1. **What database models do you know?**

Relational databases – a bunch of tables with their relationships to each other. Models that implement this partially or fully include:

Network model

Relational model

Entity-relationship model

Entity-attribute-value model

Start schema

Non-relation databases – use documents with all object data, do not store relationships. Models for it include:

Object model

Hierarchical database model

Document model

Key value model

1. **Which are the main functions performed by a Relational Database Management System (RDBMS)?**

An RDBMS handles the creation, deletion and alteration of tables and the relationship between them. It also provides capability for adding, changing, deleting and viewing the data stored in those tables and relationships. It usually has support for SQL and transactions.

1. **Define what is "table" in database terms.**

A table is data arranged in rows and columns – rows represent an actual entry for the specific table, while columns have a name and type (int, char, string, image, etc).

1. **Explain the difference between a primary and a foreign key.**

Primary key is a column in a table that uniquely identifies its rows. It is usually a number, though string indexing is also possible (albeit slower). Two entries are considered distinct (different) only if their primary keys are different. Primary keys can be composed of several columns.

Foreign keys are columns in a table that uniquely identify a row in another table. They, together with primary keys, create the unique relationships between tables in an RDBMS system.

1. **Explain the different kinds of relationships between tables in relational databases.**

One-to-One – Each primary key in the given table has exactly one corresponding foreign key in another table. One can often (but not always) avoid creating this explicit relationship by including the desired field in the primary key table.

One-to-Many – A single record in a table has many corresponding entries in another table and is also the most common relationship schema.

Many-to-Many – Records in the first table have many corresponding entries in the second table and vice versa. Implementation in SQL requires the creation of an additional table to represent the relationship.

1. **When is a certain database schema normalized? What are the advantages of normalized databases?**

Normalization of a database schema removes repeating data. There are different levels of normalization and those are:

1st normal form – Data is stored in tables, fields in the rows are atomic values, there are no repetitions within a single row, a primary key is defined for each table.

2nd normal form – Retains all requirements from 1st, adds that there are no columns that do not depend on part of the primary key.

3rd normal form – Retains all requirements from 2nd, adds that the only dependencies between columns are of the type “a column depends on the primary key”.

4th normal form – Retains all requirements from 3rd, adds that there is one column at most in each table that can have many possible values for a single key.

The main advantage of normalized databases is that they remove repeating data. They also make it easier to change data that exists in several places at once by abstracting the specific information represented by the data in a unique identifier.

1. **What are database integrity constraints and when are they used?**

Integrity constraints ensure the integrity of a database by enforcing data rules which cannot be violated. Usually used when creating, altering or removing data from a table.

1. **Point out the pros and cons of using indexes in a database.**

Pros – speed up searching for data.

Cons – delay insertion/deletion of data.

1. **What's the main purpose of the SQL language?**

The main purpose can be split into two distinct tasks. DDL or data definition language is used to facilitate the creation, alteration and deletion of tables and other objects within the database. DML or data manipulation language is used to search for, retrieve, insert, modify or delete table data (rows).

1. **What are transactions used for? Give an example.**

Transactions are used to guarantee that the table is always in a stable state and that no complex change will be executed unless all its steps can fully complete. For example, in a banking database, a transaction would ensure that a transfer from one account to another will never catastrophically fail mid execution – meaning, we will never reach the state where money has been withdrawn from one account but was never deposited in its new target account.

1. **What is a NoSQL database?**

NoSQL is a document based model (non-relational). Such a schema still supports CRUD operations, indexing and querying, and concurrency and transactions. It can append/retrieve information extremely quickly. In a NoSQL database, data is stored in documents and each document is a single entry (which in an RDBMS system might be spread over several tables with relationships between each other). Documents have no fixed structure.

1. **Explain the classical non-relational data models.**

Document model – Set of documents, for example JSON strings or XML sheets.

Key-value model – Set of key value pairs, each entry has a set number of keys related to it which have different values depending on the entry.

Hierarchical key-value – Similar to the previous one only allows for more complex key-value relationship by allowing for high-level keys – for example, an address key can be internally composed of street name, apartment number, floor number, postal code, etc.

Object model – Can be similar to the hierarchical key-value (or be considered a more complex implementation of the same). Essentially, it’s a set of OOP style objects.

1. **Give few examples of NoSQL databases and their pros and cons.**

Redis – ultra fast in-memory data structures, it holds the entire database dataset in memory. It performs extremely fast read and write operations compared to most standard databases. Its main drawback is that data durability is not as high as in traditional databases that use the hard disk.

MongoDB – uses JSON-like documents for data storage making the integration of data in certain types of applications easier and faster. The database is quite RAM heavy and 32bit servers (blessedly quite uncommon nowadays) are quite likely to experience crashes and heavy slowdowns. 64bit systems can also suffer from the same problems when they have insufficient RAM.