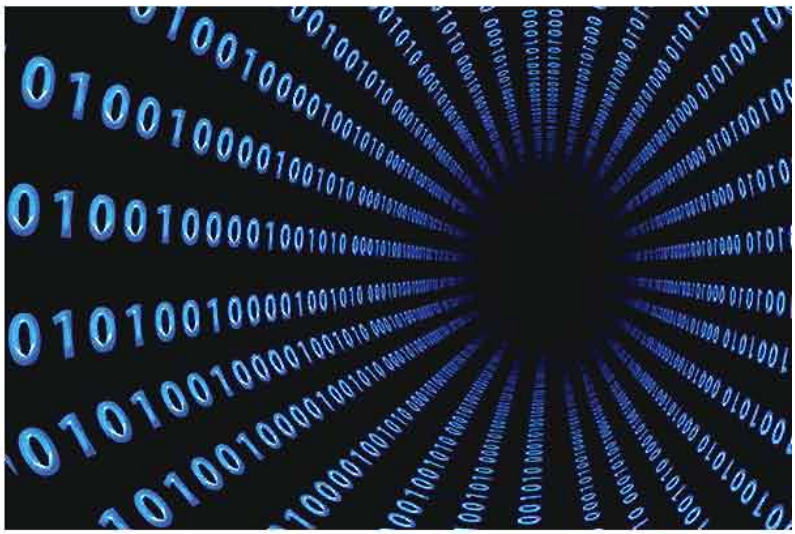
ষষ্ঠ অধ্যায়

ডেটাবেজ ম্যানেজমেন্ট সিস্টেম

Database Management System



সুশৃঙ্খলভাবে ডেটা সংগ্রহ, সংরক্ষণ এবং প্রক্রিয়াকরণ বর্তমান বিশ্বের একটি বড় চ্যালেঞ্জ

বর্তমান বিশ্বে আমাদের চারপাশের প্রায় সবকিছুই অনলাইনের মাধ্যমে হতে শুরু করেছে। আমরা কেনাকাটা কিংবা বাজার করি অনলাইনে, ব্যাংকিং করি অনলাইনে, ইলেক্ট্রিসিটির বিল দেই অনলাইনে, ট্রেনের টিকেট কিনি অনলাইনে। এ ধরনের প্রত্যেকটি কাজের জন্য কোথাও না কোথাও অসংখ্য তথ্য সংরক্ষণ করতে হয়। একসময় যে কাজগুলো করতে অসংখ্য লেজার বই কিংবা কাগজের উপর নির্ভর করতে হতো এখন সেগুলো করা হয় ডেটাবেজ ম্যানেজমেন্ট সিস্টেম দিয়ে। সেগুলো আমাদের জীবনকে অনেক সহজ করে তুললেও সেখানে এখনো চ্যালেঞ্জের অভাব নেই। সেগুলো অনেক সময় প্রয়োজনমতো বড় করা যায় না, দ্রুত প্রক্রিয়া করা যায় না কিংবা সাইবার দুর্বৃত্তরা মাঝে মাঝেই ডেটা হাতিয়ে নেয়। কাজেই কম্পিউটার বিজ্ঞানীরা ক্রমাগতভাবে ডেটাবেজ ম্যানেজমেন্ট সিস্টেমকে আরো শক্তিশালী করার চেষ্টা করে যাচ্ছেন। এই অধ্যায়ে শিক্ষার্থীদের ডেটাবেজ ম্যানেজমেন্ট সিস্টেম সম্পর্কে একটি প্রাথমিক ধারণা দেওয়া হয়েছে।

এ অধ্যায় পাঠ শেষে শিক্ষার্থীরা—

- ডেটাবেজ ম্যানেজমেন্ট-এর ধারণা ব্যাখ্যা করতে পারবে;
- ডেটাবেজ ম্যানেজমেন্ট-এর কার্যাবলি বিশ্লেষণ করতে পারবে;
- রিলেশনাল ডেটাবেজ ম্যানেজমেন্ট সিস্টেমের ধারণা ব্যাখ্যা করতে পারবে;
- রিলেশনাল ডেটাবেজ ম্যানেজমেন্ট সিস্টেমের বৈশিষ্ট্য ব্যাখ্যা করতে পারবে;
- রিলেশনাল ডেটাবেজ ম্যানেজমেন্ট সিস্টেম বর্ণনা করতে পারবে;
- ডেটাবেজ সিকিউরিটির ধারণা ব্যাখ্যা করতে পারবে;
- ডেটাবেজ সিকিউরিটির গুরুত্ব বিশ্লেষণ করতে পারবে;
- ডেটা এনক্রিপশনের প্রয়োজনীয়তা ব্যাখ্যা করতে পারবে;
- ডেটা এনক্রিপশনের উপায়সমূহ ব্যাখ্যা করতে পারবে।

