Saved Upgrade Preview Publish

Add Cover Add Subtitle

How to choose a database

To consider a database for a product we have to consider a database on technical and non-technical criteria.

Non-technical criteria

- **Querying interface:**
  - Every database provides different ways to query the data, for example, SQL in Mysql or Pgsql, more js object-like syntax in MongoDB, KQL in elastic search etc.
- **Bulk processing support:** This refers to how easy it is to import / export data or do processing on multiple records together.

Technical

Transaction Support: -> And RDBMS gives good transaction support

- **ACID support:**
  - A -> atomicity -> either the transaction will complete all the steps or, if something goes sideways then none of them will be applied. (MySQl achieves this using **undo logs**)
  - C -> consistency -> before and after transaction data is consistent (double write buffer)
  - I -> isolation -> Transactions should execute independently. using locks and isolation level
  - D -> durability -> (REDO logs)
- NoSQL->
  - A -> associative -> This allows parallel processing by enabling operations to be applied independently.
  - C -> commutative -> If multiple operations are going to be applied, in a different order, the result should be the

same.

- I -> idempotent -> If an operation is applied multiple times then its effect is only applied once.
- D -> distributed -> NoSQL databases support distributed architecture.

Scaling:

- RDBMS -> vertical scaling
- Nosql -> horizontal scaling

Normalisation:

- RDBMS -> They handle normalisation better
- NoSQL -> They don't have joins etc, so they don't handle normalisation well

An example case of MongoDB

JSON

```
{
        chat: {
            chat_room
            message
            sender_id
        }

    }
```

Chat

- Flexible schema
- Scalable
- Chat backup using import-export
- No acid required

An example of RDBMS

- Payment system
  - High acid requirement, transaction capabilities
  - Strict schema

- Normalisation, Join
-