ডাটাবেজ নরমালাইজেশন

ডাটাবেজ নরমালাইজেশন (normalization) কী জিনিস? এক কথায় আসলে উত্তর দেওয়া সম্ভব নয়। তাই বরং আসুন, আমরা বিষয়টি নিয়ে একটু বিস্তারিত আলচনা করি। কোনো কিছুকে নরমালাইজ (normalize) করার অর্থ হচ্ছে সেটিকে স্বাভাবিক (normal) অবস্থায় নিয়ে আসা। তো ডাটাবেজের ক্ষেত্রে এই নরমালাইজেশনের অর্থ হচ্ছে ডাটাবেজকে এমন অবস্থায় নিয়ে আসা যেন ডাটা রিডানডেন্সি (data redundancy) না থাকে এবং ডাটা ইন্টিগ্রিটি (data integrity) বজায় থাকে। এই যে এখন আবার নতুন দুটো জিনিস চলে এল, ডাটা রিডানডেন্সি ও ডাটা ইন্টিগ্রিটি। এগুলো আবার কী জিনিস?

ডাটা রিডানডেন্সি – ইংরেজিতে redundant শব্দের অর্থ হচ্ছে প্রয়োজনের অতিরিক্ত। আমাদের ডাটাবেজ ডিজাইনের সময় একটি ব্যাপারে লক্ষ রাখতে হবে যেন আমরা প্রয়োজনের অতিরিক্ত ডাটা না রাখি। অনেক সময়ই একই ডাটা বারবার বিভিন্ন টেবিলে এমনভাবে আসে, যেখানে টেবিলগুলো একটু অন্যভাবে ডিজাইন করলেই অনেক ডাটা বেঁচে যেত। ডাটাবেজ নরমালাইজ করার মাধ্যমে আমরা নিশ্চিত করি যে ডাটাবেজে রিডানডেন্ট ডাটা থাকছে না।

ডাটা ইনটিগ্রিটি – Integrity শব্দের অর্থ হচ্ছে শুদ্ধতা। অনেক সময় ডাটাবেজে বিভিন্ন কারণে (হার্ডওয়্যারের ত্রুটি কিংবা সফটওয়্যারের সমস্যা বা ডাটাবেজ ডিজাইনের সমস্যা) ডাটায় ভেজাল ঢুকে যায়। এই ভেজাল আবার কী জিনিস? ধরা যাক কোনো একভাবে হিসেব করলে একজন শিক্ষার্থীর মোট নাম্বার হয় ৫৪৬, আবার আরেকভাবে (যেমন অন্য কোনো টেবিল থেকে ডাটা নিয়ে) হিসেব করলে মোট নাম্বার হয় ৫৫৫। তার মানে ডাটাতে ভেজাল ঢুকে গিয়েছে বা ডাটা তার শুদ্ধতা হারিয়ে ফেলেছে। নরমালাইজেশন করলে ডাটার শুদ্ধতা বজায় থাকার সম্ভাবনা বেড়ে যায় অনেক।

এখন, নিচের উদাহরণগুলো দিয়ে নরমালাইজেশন বিষয়টি বোঝার চেষ্টা করি –

উপরের টেবিলটিতে কোনো নরমালাইজেশন নেই। এই টেবিলে নতুন ডাটা যোগ করতে (data insert), পুরনো ডাটা পরিবর্তন (data update) করতে এবং ডাটা মুছে ফেলতে (data delete) আমাদের কিছু অসুবিধা হবে

Table: Student

studentId	name	age	subject	postCode	city
101	Mark	20	CSE-101, CSE-102, CSE-103	1200	Dhaka
102	Zakir	19	CSE-101, CSE-102, CSE-103	1210	Dhaka
103	Johny	21	CSE-101, CSE-102, CSE-103	5400	Rangpur
104	Fahim	20	CSE-101, CSE-102	5400	Rangpur