ব্যাবহারিক

- ডেটাবেজ তৈরি করতে পারবে।

৬.১ ডেটাবেজ ম্যানেজমেন্ট (Database Management)

শিক্ষাপ্রতিষ্ঠান, ব্যবসায় প্রতিষ্ঠান, অফিস-আদালত, এমনকি আমাদের নিজেদের ঘর-গেরস্থালির কাজেও আমাদেরকে নানান রকম তথ্য নিয়ে কাজ করতে হয়। স্কুল-কলেজের কথাই চিন্তা করা যাক। শিক্ষার্থী ভর্তি, ক্লাস রুটিন তৈরি, শিক্ষার্থীদের উপস্থিতি, পরীক্ষার রুটিন তৈরি, পরীক্ষার ফলাফল প্রকাশ ও সংরক্ষণ, শিক্ষার্থীদের বেতনের হিসেব রাখা ইত্যাদি নানান কাজেই অনেক তথ্য তৈরি ও সংরক্ষণের প্রয়োজন হয়। যুগ যুগ ধরে মানুষ খাতা-কলমের মাধ্যমেই এসব হিসাব করে আসছে। কিন্তু কম্পিউটারের আবির্ভাব এসব কাজকে মানুষের জন্য সহজ করে দিয়েছে। কম্পিউটারের তথ্য ধারণ, সংরক্ষণ ও প্রক্রিয়াকরণের ক্ষমতা মানুষের চেয়ে অনেক অনেক বেশি। আর এই ক্ষমতাকে কাজে লাগিয়ে মানুষ এমন সফটওয়্যার তৈরি করেছে যা বিপুল পরিমাণ তথ্য ধারণ করতে পারে, সংরক্ষণ করতে পারে এবং সেসব তথ্য বিশ্লেষণ করে প্রয়োজনীয় প্রশ্নের উত্তরও দিতে পারে।

ধরা যাক, অসুস্থতা এবং অন্য কোনো কারণে একজন শিক্ষার্থীকে মাঝে মাঝে স্কুলে অনুপস্থিত থাকতে হয়েছে। তার অভিভাবক জানতে চান শিক্ষার্থীটি গত তিন মাসে ঠিক কতদিন স্কুলে ছিল। এই কাজটি করার জন্য তাঁকে স্কুলে যেতে হবে, তারপর তাঁর সন্তানের যেসব শিক্ষক আছেন, তাদের সঙ্গে দেখা করতে হবে। শিক্ষকেরা তখন গত তিন মাসের হাজিরা খাতা বের করবেন। সেই খাতা থেকে খুঁজে দেখবেন ওই শিক্ষার্থী কতদিন ক্লাসে উপস্থিত ছিল, পুরোটাই খুবই সময়সাপেক্ষ কাজ। কিন্তু ওই স্কুলে যদি সব তথ্য ব্যবস্থাপনার কাজ একটি কম্পিউটারে ডেটাবেজ ম্যানেজমেন্ট সফটওয়্যারের মাধ্যমে করা হতো, তাহলে এই তথ্য এক সেকেন্ডের মধ্যেই বের করা সম্ভব হতো। তথ্য সংরক্ষণ করার কাজটি করে ডেটাবেজ আর সেই ডেটাবেজকে ঠিকমতো পরিচালনা করার জন্য যে সফটওয়্যার, তাকেই বলা হয় ডেটাবেজ ম্যানেজমেন্ট সিস্টেম।

