**✅ Section 1: SQL – Relational Databases (e.g., MySQL, PostgreSQL)**

**🔹 Basics & Fundamentals**

1. What is a relational database?
2. What is a primary key and foreign key?
3. What are the different types of joins? Explain with examples.
4. What is normalization? Explain different normal forms.
5. What is denormalization? When would you use it?
6. What is the difference between WHERE and HAVING?
7. What are constraints in SQL? (e.g., NOT NULL, UNIQUE, CHECK, DEFAULT, FOREIGN KEY)
8. What is the difference between CHAR and VARCHAR?

**🔹 Query Writing**

1. Write a SQL query to find the second highest salary from a table.
2. Write a query to find all employees who joined in the last 3 months.
3. Write a query to get the count of employees department-wise.
4. Write a query to find duplicate records in a table.
5. How to update values in one table based on another table?
6. Write a query to find customers who haven’t placed any orders.

**🔹 Aggregate Functions & Subqueries**

1. What are aggregate functions in SQL? (e.g., SUM, AVG, COUNT, MAX, MIN)
2. What is a subquery? What are correlated subqueries?
3. Write a query using a subquery inside SELECT, WHERE, and FROM.

**🔹 Joins Deep Dive**

1. What is the difference between INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN?
2. What is a SELF JOIN? Give an example.
3. How can we join more than two tables?

**🔹 Indexes & Performance**

1. What is an index? How does it work?
2. What is a composite index?
3. What are the pros and cons of using indexes?
4. When would an index **not** be used even if it exists?

**🔹 Transactions & Locks**

1. What is a transaction? What are ACID properties?
2. What is the difference between COMMIT, ROLLBACK, and SAVEPOINT?
3. What is a deadlock? How to avoid it?
4. What is the difference between pessimistic and optimistic locking?

**🔹 Views, Triggers & Stored Procedures**

1. What is a view? How is it different from a table?
2. What is a trigger? When do we use it?
3. What are stored procedures and functions? How are they different?
4. What is the use of WITH clause (CTE - Common Table Expressions)?

**🔹 Django + SQL Integration**

1. How does Django ORM translate to SQL?
2. How do you write raw SQL queries in Django?
3. How can you optimize queries in Django? (select\_related, prefetch\_related)
4. How can you prevent N+1 query problem in Django?

**✅ Section 2: NoSQL (e.g., MongoDB, Redis)**

**🔹 Basics & Differences**

1. What is NoSQL? How is it different from SQL databases?
2. Types of NoSQL databases (Document, Key-Value, Column, Graph)?
3. When would you prefer NoSQL over SQL?
4. What are the advantages and disadvantages of NoSQL databases?

**🔹 MongoDB – Document Database**

1. What is a document in MongoDB?
2. What is the structure of a MongoDB document?
3. How do you insert, update, and delete documents in MongoDB?
4. What are embedded documents and references?
5. How do you perform queries in MongoDB (find, filter, projection)?
6. What is indexing in MongoDB?

**🔹 MongoDB vs SQL**

1. How do joins work in MongoDB? (using $lookup)
2. What are the tradeoffs of embedding vs referencing?
3. How does schema design differ in MongoDB compared to SQL?

**🔹 Redis – Key-Value Store**

1. What is Redis used for?
2. How does Redis store data? (String, List, Set, Hash)
3. How does Redis help in caching?
4. How do you set expiration for a key in Redis?
5. How is Redis different from traditional NoSQL DBs?

**🔹 Django + NoSQL**

1. Can Django work with MongoDB or Redis? How?
2. Use cases where you integrated Redis with Django (e.g., caching, Celery).

**✅ Bonus – Real-World Scenarios**

1. How would you design a database for a blogging platform?
2. If you have millions of records and slow queries, how would you optimize?
3. How to handle schema changes in production for SQL databases?
4. How would you migrate data from SQL to NoSQL?
5. What issues have you faced with DB performance in your past projects?

**✅ SQL Topics – Fully Covered**

| **Topic Area** | **Covered?** | **Notes** |
| --- | --- | --- |
| Basic SQL concepts | ✅ | Relational DBs, primary/foreign key, data types |
| Query writing (SELECT, JOINs) | ✅ | Real-world problems and join types |
| Aggregate functions & subqueries | ✅ | Includes both scalar and correlated subqueries |
| Advanced joins | ✅ | SELF JOIN, multi-table joins |
| Indexes & performance tuning | ✅ | How and when to use indexes, composite indexes |
| Transactions & ACID | ✅ | Deadlocks, locks, optimistic/pessimistic |
| Views, triggers, stored procedures | ✅ | Definitions, use cases |
| Django ORM & SQL | ✅ | Query optimization, raw SQL, select\_related, etc. |
| Real-world SQL scenarios | ✅ | Blog DB design, schema changes, millions of records |

**✅ NoSQL Topics – Fully Covered**

| **Topic Area** | **Covered?** | **Notes** |
| --- | --- | --- |
| NoSQL basics & classification | ✅ | Document, Key-Value, Column, Graph |
| MongoDB CRUD operations | ✅ | Insert, update, delete, querying |
| Embedded vs referenced docs | ✅ | Best practices and trade-offs |
| Indexing in MongoDB | ✅ | Performance, lookup usage |
| MongoDB vs SQL comparisons | ✅ | Joins, schema design |
| Redis concepts | ✅ | Data types, caching, expiration |
| Django with NoSQL (Mongo/Redis) | ✅ | Use cases, integration techniques |

**✅ Interview-Oriented Content**

| **Type** | **Covered?** | **Notes** |
| --- | --- | --- |
| Theoretical questions | ✅ | All key theory for both SQL and NoSQL |
| Query writing (hands-on) | ✅ | Frequently asked patterns |
| Performance & optimization | ✅ | Indexes, Django N+1, Redis caching |
| System design thinking | ✅ | Real-world DB design questions |
| Django + DB integration | ✅ | ORM, caching, Celery, raw SQL, query tuning |

**❗ What is *not* included (because it's NOT asked in interviews):**

* Internal DB engine architecture
* Deep mathematical optimization theory
* Admin/infra level topics (e.g., setting up replication manually)
* GraphQL or proprietary DB query languages