(সেগুলোকে নরমালাইজেশন দ্বারা দূর করা যায়)। যেমন, আমরা যদি নতুন একজন স্টুডেন্ট এই টেবিলে যোগ করতে চাই যে কোনো সাবজেক্টই নেয় নি, তাহলে subject কলামে NULL ভ্যালু যাবে। আবার আমারা যদি একজন স্টুডেন্ট এর সাবজেক্ট বাড়াতে বা কমাতে চাই তাহলে আমরা খুব সহজে তা করতে পারব না, কারণ subject কলামে ডাটা কমা দিয়ে আলাদা করা আছে।

এখন আমরা Student টেবিলটিকে First normal form (1NF) নিতে চাই। First normal form (1NF) এর শর্ত হচ্ছে টেবিলের সব কলামের ভ্যালু একক (atomic) হতে হবে। আমারা দেখতে পাচ্ছি যে subject কলামের ডাটা একক (atomic) নয়। নিচে দেখানো উপায়ে আমরা Student টেবিলটিকে পরিবর্তন করে First normal form (1NF) এ নিলাম –

Table: Student

studentId	name	age	subject	postCode	city
101	Mark	20	CSE-101	1200	Dhaka
101	Mark	20	CSE-102	1200	Dhaka
101	Mark	20	CSE-103	1200	Dhaka
102	Zakir	19	CSE-101	1210	Dhaka
102	Zakir	19	CSE-102	1210	Dhaka
102	Zakir	19	CSE-103	1210	Dhaka
103	Johny	21	CSE-101	5400	Rangpur
103	Johny	21	CSE-102	5400	Rangpur
103	Johny	21	CSE-103	5400	Rangpur
104	Fahim	20	CSE-101	5400	Rangpur
104	Fahim	20	CSE-102	5400	Rangpur

এখন Student টেবিলের সব কলামের ভ্যালু একক (atomic) হয়েছে। এবার আমারা দেখতে পাচ্ছি যে, একই ডাটা বার বার আসছে। শুধুমাত্র subject কলামের ডাটা পরিবর্তন হচ্ছে।

আমারা এবার Second normal form (2NF) এ আমদের Student টেবিলটিকে নিয়ে যাব। এর জন্য নিচের শর্ত দুটি পূরণ করতে হবে –
১) টেবিলটি First normal form (1NF) এ থাকতে হবে
২) কোনও non-prime অথবা non-key attribute, candidate key এর subset এর উপর নির্ভরশীল হতে পারবে না।

[candidate key মানে এমন কলাম বা কলামের সমষ্টি যা একটি টেবিলের প্রতিটি রেকর্ড কে আলাদা ভাবে চিহ্নিত করতে পারে। একটি টেবিলের এক বা একাধিক candidate key থাকতে পারে। এর মধ্যে একটি বিশেষ

candidate key কে আমরা primary key বলি। যে attribute/column কোনও candidate key এর অংশ নয় তাকে non-prime attribute অথবা non-key attribute বলে।]।

আমদের Student টেবিলটি First normal form (1NF) এ আছে। আমাদের দ্বিতীয় শর্তটি পূরণ করতে হবে। Student টেবিল থেকে আমরা লক্ষ করি যে, {studentId, subject} কলাম দুটি মিলে হচ্ছে একটা candidate key এবং name, age, postCode, city কলামগুলো হচ্ছে non-prime attribute।

এখন name, age, postCode, city কলামগুলি শুধুমাত্র studentId কলামের উপর নির্ভরশীল এবং studentId হল candidate key: {studentId, subject} এর একটি subset।

আমরা Student টেবিলটিকে নিচের মত করে Second normal form (2NF) এ নিতে পারি। আমারা একটি নতুন টেবিল Student_Subject তৈরি করলাম স্টুডেন্ট এবং সাবজেক্ট এর মধ্যে সম্পর্ক ঠিক রাখার জন্য।

Table: Student

studentId	name	age	postCode	city
101	Mark	20	1200	Dhaka
102	Zakir	19	1210	Dhaka
103	Johny	21	5400	Rangpur
104	Fahim	20	5400	Rangpur