৬.১.১ কম্পিউটারের মেমোরি ও ফাইল

কম্পিউটারের প্রোগ্রাম যখন কোনো ডেটা নিয়ে কাজ করে, সেই ডেটা অস্থায়ী মেমোরিতে লোড করে তারপর কাজ করে। এই অস্থায়ী মেমোরিকে বলা হয় র‍্যাম (RAM)। কম্পিউটার বন্ধ হয়ে গেলে র‍্যাম থেকে ডেটা মুছে যায়। আবার প্রোগ্রাম বন্ধ করে দিলেও প্রোগ্রাম র‍্যাম-এর ডেটার নিয়ন্ত্রণ হারিয়ে ফেলে। তাই যেসব তথ্য সংরক্ষণের দরকার হয়, সেগুলোকে স্থায়ী মেমোরি, যেমন—হার্ড ডিস্কে সংরক্ষণ করা হয়। হার্ড ডিস্কের যেসব ডেটা আমরা ব্যবহার করি, সেগুলোকে ফাইল (file) নামক একটি ব্যবস্থার মাধ্যমে আমরা অ্যাকসেস করি। যেমন— একটি টেক্সট ফাইলে আমরা বিভিন্ন তথ্য লিখে সেভ করে রাখতে পারি। তারপরে চাইলে কম্পিউটার বন্ধ করে দিতে পারি। আবার যখন কম্পিউটার চালু করব, তখন চাইলে সেই ফাইলটি খুলে আগে লেখা তথ্য দেখতে পারি।

কম্পিউটারকে যখন মানুষ বিভিন্ন তথ্য সংরক্ষণ করার কাজে ব্যবহার করা শুরু করল, তখন বিভিন্ন প্রোগ্রাম লেখা হতো, যেগুলো তথ্য প্রক্রিয়াকরণের পরে ফাইলে সংরক্ষণ করত। সেই ফাইলের তথ্য পরিবর্তন বা তথ্য নিয়ে অন্য কোনো কাজ করতে হলে আবার নতুন প্রোগ্রাম লিখে কাজগুলো করতে হতো। যেমন ধরা যাক, একটি অ্যাড্রেস বুক (address book), যেখানে বিভিন্নজনের নাম, ঠিকানা, ফোন নম্বর ইত্যাদি তথ্য সংরক্ষিত থাকবে এবং প্রয়োজন অনুসারে সেই তথ্য খুঁজে বের করা যাবে। তাহলে নতুন তথ্য যোগ করার

(একে ডেটা এন্ট্রি– data entry বলা হয়) জন্য প্রোগ্রাম লিখতে হবে। তথ্য খুঁজে বের করার জন্যও প্রোগ্রাম লিখতে হবে। এখন কেউ যদি বলে ফোন নম্বরের পর ইমেইল ঠিকানাও সংরক্ষণ করতে হবে, তখন আবার ডেটা এন্ট্রি করার প্রোগ্রামটি এবং বাকি সব প্রোগ্রাম পরিবর্তন করতে হবে। তারপর ধরা যাক, কেউ বলল, নাম দিয়ে কারো তথ্য খুঁজে বের করার পাশাপাশি ইমেইল ঠিকানা দিয়েও খোঁজার ব্যবস্থা রাখতে হবে, তখন আবার নতুন ফাংশন লিখতে হবে। এভাবে তথ্য ব্যবস্থাপনার কাজটি বেশ জটিল ও প্রোগ্রামারদের জন্য পরিশ্রমসাধ্য হয়ে যায়। ডেটাবেজ ম্যানেজমেন্ট সফটওয়্যার সেই কাজটি খুব সহজ করে দেয়।

৬.১.২ ডেটাবেজ

আক্ষরিক অর্থে, ডেটাবেজ হচ্ছে ডেটার সমাহার, অর্থাৎ যেখানে অনেক ডেটা থাকে। তবে সফটওয়্যারের জগতে ডেটাবেজ হচ্ছে এমন একটি সফটওয়্যার যেখানে প্রচুর পরিমাণ তথ্য একসঙ্গে সংরক্ষণ করা যায়, দরকারি তথ্য বের করা যায়, নতুন তথ্য যোগ করা যায় এবং প্রয়োজনমতো কোনো তথ্য পরিবর্তন, পরিবর্ধন ও মুছে ফেলা যায়। আর সেই ডেটাবেজকে সুষ্ঠুভাবে পরিচালনা করার জন্য কিছু প্রোগ্রাম বা সফটওয়্যার মিলেই গঠিত হয় ডেটাবেজ ম্যানেজমেন্ট সিস্টেম।

