



# System Design

## Database Basics

---

---

---



# Relational Database (RDBMS) কী?

Relational Database এটাকে আমরা এমন একটি ডাটাবেজ বলবো, যেখানে আমরা ডাটা টেবিল আকারে (rows & columns) স্টোর করে থাকি। এবং টেবিল গুলোর মধ্যে রিলেশন করে থাকি আমাদের চাহিদা অনুযায়ী। এবং SQL দিয়ে যেখানে ডাটা ম্যানিপুলেট করা হয়।

যদি উদাহরণ দেখি :MySQL, PostgreSQL, Oracle, SQL Server

# Non-relational Database (NoSQL) কী?

Non-relational database বলতে সাধারণত আমরা বুঝি যা এমন একটি ডাটাবেজ যেখানে টেবিলের পরিবর্তে ডাটা স্টোর করে Document, Key-Value, Graph বা Column-based ফরম্যাটে। যেখানে আপনার স্কিমা ফিক্সড না – মানে এইটা স্কিমা-লেস (Schema-less)

উদাহরণ হলো :MongoDB, Cassandra, Redis, Firebase, DynamoDB

## RELATIONAL DATABASE আমরা কেন ব্যবহার করবো?

1. এখানে ডাটা স্ট্রাকচার আগে থেকেই জানা থাকে, মানে (Fixed schema)
2. টেবিলের মধ্যে রিলেশন থাকে যেখানে কি এর মাধ্যমে জয়েন হয় (Foreign key, প্রাইমারি Key)
3. ACID প্রিন্সিপল দরকার পরে (যেমন: ব্যাংকিং অ্যাপ)
4. ছোট থেকে মাঝারি স্কেল বা ট্রানজেকশন heavy অ্যাপ্লিকেশন এর ক্ষেত্রেও দরকার পরে

## NON-RELATIONAL DATABASE কেন ব্যবহার করবো ?

1. যখন আমাদের ফ্লেক্সিবল ডাটা স্ট্রাকচার দরকার, এবং এটি (Schema-less)
2. যদি অনেক দ্রুত স্কেল করতে হয় (Big Data, Real-time) ধরেন আপনার অ্যাপ্লিকেশন বা সিস্টেমে যদি হঠাৎ ১০০ জন ইউজারের বদলে ১ লাখ ইউজার চলে আসে, তাহলে সেটিকে খুব দ্রুত আরও সার্ভার যুক্ত করে স্কেল করতে হবে, যাতে সবার জন্য পারফরম্যান্স ঠিক থাকে
3. রিলেশন দরকার নেই বা একদম কম
4. Distributed system, High-performance অ্যাপ্লিকেশন যদি হয়।

## ACID PRINCIPLES – কী?

ACID হল সাধারণত একটি ধারণা বা বৈশিষ্ট বা নীতিতে বলতে পারেন যা মূলত Relational Database (SQL)-এ ডেটা সঠিকভাবে সংরক্ষণ এবং ট্রানজেকশন নিরাপদভাবে পরিচালনা করতে সাহায্য করে।

ACID প্রতিটির আলাদা আলাদা নাম এবং বৈশিষ্ট আছে চলুন দেখি।

### A – Atomicity

"সব হবে, না হলে কিছুই হবে না"

যেকোনো ট্রানজেকশন পুরোপুরি সম্পন্ন হবে, নয়তো একেবারেই হবে না।

উদাহরণ: ধরুন একাউন্ট থেকে টাকা পাঠালেন মাঝখানে সার্ভার সমস্যা হলো বা অন্য যেকোনো কিছু হলো টাকা যাবে না এবং আপনার টাকা পুনরায় আপনার একাউন্ট এ চলে আসবে।

### C – Consistency

ডাটাবেসের ডেটা সবসময় সঠিক ও নির্ভরযোগ্য থাকবে।

উদাহরণ: অর্ডার প্লেস করার পর যদি স্টক কমে যায়, তাহলে ডাটাবেস আপডেট হয়ে স্টক সবসময় সঠিক দেখাবে

## I – Isolation

একাধিক ট্রানজেকশন একে অপরকে প্রভাবিত করতে পারবে না মানে হলো

একই সময় একাধিক ট্রানজেকশন চললেও একে অপরের ডেটা পরিবর্তন করতে পারবে না।

উদাহরণ: আপনি এবং অন্য কেউ একই প্রোডাক্ট কিনছেন – আইসোলেশন নিশ্চিত করে যেন কারো অর্ডার অন্যজনকে প্রভাবিত না করে।

