Databases
Database Server
Relational Database Management Systems

A way of saving and accessing data on
Why Use an RDBMS?
Why Use an RDBMS

• Concurrent Access
  – atomic updates via transactions

• Fault Tolerance
  – replicated dbs for instant failover on machine/disk crashes

• Data Integrity
  – aids to keep data meaningful
Why Use an RDBMS

• Scalability
Relational Model

- First published by Edgar F. Codd in 1970 - Received Turing Award in 1981
- A relational database consists of a collection of tables
- Each row represents a record
- Each column represents an attribute of the records contained in the table
Example
RDBMS Technology

• Client/Server Databases
  – Oracle, Sybase, MySQL, SQLServer, PostgreSQL

• Personal Databases
  – Access

• Embedded Databases
  – Pointbase
“NoSQL” = “Not Only SQL”

Not every data management/analysis problem is best solved using a traditional DBMS

Database Management System (DBMS) provides...

...efficient, reliable, convenient, and safe multi-user storage of and access to massive amounts of persistent data.
"NoSQL" = "Not Only SQL"

NoSQL Systems

Alternative to traditional relational DBMS

+ Flexible schema
+ Quicker/cheaper to set up
+ Massive scalability
+ Relaxed consistency → higher performance & availability

– No declarative query language → more programming
– Relaxed consistency → fewer guarantees
**NoSQL Database Types**

**Document** databases pair each key with a complex data structure known as a document. Documents can contain many different key-value pairs, or key-array pairs, or even nested documents.

**Graph stores** are used to store information about networks, such as social connections. Graph stores include Neo4J and HyperGraphDB.
Key-value stores are the simplest NoSQL databases. Every single item in the database is stored as an attribute name (or "key"), together with its value. Examples of key-value stores are Riak and Voldemort. Some key-value stores, such as Redis, allow each value to have a type, such as "integer", which adds functionality.

Wide-column stores such as Cassandra and HBase are optimized for queries over large datasets, and store columns of data together, instead of rows.
Flexible schema

Relational databases require that schemas be defined before you can add data. This fits poorly with agile development approaches, because each time you complete new features, the schema of your database often needs to change.

NoSQL databases are built to allow the insertion of data without a predefined schema.
Auto-sharding

Because of the way they are structured, relational databases usually scale vertically—a single server has to host the entire database to ensure reliability and continuous availability of data.

NoSQL databases, on the other hand, usually support auto-sharding, meaning that they natively and automatically spread data across an arbitrary number of servers.
NoSQL databases also support automatic replication, meaning that you get high availability and disaster recovery without involving separate applications to manage these tasks.
Integrated Caching

Many NoSQL database technologies have excellent integrated caching capabilities, keeping frequently-used data in system memory as much as possible and removing the need for a separate caching layer that must be maintained.