ডেটাবেজকে মোটাদাগে দুভাগে ভাগ করা যায় : রিলেশনাল ডেটাবেজ (Relational Database) ও নোএসকিউএল (NoSQL)। রিলেশনাল ডেটাবেজের ধারণা প্রায় ৫০ বছর আগের, তবে এখনো এটি সবচেয়ে বেশি ব্যবহৃত ও গুরুত্বপূর্ণ ডেটাবেজ। আর নোএসকিউএল ডেটাবেজ অপেক্ষাকৃত নতুন এবং বেশ কিছু ক্ষেত্রে, বিশেষ করে ওয়েবভিত্তিক বিভিন্ন অ্যাপ্লিকেশনে এর ব্যবহার দিন দিন বাড়ছে। তবে বিভিন্ন সীমাবদ্ধতার কারণে নোএসকিউএল সবক্ষেত্রে ব্যবহার করা সম্ভব হয় না।

৬.২ রিলেশনাল ডেটাবেজ (Relational Database)

রিলেশনাল ডেটাবেজ-এ ডেটাকে এক বা একাধিক টেবিলে সংরক্ষণ ও প্রকাশ করা হয়। কিছু কিছু টেবিলের মধ্যে অনেক সময় সম্পর্ক (relation) থাকতে পারে। যেমন– ধরা যাক, একটি স্কুলের ডেটাবেজে ওই স্কুলের শিক্ষক ও শিক্ষার্থীর নানান ধরনের তথ্য থাকতে পারে। আবার পরীক্ষার ফলাফল, ক্লাসের রুটিন, এসব তথ্যও ডেটাবেজে থাকতে পারে। একই ধরনের সব তথ্য একটি টেবিলে থাকবে। যেমন শিক্ষকদের তথ্যের জন্য teacher টেবিল, শিক্ষার্থীদের তথ্যের জন্য student টেবিল, পরীক্ষার ফলাফল রাখার জন্য result টেবিল তৈরি করতে হবে। student টেবিল ও result টেবিলের মধ্যে একটি সম্পর্ক থাকবে, যেন দুটো টেবিল থেকে একজন শিক্ষার্থীর ব্যক্তিগত তথ্য ও পরীক্ষার ফলাফল সংক্রান্ত তথ্য একসঙ্গে পাওয়া যায়। আর এসব টেবিল মিলে তৈরি হবে school ডেটাবেজ।

একটি ডেটাবেজ টেবিলের দুটি অংশ থাকে, টেবিল হেডার (table header) ও টেবিল বডি (table body)। টেবিল হেডারে থাকে বিভিন্ন কলামের নাম এবং সেই কলামে কী ধরনের ডেটা রাখা হবে তার তথ্য। আর টেবিলের বডিতে থাকে মূল তথ্য। প্রতিটি সারি (row)-তে একটি নির্দিষ্ট বিষয়ের তথ্য থাকে। একটি টেবিলে কী কী ডেটা রাখা হবে এবং সেগুলো কী ধরনের হবে, সেটি আগে ঠিক করতে হয়। যেমন–

শিক্ষার্থীর টেবিলে থাকতে পারে শিক্ষার্থীর নাম, রোল নম্বর, ক্লাস, বিভাগ/শাখা, অভিভাবকের নাম, অভিভাবকের ফোন নম্বর, বাসার ঠিকানা ইত্যাদি।

ডেটার ধরন বিভিন্ন রকমের হতে পারে। সি প্রোগ্রামিং ভাষায় যেমন নির্দিষ্ট কিছু ডেটা টাইপ রয়েছে, রিলেশনাল ডেটাবেজেও তেমনি কিছু ডেটা টাইপ রয়েছে। বিভিন্ন ডেটাবেজ নির্মাতারা নিজেদের মতো ডেটা টাইপ নির্দিষ্ট করে দেন, তবে বেশ কিছু ডেটা টাইপ সব ডেটাবেজেই পাওয়া যাবে। যেমন– টেক্সট (text), পূর্ণসংখ্যা (integer), দশমিকযুক্ত সংখ্যা (decimal number), তারিখ (date) ইত্যাদি।

এখন আমরা একটি টেবিলের উদাহরণ দেখি :

