**Database Systems Overview Homework**

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

**Hierarchical Model and Network Model** - organizes data in a tree structure.

**Relational Model -** the data and relations between them are organized in tables.

**Object/Relational Model -** add new object storage capabilities to the relational systems

**Object-Oriented Model -** add database functionality to object programming languages.

**Semi-structured Model -** the information that is normally associated with a schema is contained within the data, which is sometimes called ``self-describing''

**Associative Model -** divides the real-world things about which data is to be recorded into two sorts: Entities or Associations

**Entity-Attribute-**Value (EAV) data model - involves a single table with three columns, an entity, an attribute and a value for the attribute.

**Context Model -** combines features of all the above models.

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

Setting and Unsetting a Connection, Defining Transactions, Performing Insert Operations, Performing Update Operations on the Database, Performing Delete Operations on the Database, Performing Query Operations and more.

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

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 multi-set (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.

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

Primary key uniquely identifies each row in a table whereas a foreign key is used to establish and enforce a link between the data in two tables to control the data that can be stored in the foreign key table. A table can contain only one primary key constraint and it cannot exceed 16 columns. A foreign key doesn’t have these constrains.

1. **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. They're like spouses—you may or may not be married, but if you are, both you and your spouse have only one spouse. Most one-to-one relationships are forced by business rules and don't flow naturally from the data. In the absence of such a rule, you can usually combine both tables into one table without breaking any normalization rules.
* **One-to-many -** The primary key table contains only one record that relates to none, one, or many records in the related table. This relationship is similar to the one between you and a parent. You have only one mother, but your mother may have several children.
* **Many-to-many -** Each record in both tables can relate to any number of records (or no records) in the other table. For instance, if you have several siblings, so do your siblings (have many siblings). Many-to-many relationships require a third table, known as an associate or linking table, because relational systems can't directly accommodate the relationship.

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

The database schema is normalized when repeating data is removed by dividing a database into two or more tables and defining relationships between them. In a normalized database additions, deletions, and modifications of a field can be made in just one table and then propagated through the rest of the database via the defined relationships.

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

The database integrity constraints are conditions, which ensure that the data in a database is right (accurate) and consistent (in good condition). They are used so all instances of the Database can correctly model the real world.

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

**Pros:**

Indexes are sort of a bookmark within a table or a record, to mark a record, such as a primary key. They allow the database application to find data fast, without reading the whole table. Somewhat like using a bookmark in a book. Index can avoid accessing the table & satisfy the SELECT, which provides result set faster. Indexes are used to enforce unique and primary key constraints. Indexes help reduce lock contention, e.g. during the child table update where a foreign key exists.

**Cons**:  
  
Indexes increase the disk space requirements of the database, they slow down dml (inserts, updates and deletes), they increase the maintenance needs of your database, they may make your queries slower instead of faster.

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

To manage data held in a [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) or in other words designed for organizing, managing, developing and querying large relational databases over computer networks.

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

Transactions in a database environment have two main purposes:

* To provide reliable units of work that allow correct recovery from failures and keep a database consistent even in cases of system failure, when execution stops (completely or partially) and many operations upon a database remain uncompleted, with unclear status.
* To provide isolation between programs accessing a database concurrently. If this isolation is not provided, the program's outcome are possibly erroneous.

Example: When the owner of a coffee-shop updates the amount of coffee sold each week, he will also want to update the total amount sold to date. However, the amount sold per week and the total amount sold should be updated at the same time; otherwise, the data will be inconsistent. The way to be sure that either both actions occur or neither action occurs is to use a transaction.

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

A NoSQL database environment is, simply put, a non-relational and largely distributed database system that enables rapid, ad-hoc organization and analysis of extremely high-volume, disparate data types. NoSQL databases are sometimes referred to as cloud databases, non-relational databases, Big Data databases and a myriad of other terms and were developed in response to the sheer volume of data being generated, stored and analyzed by modern users (user-generated data) and their applications (machine-generated data).

