LET'S START WITH DBMS :)

Denormalization

This is the opposite of normalization. It involves intentionally introducing some redundancy into a well-normalized database schema to improve query performance.

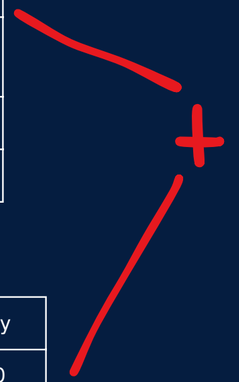
Consider if you wish to find the salary of Rahul
so first you have to make a query in employee table to find department of Rahul and then in department table to find the salary of Rahul

Employee

id	name	age	department
1	Rahul	25	IT
2	Afsara	26	HR
3	Abhimanyu	27	IT
4	Aditya	25	HR
5	Raj	24	HR

Department

department	Manager	salary
IT	Raj	1500
HR	Avinash	1000



Employee

id	name	age	department	Manager	salary
1	Rahul	25	'IT'	Raj	1500
2	Afsara	26	'HR'	Avinash	1000
3	Abhimanyu	27	'IT'	Raj	1500
4	Aditya	25	'HR'	Avinash	1000
5	Raj	24	'HR'	Avinash	1000

LET'S START WITH DBMS :)

Denormalization

Benifits

- **Faster Queries** : It can reduce the need for complex joins between tables during queries which can eventually improve the speed of retrieving frequently accessed data.
- **Simpler Queries**: It can simplify queries by allowing them to be executed on a single table instead of requiring joins across multiple tables.

Disadvantages

- Increased Data Redundancy
- Less Data Consistency
- Denormalization can make the database schema less flexible for future changes. like adding/modifying new data elements