টেবিলের নাম : শিক্ষার্থী

শিক্ষার্থীর নাম (টেক্সট)	রোল নম্বর (পূর্ণসংখ্যা)	শ্রেণি (পূর্ণসংখ্যা)	শাখা (টেক্সট)	অভিভাবকের নাম (টেক্সট)	ফোন নম্বর (টেক্সট)
মিজানুর রহমান	১	৪	দিবা	আব্দুর রহমান	০২০৩০২
মোশাররফ হোসেন	২	৪	দিবা	সেলিনা খাতুন	০২০৩০৪
সৌরভ দাস	১	৫	প্রভাতি	অজয় দাস	০৩০৪০২
শাকিল মিয়া	৩	৫	প্রভাতি	মনসুর মিয়া	

এই শিক্ষার্থী টেবিলের প্রথম সারিটি হচ্ছে টেবিলের হেডার। এই সারিতে যেসব ঘর আছে, প্রতিটি হচ্ছে একটি কলাম (column)-এর নাম এবং তার ডেটা টাইপ (মানে ওই কলামে কী ধরনের ডেটা থাকবে)। যেমন– শিক্ষার্থীর নাম হচ্ছে একটি কলামের নাম এবং সেখানে টেক্সট টাইপের ডেটা থাকবে। এর নিচে যত ঘর থাকবে, সব ঘরে বিভিন্ন শিক্ষার্থীর নাম থাকবে, অন্য কোনো তথ্য থাকতে পারবে না। আর দ্বিতীয় সারি থেকে প্রতিটি সারিতে একজন করে শিক্ষার্থীর তথ্য দেওয়া আছে। যেমন– দ্বিতীয় সারির প্রথম ঘরে আছে মিজানুর রহমান, যা একজন শিক্ষার্থীর নাম, তারপরের ঘরে আছে তার রোল নম্বর, তারপরের ঘরে আছে তার শ্রেণি, অর্থাৎ সে কোন শ্রেণিতে পড়ছে, ইত্যাদি। একটি সারিতে কেবল একজন শিক্ষার্থীর তথ্য থাকবে, কখনো একাধিক শিক্ষার্থীর তথ্য থাকবে না।

প্রতিটি সারিকে ইংরেজিতে বলে রো (row)। এগুলোকে রেকর্ড (record)-ও বলা হয়ে থাকে। আর টেবিলের প্রতিটি ঘর হচ্ছে একেকটি ফিল্ড (field)।

বর্তমানে বিশ্বব্যাপী জনপ্রিয় ও বহুল ব্যবহৃত রিলেশনাল ডেটাবেজ হচ্ছে ওরাকল (Oracle), মাইএসকিউএল (MySQL), মাইক্রোসফট এসকিউএল সার্ভার (Microsoft SQL Server), পোস্টগ্রেস (PostgreSQL), মাইক্রোসফট অ্যাকসেস (Microsoft Access) ও এসকিউলাইট (SQLite)। এগুলোর মধ্যে মাইসিক্যুয়েল, পোস্টগ্রেস ও এসকিউলাইট হচ্ছে ফ্রি এবং ওপেন-সোর্স (free & open source) ডেটাবেজ। অর্থাৎ এগুলো ব্যবহার করার জন্য টাকা দিতে হয় না, এবং এগুলোর সোর্সকোডও উন্মুক্ত।

নোট: উচ্চারণের সুবিধার জন্য এসকিউএল শব্দটি অনেকে সিক্যুয়েল বলে উচ্চারণ করে। SQL শব্দটির পূর্ণরূপ, স্ট্রাকচারড কুয়েরি ল্যাঙ্গুয়েজ (Structured Query Language)

৬.২.১ নাল ভ্যালু (Null Value)

