SQL versus NoSQL Terminology

Why NoSQL

- Scaling
- Distributed Data Sources
- High cost of joins
- Great variation in data
- Design focuses on the query needs of the application
- RDBMS do not always match the needs of the application
- RDBMS not going away

Some NoSQL Characteristics

- No predefined schema
- Limited or no support for declarative query language
- Focus on scalability, availability and performance

SQL versus NoSQL

- Transactions
- ACID Properties
  - Atomicity
  - Consistency
  - Isolation
  - Durability

SQL versus NoSQL

- BASE
  - **B**asically **A**vailable
  - **S**oft state
  - **E**ventually consistent

SQL versus NoSQL

- CAP
  - Consistent
    - All replicas contain the same view of the data
    - Clients always see the same view of the data
  - Available
    - System remains operational in the presence of failures
    - All clients can always read and write
  - Partition Tolerance
    - System remains operational in presence of communication failures or network partition
- Cap “Theorem”
  - Systems can only support 2 of 3
  - The idea is widely debated

SQL versus NoSQL

- Scalability
  - Horizontal
    - Distribute data and load over many servers
    - The servers do not share RAM or Disks
  - Vertical
    - Distribute load over many cores or processors
    - The cores or processors share RAM and Disks

SQL versus NoSQL

- Partitioning
  - Horizontal (Sharding)
    - Storing records on different servers
  - Vertical
    - Storing parts of a record on different servers
- Replication
  - Storing multiple copies of the same data

SQL versus NoSQL

- Taxonomy of NoSQL
  - Key-value
  - Column Based
  - Document
  - Graph Database