Q1. What is MongoDB? Explain non-relational databases. When to use MongoDB?

MongoDB:
MongoDB is a **NoSQL, document-oriented database** that stores data in **JSON-like documents** (BSON format). It is **schema-less** and supports high performance and scalability.

Non-relational databases:

- Do **not use tables** like relational databases.
- Store data in **documents, key-value pairs, wide-columns, or graphs**.
- Flexible schema allows dynamic and unstructured data.

When MongoDB is preferred over SQL:

1. Rapid development with **changing data structures**
2. Handling **large volumes of unstructured or semi-structured data**
3. Need for **horizontal scalability** and distributed systems
4. Real-time analytics or content management systems

Q2. Features of MongoDB

Feature	Description
Document-oriented	Stores data in JSON/BSON documents
Schema-less	Flexible, no fixed schema required
High Performance	Optimized for read and write operations
Scalable	Supports horizontal scaling via sharding
Indexing	Supports indexes to improve query performance

Replication	Provides data redundancy and high availability via replica sets
Aggregation	Supports powerful aggregation operations
Built-in support for GridFS	Stores large files like images, videos, etc.

Q3. Connect MongoDB to Python and create a database & collection

```
# Install pymongo if not already installed
# pip install pymongo

from pymongo import MongoClient

# Connect to MongoDB
client = MongoClient("mongodb://localhost:27017/")

# Create a database
db = client["SchoolDB"]

# Create a collection
collection = db["Students"]

print("Database and collection created successfully!")
```

Q4. Insert one record, many records, and use `find()` and `find_one()`

```
# Insert one record
student1 = {"Name": "Alice", "Age": 20, "Grade": "A"}
collection.insert_one(student1)

# Insert many records
students = [
```

```
    {"Name": "Bob", "Age": 21, "Grade": "B"},
    {"Name": "Charlie", "Age": 22, "Grade": "A"},
    {"Name": "David", "Age": 23, "Grade": "C"}
]
collection.insert_many(students)

# Retrieve one record
print("One record:", collection.find_one())

# Retrieve all records
print("All records:")
for record in collection.find():
    print(record)
```

Q5. Using `find()` to query MongoDB

- `find()` returns a **cursor object** containing documents that match the query.
- Example: Find students with Grade "A".

```
query = {"Grade": "A"}
results = collection.find(query)

print("Students with Grade A:")
for student in results:
    print(student)
```

- `find_one()` returns **the first matching document**.

Q6. Explain `sort()` method

- `sort()` is used to **sort the result of a query**.

- Syntax: `collection.find().sort("fieldname", direction)`
    - `1` → Ascending
    - `-1` → Descending

Example: Sort students by Age ascending

```
for student in collection.find().sort("Age", 1):
    print(student)
```

Q7. Why `delete_one()`, `delete_many()`, and `drop()` are used

Method	Purpose	Example
`delete_one()`	Deletes the **first document** that matches the filter	`collection.delete_one({"Name": "Alice"})`
`delete_many()`	Deletes **all documents** that match the filter	`collection.delete_many({"Grade": "C"})`
`drop()`	Deletes the **entire collection** permanently	`collection.drop()`