অনেক সময় ডেটাবেজ টেবিলে কিছু কিছু রেকর্ডে কোনো কলামের মান যদি অজানা থাকে, তখন সেখানে Null (বা NULL) ব্যবহার করা হয়। যেমন– student টেবিলে ফোন নম্বর বলে একটি কলাম আছে। এখন সবার যে ফোন নম্বর থাকতেই হবে, এমন কোনো কথা নেই, যেমন– মানিক মিয়া নামক শিক্ষার্থীর ফোন নম্বর নেই, তাই তার রেকর্ডে ফোন নম্বর ফিল্ডটি ফাঁকা রয়েছে। ডেটাবেজ ধরে নেবে এর মান নাল (NULL)। আবার, ধরা যাক ওই টেবিলে মাসিক পারিবারিক আয় নামে আরেকটি কলাম আছে। মাসিক পারিবারিক আয় একটি সংখ্যা। এখন, কেউ যদি তার মাসিক পারিবারিক আয় প্রকাশ করতে না চায় তো, সেখানে কিন্তু 0 বসবে না, বরং সেটি হবে ফাঁকা বা NULL। কারণ এক্ষেত্রে 0 মানে তার পরিবারের কারো কোনো আয় নেই। আবার অনেক সময় একটি টেবিলে বিভিন্ন মানুষের পেশা যদি রাখার প্রয়োজন হয়, সেখানেও সবার যে পেশা থাকতেই হবে, এমন কোনো কথা নেই। যেমন– ওই টেবিলে যদি তিন বছর বয়সি একটি শিশুর তথ্য থাকে, তাহলে তার পেশা হবে NULL, কারণ তাকে শিক্ষার্থী, চাকুরিজীবী, ব্যবসায়ী কিংবা বেকার–কোনো পেশাতেই ফেলা যাবে না। এখন পেশা যদি টেক্সট টাইপের হয়, তখন কিন্তু ফাঁকা স্ট্রিং (অর্থাৎ '') বসানো যাবে না, বরং ঘরটি ফাঁকা রাখতে হবে। ফাঁকা ঘরটিকে ডেটাবেজ NULL বলে বিবেচনা করবে।

৬.২.২ প্রাইমারি কি (Primary Key)

প্রাইমারি কি হচ্ছে একটি টেবিলের নির্দিষ্ট কলাম, যেটি দিয়ে প্রতিটি রেকর্ডকে আলাদাভাবে চিহ্নিত করা যায়। শিক্ষার্থী টেবিলে কোন কলাম দিয়ে প্রতিটি শিক্ষার্থীকে আলাদাভাবে চিহ্নিত করা যায়? নাম দিয়ে করা যাবে না, কারণ একই শ্রেণিতে কিংবা আলাদা শ্রেণিতে একই নামে একাধিক শিক্ষার্থী থাকতে পারে। টেবিলে আমরা দেখতে পাচ্ছি, 'মিজানুর রহমান' নামটি চতুর্থ শ্রেণির একজন শিক্ষার্থীর। কিন্তু ষষ্ঠ শ্রেণিতেও একজন মিজানুর রহমান থাকতে পারে। আবার আমরা যদি বলি, চতুর্থ শ্রেণির মিজানুর রহমান, তখন চতুর্থ শ্রেণিতে যদি একাধিক মিজানুর রহমান থাকে, তাহলে তাকে আলাদাভাবে চিহ্নিত করা যাবে না। তাই নাম প্রাইমারি কি (key) হতে পারবে না। রোল নম্বরও প্রাইমারি কি হতে পারে না, কারণ প্রতিটি শ্রেণিতেই রোল 1, 2, 3 ইত্যাদি রয়েছে। ফোন নম্বরও প্রাইমারি কি হতে পারবে না, কারণ সবার ফোন নম্বর নাও থাকতে পারে। তাহলে উপরে যে শিক্ষার্থী টেবিল তৈরি করা হয়েছে, সেখানে কোনো প্রাইমারি কি নেই। তবে, শ্রেণি, শাখা ও রোল নম্বর– এই তিনটি কলাম মিলে একটি প্রাইমারি কি হতে পারে, কারণ এই তিনটি তথ্য একসঙ্গে করলে আমরা প্রতিটি শিক্ষার্থীকে আলাদা করতে পারি। যখন একাধিক কলাম মিলে প্রাইমারি কি তৈরি হয়, তখন তাকে বলা হয় কম্পোজিট কি (composite key)।

