# What database models do you know?

* Hierarchical Model
* Network Model
* Relational Model
* Object/Relational Model
* Object-Oriented Model
* Semistructured Model
* Associative Model
* Entity-Attribute-Value (EAV) data model
* Context Model

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

In broader terms:

- Present the data to the user as relations (a presentation in tabular form, i.e. as a collection of tables with each table consisting of a set of rows and columns);

- Provide relational operators to manipulate the data in tabular form.

More specific rules for were provided by Edgar Codd - Codd's twelve rules, designed to define what is required from a database management system in order for it to be considered relational. Link to source: <http://en.wikipedia.org/wiki/Codd%27s_12_rules>

# Define what is "table" in database terms.

A table is a collection of related data held in a structured format within a database. It consists of fields (columns), and rows.

In relational databases and flat file databases, a table is a set of data elements (values) using a model of vertical columns (which are identified by their name) and horizontal rows, the cell being the unit where a row and column intersect. A table has a specified number of columns, but can have any number of rows. Each row is identified by the values appearing in a particular column subset which has been identified as a unique key index.

Table is another term for relation; although there is the difference in that a table is usually a multiset (bag) of rows where a relation is a set and does not allow duplicates. Besides the actual data rows, tables generally have associated with them some metadata, such as constraints on the table or on the values within particular columns

# Explain the difference between a primary and a foreign key.

A primary key is a field or combination of fields that uniquely identify a record in a table, so that an individual record can be located without confusion.

A foreign key (sometimes called a referencing key) is a key used to link two tables together. Typically you take the primary key field from one table and insert it into the other table where it becomes a foreign key (it remains a primary key in the original table).

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

**There are three types of relationships:**

* **One-to-one** - Both tables can have only one record on either side of the relationship. Each primary key value relates to only one (or no) record in the related table.
* **One-to-many -** The primary key table contains only one record that relates to none, one, or many records in the related table.
* **Many-to-many -** Each record in both tables can relate to any number of records (or no records) in the other table.

# When is a certain database schema normalized? What are the advantages of normalized databases?

Normalization is the process of organizing data in a database. This includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency.

Redundant data wastes disk space and creates maintenance problems. If data that exists in more than one place must be changed, the data must be changed in exactly the same way in all locations.

# What are database integrity constraints and when are they used?

Three types of integrity constraints are an inherent part of the relational data model: entity integrity, referential integrity and domain integrity:

* **Entity integrity -** concerns the concept of a primary key. Entity integrity is an integrity rule which states that every table must have a primary key and that the column or columns chosen to be the primary key should be unique and not null.
* **Referential integrity -** concerns the concept of a foreign key. The referential integrity rule states that any foreign-key value can only be in one of two states. The usual state of affairs is that the foreign key value refers to a primary key value of some table in the database. Occasionally, and this will depend on the rules of the data owner, a foreign-key value can be null. In this case we are explicitly saying that either there is no relationship between the objects represented in the database or that this relationship is unknown.
* **Domain integrity -** specifies that all columns in relational database must be declared upon a defined domain. The primary unit of data in the relational data model is the data item. Such data items are said to be non-decomposable or atomic. A domain is a set of values of the same type. Domains are therefore pools of values from which actual values appearing in the columns of a table are drawn.

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

There are three main **advantages** to using an index-organized table:

* + Increased performance - There is no need to access a row in the database from an index structure, so you can reduce the total number of I/O operations needed to retrieve data.
  + Reduced table space - Because you do not need to link to a row in a table, there is no need to store the row ID in the index. The overall space required for the table is reduced.
  + Presorted data: The data in the leaf nodes is already sorted by the value of the primary key.

Index-organized tables also have **disadvantages**:

* + You must have a primary key on the table with a unique value.
  + You cannot have any other indexes on the data.
  + You cannot partition an index-organized table.
  + An index-organized table cannot be a part of a cluster.

# What's the main purpose of the SQL language?

SQL is a query language designed for organizing, managing, developing and querying large relational databases over computer networks. The purpose of the SQL language is specific to working with relational databases. It cannot be used to create stand-alone computer applications or operating systems. SQL is often referred to as a sub-language, since it can be used within other languages or applications.

# What are transactions used for? Give an example.

A transaction, in the context of a database, is a logical unit that is independently executed for data retrieval or updates.

Example: A transfer of funds from one bank account to another, even involving multiple changes such as debiting one account and crediting another, is a single transaction.

# What is a NoSQL database?

A NoSQL or Not Only SQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. Motivations for this approach include simplicity of design, horizontal scaling and finer control over availability. The data structure differs from the RDBMS, and therefore some operations are faster in NoSQL and some in RDBMS. NoSQL databases are increasingly used in big data and real-time web applications.

# Explain the classical non-relational data models.

In a **hierarchical model**, data is organized into a tree-like structure, implying a single parent for each record. A sort field keeps sibling records in a particular order. Hierarchical structures were widely used in the early mainframe database management systems.

The **network model** expands upon the hierarchical structure, allowing many-to-many relationships in a tree-like structure that allows multiple parents. It was the most popular before being replaced by the relational model.

In an **inverted file** or inverted index, the contents of the data are used as keys in a lookup table, and the values in the table are pointers to the location of each instance of a given content item. This is also the logical structure of contemporary database indexes, which might only use the contents from a particular columns in the lookup table. The inverted file data model can put indexes in a second set of files next to existing flat database files, in order to efficiently directly access needed records in these files.

# Give few examples of NoSQL databases and their pros and cons.

Examples are MongoDB, Cassandra, HBase, Neo4j.

* The Benefits of NoSQL

When compared to relational databases, NoSQL databases are more scalable and provide superior performance, and their data model addresses several issues that the relational model is not designed to address:

* Large volumes of structured, semi-structured, and unstructured data
* Agile sprints, quick iteration, and frequent code pushes
* Object-oriented programming that is easy to use and flexible
* Efficient, scale-out architecture instead of expensive, monolithic architecture
* Cons against NoSQL data:
* It’s very hard to move data out from one NoSQL to some other system, even other NoSQL. There is a very hard lock in when it comes to NoSQL. If you ever have to move to another database, you have basically to re-implement a lot of your applications from scratch.
* There is no standard way to access a NoSQL data store.All tools that already exist for SQL has to be recreated to each of the NoSQL databases. This means that it will always be harder to access data in NoSQL than from SQL.