## Taking Decisions on Databases

- If compliance of ACID (Atomicity, Durability, Consistency, and Durability) properties is essential, RDBMS is the best solution. For example, databases for financial transactions.
- If your input data is particularly heterogeneous and difficult to encapsulate according to a normalization schema, consider using a NoSQL DBMS. For example, data from Social Media.
- If your goal is to scale database vertically, a RDBMS is suggested and if you want to scale horizontally, a NoSQL DBMS may be preferred.
- If you have a massively distributed system and can settle for eventual consistency on some nodes/partitions, you might consider a wide column store.
- **Normalized(SQL) Vs De-normalized Data(NoSQL)**

  Normalization is a database business process to break up data into the smallest possible parts. Instead of storing first and last name in one bucket, or field, normalization requires that you store the first name separately from the last name. This is helpful if you want to sort the data by last name or by first name. Relational databases are the most common database systems and require that data is **normalized**. These data is stored in columns and rows, which in turn make up tables like a spreadsheet. A set of tables makes up a schema. A number of schemas create a database. Many databases can be created on a single server. RDBMS system feature much better performance for managing data over desktop database programs. RDBMS allow multiple users (even thousands!) to work with the data at the same time, creating advanced security for access to the data.

  Databases like NoSQL and object-oriented do not follow the table/row/column approach of RDBMS. Instead, they build bookshelves of elements and allow access per bookshelf. So, instead of tracking individual words in books, these databases narrow down the data being searched for by pointing to the bookshelf, then a mechanical assistant works with the books to identify the exact word you are looking

for. NoSQL specifically attempts to simplify bookshelves by storing data in a **denormalized** way; this means storing it in large chunks.

## Strengths and weaknesses of databases

There are many databases to choose from but selecting the right kind of database is very essential. All databases are not equal and each of them specific strengths and weaknesses. Lets have a view of some of the databases:

### 1. Relational database management systems

It is over 4 decades when RDBMS were first come into the real world to handle the increasing flood of data being produced. And since then, RDBMS have a solid foundational theory and have influenced nearly every database system in use today. RDBMS store normalized data in form of data sets called **relations** i.e. tables with rows and columns where all information is stored as a value of a specific cell. SQL is used to manage data in an RDBMS system. SQL is standardized and provides a level of predictability and utility.

**Strengths**

- Relational databases have expertise in handling highly structured data and provide support for ACID (Atomicity, Consistency, Isolation, and Durability) transactions.
- The structure can be scaled up quickly because adding data without modifying existing data is simple.
- Data can be easily stored, modified and retrieved using SQL queries.
- Access rights can be assigned to certain users, like to only view the data, modify the data / data structure or full access etc. Due to these, relational databases are well-suited to applications that require tiered access.

**Weaknesses**

- RDBMS are expensive to set up and grow. Horizontal scaling, or scaling by adding more servers, is usually both faster and more economical than vertical scaling, which

involves adding more resources to a server is supported by RDBMS. Further, the structure of relational databases complicates the process.

- RDBMS are good as they are handling structured data, but difficult to handle unstructured data. Representing real world entities in context is difficult in the bounds of an RDBMS. Now a day, a lot of unstructured data is playing in the real world.

- For better readability, the data has to be reorganized from tables into a more readable form, and this also impact the speed. The fixed schema of these databases doesn't support such changes in a positive way.

- Sharding is necessary to scale out a relational database. In shredding, data is horizontally partitioned and distributed across a collection of machines. But maintaining ACID compliance while Sharding RDBMS is again a big issue.

**Final verdict**

Based on the various parameters, strengths and weaknesses, a relational database may be used for

- ✓ Situations where data integrity is absolutely necessary for example financial applications, banking application, military defense and security applications, private health information systems etc.

- ✓ Applications having Highly structured data

- ✓ Applications having automation of internal processes

**Some example of RDBMS are** Oracle, MySQL, MS Server, Sybase, Informix, PostgreSQL.

*To be continued…*

**Assignments**

1. What database is suggested for Normalized and De-normalized Data? Justify your response.

2. What are the strengths and weaknesses of RDBMS and how it is important in selecting the right database? Explain.