## D – Durability

যদি একবার ডাটা সেভ হয় সেটা পার্মানেন্ট ভাবে হয়ে যাবে। ট্রানজেকশন সফল হলে সার্ভার অফ হলেও ডেটা থাকবে।

উদাহরণ: একবার অর্ডার কনফার্ম হলে, সার্ভার রিস্টার্ট হলেও অর্ডার মুছে যাবে না।

## কেন ব্যবহার করবেন?

যেখানে ট্রানজেকশনের নিরাপত্তা ও নির্ভুলতা খুবই গুরুত্বপূর্ণ। ব্যাংকিং, পেমেন্ট গেটিওয়ে, POS, ERP ইত্যাদি

## কিভাবে কাজ করে?

SQL ডেটাবেজে ট্রানজেকশন শুরু হয়, একাধিক কাজ হয়, সব সফল হলে COMMIT, না হলে ROLLBACK হয়

## BASE PRINCIPLES – কী?

BASE প্রিন্সিপাল মূলত NoSQL Database এর বৈশিষ্ট্য বা নীতিমালা বোঝাতে ব্যবহৃত হয়। এটি সাধারণত বড়, ডিসট্রিবিউটেড সিস্টেম-এর জন্য ডিজাইন করা হয়, যেখানে আমাদের ডেটা দ্রুত ও স্কেলযোগ্যভাবে পাওয়া দরকার।

### B – Basically Available

সার্ভার সবসময় কিছু না কিছু রেসপন্স করবেই আপনাকে। ফুল গ্যারান্টি না থাকলেও আপনার অ্যাপ্লিকেশন চলতে থাকবে।  
উদাহরণ: যেমন ধরুন আপনার নিউস ফিড লোড হচ্ছে, কিছু পোস্ট দেরিতে আসছে – কিন্তু অ্যাপ বন্ধ হচ্ছে না।

### A – Soft State

ধরুন কোনো ডেটা পাঠানো হয়েছে সেটা সাথে সাথেই চলে আসলো বা একটু পরে আসলো মানে সময়ের সাথে ডাটা পরিবর্তন। এখানে কিছু সময়ের জন্য ডাটা আপডেট না হলেও সমস্যা নেই।

উদাহরণ: আমাদের চ্যাট এপ্লিকেশন মেসেজ সাথে সাথে না এলেও কিছুক্ষণ পরে আপডেট হয়ে যায়।

## E – Eventually Consistent

এটার মানে হলো ধরুন শুরুতে সব সার্ভারে এক রকম ডেটা নাও থাকতে পারে মানে আপডেট না ও হতে পারে, কিন্তু সময়ের সাথে মিলে ডেটা ধীরে ধীরে সব জায়গায় আপ টু ডেট হয়ে যাবে।

উদাহরণ: যখন মাঝে মাঝে Amazon-এ অর্ডার দিবেন প্রথমে লিস্ট থেকে সরে না গেলেও দেখবেন পরে সরে যায়।

BASE প্রিন্সিপাল কেন দরকার ?

- যেখানে স্পিড, স্কেল আর পারফরম্যান্স আমাদের বেশি দরকার
- সবসময় ১০০% পারফেক্ট না হলেও অ্যাপ চলতে হবে বন্ধ হওয়া যাবেনা

এটি কিভাবে কাজ করে?

- NoSQL সিস্টেমগুলো সার্ভার বিভক্ত করে রাখে, মানে সারডিং করে রাখে।
- অনেক ইউজারকে সার্ভিস দিতে গিয়েই trade-off নেয় consistency এর তার মানে হলো সবার আগে ইউজার সার্ভিস পাবে – ডেটা পরে সবার কাছে আপডেট হবে। সুতরাং CAP থিওরেম এর consistency এর কিছুটা ছাড় দেয়।

# কখন কোনটা ব্যবহার করবেন?

পরিস্থিতি	ব্যবহার করবেন
ব্যাংকিং, ট্রানজেকশন, ইনভয়েস	✓ ACID (SQL)
Social Media, Chat App	✓ BASE (NoSQL)
ERP, Inventory	✓ ACID
IoT Sensor, Realtime Tracking	✓ BASE
যেখানে consistency critical	ACID
যেখানে availability & speed critical	BASE