Table: Student_Subject

studentId	subject
101	CSE-101
101	CSE-102
101	CSE-103
102	CSE-101
102	CSE-102
102	CSE-103
103	CSE-101
103	CSE-102
103	CSE-103
104	CSE-101
104	CSE-102

তাহলে আমাদের ডাটাবেজ এখন Second normal form (2NF)-এ চলে আসল। এবারে আমরা শেষ ধাপে যাব এবং একে Third normal form (3NF)-এ নেব। যার মাধ্যমে আমাদের নরমালাইজেশন করার প্রক্রিয়াটি সম্পন্ন হবে।

Third normal form (3NF) এ নেওয়ার জন্য আমদের নিচের দুটি শর্ত পূরণ করতে হবে –
১) টেবিল Second normal form (2NF) এ থাকতে হবে
২) কোনো Transitive functional dependency থাকতে পারবে না

[Transitive functional dependency – মনে করি একটি টেবিলের প্রাইমারি কি (primary key) হচ্ছে A এবং এই টেবিলের দুটি নন-প্রাইম (non-prime) কলাম হচ্ছে B এবং C, যেখানে C কলামের ভ্যালু যতটা A কলামের ভ্যালুর উপরে নির্ভরশীল তার চাইতে B কলামের ভ্যালুর উপর বেশি নির্ভরশীল, আবার B কলামের ভ্যালু A কলামের ভ্যালুর উপরে সরাসরি নির্ভরশীল, তাহলে আমরা বলতে পারি যে C কলাম transitively কলাম A এর উপর নির্ভরশীল। ওই যে, ছাগল ঘাস খায়, মানুষ ছাগল খায়, তার মানে মানুষ ঘাস খায় – এরকম লজিক আর কী।]

আমাদের Student টেবিলটিতে studentId হচ্ছে প্রাইমারি কি (primary key) এবং postCode আর city হচ্ছে দুটি নন-প্রাইম (non-prime) কলাম। আমরা লক্ষ করি যে, city কলামটি যতটা studentId কলামের উপরে নির্ভরশীল তার চাইতে বেশি নির্ভরশীল postCode কলামটির উপরে এবং postCode কলামটি আবার studentId কলামের উপরে সরাসরি নির্ভরশীল। সুতরাং আমরা বলতে পারি যে city কলামটি transitively কলাম studentId এর উপর নির্ভরশীল।

তাই Student টেবিলটিকে Third normal form (3NF)-এ নিতে নিচের মতো পরিবর্তন করতে পারি এবং PostCode_City নামে একটি নতুন টেবিল তৈরি করতে পারি –

Table: Student

studentId	name	age	postCode
101	Mark	20	1200
102	Zakir	19	1210
103	Johny	21	5400
104	Fahim	20	5400

Table: PostCode_City

postCode	city
1200	Dhaka
1210	Dhaka
5400	Rangpur

Table: Student_Subject

studentId	subject
101	CSE-101
101	CSE-102
101	CSE-103
102	CSE-101
102	CSE-102
102	CSE-103
103	CSE-101
103	CSE-102
103	CSE-103
104	CSE-101
104	CSE-102

তাহলে নরমালাইজ করতে গিয়ে আমরা একটি টেবিল ভেঙ্গে তিনটি টেবিল তৈরি করলাম। আর সেই সাথে প্রয়জনাতিরিক্ত ডাটা যেমন কমে গেল, ডাটাতে গন্ডগল হওয়ার সম্ভাবনাও কমল। তো শুরুতে নিয়মকানুন মেনে নরমালাইজেশনের চর্চা করতে হবে। এজন্য বইয়ের উদাহরণ, অনুশীলনী ও ক্লাসে শিক্ষকের দেখানো উদাহরণ বুঝতে হবে ও চর্চা করতে হবে। একসময় নরমালাইজেশন করা ডাল-ভাত হয়ে যাবে, তখন আর সূত্র বা নিয়ম মনে করে চিন্তা করতে হবে না।

লেখক – মোঃ শফিউজ্জামান রাজিব, বিগডাটা প্রফেশনাল।