**NoSQL** provides a mechanism for [storage](http://en.wikipedia.org/wiki/Computer_data_storage) and [retrieval](http://en.wikipedia.org/wiki/Data_retrieval) of data that is modeled in means other than the tabular relations used in [relational databases](http://en.wikipedia.org/wiki/Relational_database). Motivations for this approach include simplicity of design, [horizontal scaling](http://en.wikipedia.org/wiki/Horizontal_scaling#Horizontal_and_vertical_scaling) and finer control over availability. The data structure (e.g. key-value, graph, or document) differs from the [RDBMS](http://en.wikipedia.org/wiki/RDBMS), and therefore some operations are faster in NoSQL and some in RDBMS.

In general, NoSQL databases have become the first alternative to relational databases, with scalability, availability, and fault tolerance being key deciding factors. They go well beyond the more widely understood legacy, relational databases (such as Oracle, SQL Server and DB2 databases) in satisfying the needs of today’s modern business applications. A very flexible and [schema-less data model](http://www.planetcassandra.org/blog/post/schema-vs-schema-less), horizontal scalability, distributed architectures, and the use of languages and interfaces that are “not only” SQL typically characterize this technology.

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

* With the **column-family** model, you have rows and columns like you would expect, but the rows are [sparse](http://en.wikipedia.org/wiki/Sparse_array), meaning each row can have as many or as few columns as desired, and columns do not need to be defined ahead of time.
* The **Key/value** model is simple and easy to implement, but it is inefficient when you are only interested in querying or updating part of a value. It’s also [difficult to implement more-sophisticated structures on top of distributed key/value](http://spyced.blogspot.com/2009/05/why-you-wont-be-building-your-killer.html).
* With **Document** databases, you get essentially the next level of Key/value, allowing nested values associated with each key. Document databases support querying those more efficiently than simply returning the entire [blob](http://en.wikipedia.org/wiki/Binary_large_object" \t "_blank)each time.
* Neo4J uses the unique **Graph** data model, storing objects and relationships as nodes and edges in a graph. For queries that fit this model (e.g., hierarchical data), they can be [thousands of times faster](http://www.slideshare.net/emileifrem/neo4j-the-benefits-of-graph-databases-oscon-2009) than alternatives.
* Unlike the others, Scalaris offers distributed transactions across multiple keys. There are, however, [trade-offs between consistency and availability](http://queue.acm.org/detail.cfm?id=1394128) that you should keep in mind.

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

### MongoDB

[Mongodb](http://www.thegeekstuff.com/2013/01/install-mongodb/) is one of the most popular document based NoSQL database as it stores data in JSON like documents. It is non-relational database with dynamic schema. It has been developed by the founders of DoubleClick, written in C++ and is currently being used by some big companies like The New York Times, Craigslist, MTV Networks. The following are some of MongoDB benefits and strengths:

Speed: For simple queries, it gives good performance, as all the related data are in single document which eliminates the join operations.

Scalability: It is horizontally scalable i.e. you can reduce the workload by increasing the number of servers in your resource pool instead of relying on a stand-alone resource.

Manageable: It is easy to use for both developers and administrators. This also gives the ability to shard database

Dynamic Schema: Its gives you the flexibility to evolve your data schema without modifying the existing data

### CouchDB

[CouchDB](http://www.thegeekstuff.com/2012/06/install-couch-db/) is also a document based NoSQL database. It stores data in form of JSON documents. The following are some of CouchDB benefits and strengths:

Schema-less: As a member of NoSQL family, it also have dynamic schema which makes it more flexible, having a form of JSON documents for storing data.

HTTP query: You can access your database documents using your web browser.

Conflict Resolution: It has automatic conflict detection which is useful while in a distributed database.

Easy Replication: Implementing replication is fairly straight forward

### Redis

[Redis](http://redis.io/) is another Open Source NoSQL database which is mainly used because of its lightening speed. It is written in ANSI C language. The following are some of Redis benefits and strengths:

Data structures: Redis provides efficient data structures to an extend that it is sometimes called as data structure server. The keys stored in database can be hashes, lists, strings, sorted or unsorted sets.

Redis as Cache: You can use Redis as a cache by implementing keys with limited time to live to improve the performance.

Very fast: It is consider as one of the fastest NoSQL server as it works with the in-memory dataset.