## ACID vs BASE – তুলনা ছকে:

বৈশিষ্ট্য	ACID	BASE
Focus	Correctness	Availability & Performance
Consistency	Strong	Eventually
Scalability	Low (Vertical)	High (Horizontal)
Complexity	High Accuracy	High Flexibility
ব্যবহার হয়	SQL DB	NoSQL DB



## OLTP (Online Transaction Processing) কী

OLTP হলো এমন একটি ডেটাবেস সিস্টেম যেটা রিয়েল-টাইমে ছোট ছোট ট্রান্সাকশন হ্যান্ডেল করে। যেমন: ইউজার যখন ওয়েবসাইটে অর্ডার করে, টাকা দেয়, লগিন করে, তখন এই ডেটাগুলো সাথে সাথে সেভ হয় – এটিই OLTP।

OLTP (ONLINE TRANSACTION PROCESSING) এর বৈশিষ্ট্য

- প্রতিটি ট্রান্সাকশনের রেসপন্স টাইম মিলিসেকেন্ড-লেভেলে হয়।
- প্রতিদিন লক্ষ লক্ষ ইউজার ছোট ছোট কাজ করে (যেমন: লগইন, অর্ডার করা, টাকা ট্রান্সফার)।
- প্রায় প্রতিটি ইউজারই কোনো না কোনোভাবে ডেটা তৈরি, দেখা, পরিবর্তন বা ডিলিট করে।
- ডেটা নরমালাইজেশন কমাতে বিভিন্ন টেবিলে ভাগ করা হয় (NORMALIZATION)।

ব্যবহার ক্ষেত্র (USE CASES):

- ই-কমার্স অ্যাপ (অর্ডার করা, পেমেন্ট, ডেলিভারি ট্র্যাকিং)
- ব্যাংকিং সিস্টেম (অ্যাকাউন্ট ব্যালেন্স, লেনদেন)
- টিকেট বুকিং সিস্টেম (সিট বুকিং, কনফার্মেশন)

# OLAP (Online Analytical Processing) – বড় ডেটা বিশ্লেষণ ও রিপোর্টিং

OLAP মূলত বিশ্লেষণের জন্য ব্যবহৃত হয়। এখানে লক্ষ্য থাকে অনেক ট্রান্সাকশনের উপর ভিত্তি করে রিপোর্ট তৈরি করা। এটি REALTIME না, বরং ব্যাকএন্ডে DATA WAREHOUSING এর মাধ্যমে বিশ্লেষণ চালানো হয়।

## বৈশিষ্ট্যসমূহ

-> OLAP সিস্টেমে ডেটা সাধারণত রিড (পড়ার) জন্য বেশি ব্যবহার হয়, যেমন: রিপোর্ট তৈরি, গ্রাফ ড্র, বিশ্লেষণ ইত্যাদি। আপডেট বা ডিলিট খুব কম হয়।

-> এখানে অনেক বড় বড় এবং জটিল SQL QUERY চালানো হয় – যেমন:

- MULTIPLE টেবিলের JOIN
- GROUP BY দিয়ে গ্রুপ করা
- SUM, AVG, MAX, MIN দিয়ে বিশ্লেষণ

-> এখানে ডেটা একত্র করে সারাংশ (SUMMARY) হিসেবে দেখানো হয়। যেমন:

- "গত মাসে কোন প্রোডাক্ট কতবার বিক্রি হয়েছে?"
- "কোন অঞ্চলে সবচেয়ে বেশি লাভ হয়েছে?"

-> পারফরমেন্সের জন্য OLAP সিস্টেমে ডেটা সাধারণত DENORMALIZED করা হয় (একই ইনফো একাধিক জায়গায় রাখা হয় যাতে দ্রুত অ্যাক্সেস করা যায়)।

# শেষ কথা:

পরবর্তীতে আমরা আরো বেশ কিছু অ্যাডভান্সড টপিকস নিয়েও আলোচনা করবো।

1. Normalization (1NF, 2NF, 3NF – কখন দরকার, কখন avoid করা উচিত)
2. Denormalization – কেন করব, কখন করব
3. ER Diagram, Entity Relationship Modeling
4. Primary Key, Foreign Key, Indexing Concepts