natively supported in JSON.

- This is not a MongoDB course. Hence, we are just learning through surface essential for now. You can learn in detail from our course.

THAPA TECHNICAL YOUTUBE CHANNEL- NODEJS + DATABASE SERIES

Installing and Setting Up MongoDB

- **Download MongoDB**
  - You can download **MongoDB Community** version from this link: https://www.mongodb.com/docs/manual/administration/install-community/
  - Choose the version compatible with your operating system (Windows, macOS, Linux).
- Install MongoDB
  - Follow the installer's instructions to install MongoDB on your system.
  - MongoDB Compass is also installed by default. If it's not installed, then you can install manually by searching online.
  - MongoDB Compass is just a software for interacting with your MongoDB database.
  - Mongosh is a command-line tool for interacting with MongoDB programmatically.
    - This is not installed by default, but you can install yourself if you want.

MongoDB Drivers

- MongoDB drivers are language-specific libraries that allow applications to interact with MongoDB databases.
- They provide APIs for connecting, querying, and managing data within MongoDB.
- MongoDB drivers are available for various programming languages, including Node.js, Python, Java, C++, and more.
- The official MongoDB Node.js driver is mongodb, used for connecting to and performing CRUD operations in Node.js.
- https://www.mongodb.com/docs/drivers/
- As we are using Node.js, we will install **mongodb** npm package.

Introduction to Mongoose

- Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.js.
- It provides a higher-level abstraction over the native MongoDB driver, allowing developers to define schemas and interact with MongoDB using objects.
- Mongoose allows you to define models based on schemas which represent the structure of documents in a MongoDB collection.
- It provides built-in methods for CRUD operations (Create, Read, Update, Delete) and validates data before saving it to the database.
- Mongoose supports middleware (also called hooks) for functions like validation, pre-save, and post-save, allowing developers to perform tasks like encryption, logging, or data transformation.
- It helps to manage relationships between data, providing support for features like population (joining data from different collections).

Introduction to MySQL

- MySQL is a widely used open-source relational database management system (RDBMS).
- It stores data in tables, with rows representing individual records and columns representing data attributes.
- MySQL uses Structured Query Language (SQL) for managing and manipulating data.
- It is known for its reliability, performance, and ease of use, making it suitable for both small and large-scale applications.
- MySQL is cross-platform, meaning it works on different operating systems such as Windows, Linux, and macOS.

Key Concepts

- A **table** in MySQL is a collection of data organized in rows and columns. It is similar to a spreadsheet where each row is a record, and each column holds data attributes of that record.
- A **row** (also called a record or tuple) represents a single, data entry in a table. Each row in a table has a unique identifier, often the primary key.
- A **column** is a vertical structure in a table that holds a specific type of data. Each column has a data type, such as integers, strings, or dates, and each column represents an attribute of the data stored in the table (e.g., "name", "age", "email").
- Each **row** in a table contains values for each of the columns, which together make up the full record.
- Tables in MySQL are defined by a schema that specifies the names and types of columns, along with any constraints like primary keys or unique values.
- **Primary Key** is a special column or combination of columns that uniquely identifies each row in a table.
- **Foreign Key** is a column that creates a relationship between two tables, linking the row in one table to a row in another table.

Installing MySQL

- Download the MySQL installer from the official MySQL website (https://dev.mysql.com/downloads/).
- Choose the appropriate version for your operating system (Windows, macOS, or Linux).
- Run the installer and follow the on-screen instructions to install MySQL on your system.
- During installation, choose the "Developer Default" setup type to install essential tools like MySQL Server, Workbench, and command-line utilities.
- Set a root password during the installation process. This password will be used to access the MySQL root account.
- Install MySQL Workbench from https://dev.mysql.com/downloads/workbench/
- Open MySQL Workbench or use the command line interface (CLI) to connect to the MySQL server:
  - Use the command mysql -u root -p to login using the root account and enter the password you set during installation.

Using MySQL with Node.js

- To connect to MySQL from a Node.js application, use the mysql2 or mysql package, which provides a simple and efficient way to interact with MySQL databases.
- For other SQL based databases, you can search online.
- First, install the mysql2 package by running:
  - npm install mysql2