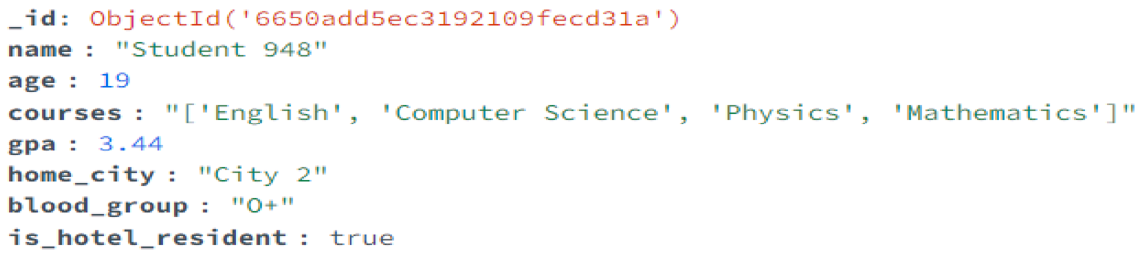
Documents,Collections,Database

Documents:

In Mongodb, documents are the fundamental unit of data storage. They act like containers holding information in a flexible schema. Here's a breakdown of what documents are in MongoDB

Example:



```
_id: ObjectId('6650add5ec3192109fecd31a')
name : "Student 948"
age : 19
courses : "['English', 'Computer Science', 'Physics', 'Mathematics']"
gpa : 3.44
home_city : "City 2"
blood_group : "O+"
is_hotel_resident : true
```

- ➢ **_id** is a special field that uniquely identifies each document by default.
- ➢ Other fields like **name, age**, and **courses** hold specific data about the document.
- ➢ Nested documents like **home_city** can be used to group related information.

Benefits:

- ○ The flexible schema of documents allows you to store data without rigid table structures like in relational databases.
- ○ Documents can efficiently represent complex data hierarchies.

Documents can be represented as JSON

In MongoDB, documents are the fundamental unit of data storage. Each document represents a record and uses a JSON-like structure. This structure allows for storing various data types within a document and enables easy manipulation and querying of data.

So, while JSON is not exclusive to documents, it serves as a powerful and widely used way to represent them, especially in NoSQL databases like MongoDB.

Collections:

In MongoDB, collections act like containers that hold documents, which are essentially JSON-like structures that store your actual data. You can think of collections as similar to tables in relational databases, but with some key differences.

Examples:

- _id
- name
- email
- orders

Database:

- MongoDB groups collections into databases.
- A single instance of MongoDB can host several databases, each grouping together zero or more collections.
- A database has its own permissions, and each database is stored in separate files on disk.

Datatype:

MongoDB stores data using a format called BSON (Binary JSON), which extends JSON with additional data types. Here's a rundown of the common data types you'll encounter in MongoDB

Basic Types:

- **String:** The most common type, used for textual data. It must be valid UTF-8.
- **Integer:** Stores whole numbers, can be 32-bit or 64-bit depending on your server configuration.
- **Double:** Stores floating-point numbers with higher precision.
- **Boolean:** Represents true or false values.

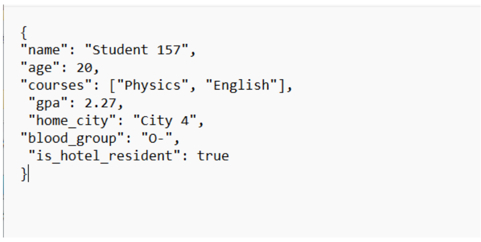
Complex Types:

- **Array:** Stores an ordered list of values of various data types.
- **Object:** Used for embedding documents within documents, helpful for representing hierarchical data.
- **Date:** Stores dates and times as milliseconds since the Unix epoch.
- **ObjectId:** A unique 12-byte identifier automatically generated for each document when inserted.

Other Data Types:

- **Null:** Represents missing or undefined values.
- **Binary:** Stores raw binary data like images or files.
- **Code:** Allows storing JavaScript code within a document for server-side execution.
- **Symbol:** Similar to strings but less common, often used for specific data types in some programming languages.
- **Timestamp:** Represents a specific point in time with nanosecond precision.
- **Decimal128:** Stores high-precision decimal numbers.

Example:



```
{
"name": "Student 157",
"age": 20,
"courses": ["Physics", "English"],
 "gpa": 2.27,
 "home_city": "City 4",
"blood_group": "O-",
 "is_hotel_resident": true
}
```

This JSON document uses:

- **String:** "name", "courses", "home_city", "blood_group"
- **Integer:** "age"
- **Double:** "gpa"
- **Boolean:** "is_hotel_resident"
- **Array:** "courses"