ডেটাবেজে টেবিল তৈরির সময় কোন কলামটি প্রাইমারি কি হতে পারে তা চিহ্নিত করতে পারলে সেটি আলাদাভাবে উল্লেখ করে দিতে হয়। আবার কোনো কোনো সময় প্রাইমারি কি চিহ্নিত করা সম্ভব নাও হতে পারে। তখন শুরুতে একটি কলাম যোগ করা হয়। এটি একটি সংখ্যার কলাম হবে এবং প্রতিটি রেকর্ড বা রো-এর জন্য আলাদা হবে। সাধারণত, টেবিলে id নামের একটি কলাম যোগ করা হয়, যেটি ইন্টিজার টাইপের ডেটা ধারণ করে এবং এর সঙ্গে অটো ইনক্রিমেন্ট (Auto Increment) বৈশিষ্ট্য জুড়ে দেওয়া হয়, যেন প্রতিটি রো ইনসার্ট (insert) করার সময় এর মান এক-এক করে বাড়ে (এই কলামের জন্য তাই কোনো মান নিজে থেকে দিতে হয় না, ডেটাবেজ সিস্টেম নিজেই এটি নিয়ন্ত্রণ করে)।

রিলেশনাল ডেটাবেজে সব টেবিলেই প্রাইমারি কি থাকতে হয়। যদিও প্রাইমারি কি ছাড়াও টেবিল তৈরি করা যায়। সেক্ষেত্রে অনেক সময় ডেটাবেজ নিজেই একটি প্রাইমারি কি তৈরি করে নেয়।

শিক্ষার্থী টেবিলে শ্রেণি, শাখা ও রোল– এই তিনটি কলাম মিলে প্রাইমারি কি তৈরি করা যায়। তবে এখানে একটি সমস্যা হতে পারে। এভাবে টেবিল তৈরি করলে আমরা কেবল বর্তমান শিক্ষার্থীদের তথ্যই রাখতে পারব। অতীতের শিক্ষার্থীদের তথ্য রাখা সম্ভব হবে না, যেমন– পাঁচ বছর আগের কোনো শিক্ষার্থী, যে পড়ত সপ্তম শ্রেণির দিবা শাখায় এবং যার রোল নম্বর ছিল দুই, তাকে আলাদাভাবে বের করা যাবে না। তাই আমরা নতুন একটি কলামে প্রতিটি শিক্ষার্থীর জন্য পৃথক একটি আইডি দিতে পারি। আবার কোনো কোনো প্রতিষ্ঠানে রোল নম্বর এমনভাবে তৈরি করা হয়, যেন রোল নম্বর দেখলেই বোঝা যায় যে, সে কোন বছরের কোন ক্লাসের কোন শাখার কত নম্বর শিক্ষার্থী। আবার অনেক প্রতিষ্ঠানে একে রেজিস্ট্রেশন নম্বরও বলা হয়, যা একজন শিক্ষার্থীর জন্য সবসময় একই থাকে। ওপরের ক্লাসে উঠলে রোল নম্বর পরিবর্তন হবে, কিন্তু রেজিস্ট্রেশন নম্বর পরিবর্তন হবে না।

বাংলাদেশে প্রতিটি প্রাপ্তবয়স্ক মানুষেরই একটি করে জাতীয় পরিচয়পত্র আছে (যাকে ন্যাশনাল আইডি কার্ড-National ID Card-ও বলা হয়)। সেখানে কিন্তু প্রতিটি মানুষকে আলাদা নম্বর দিয়ে চিহ্নিত করা হয়, এবং কখনোই দুজন মানুষের নম্বর একরকম হতে পারবে না।

চিত্র 6.1 : বাংলাদেশের জাতীয় পরিচয়পত্র

তাই বিভিন্ন টেবিলে যদি প্রাপ্তবয়স্ক মানুষের তথ্য রাখা হয়, সেসব জায়গায় জাতীয় পরিচয়পত্রের নম্বর প্রাইমারি কি হিসেবে ব্যবহার করা যেতে পারে।

৬.২.৩ ডেটাবেজ রিলেশন (Database Relation)

ডেটাবেজ রিলেশন বলতে আসলে ডেটাবেজের টেবিলগুলোর মধ্যে সম্পর্ক বোঝানো হয়। একটি ডেটাবেজে এক বা একাধিক টেবিল থাকতে পারে। যখন একাধিক টেবিল থাকে, তখন প্রায়শই টেবিলগুলোর মধ্যে সম্পর্ক বা রিলেশন (relation) থাকে। এই রিলেশন আবার তিন ধরনের হতে পারে :

১. ওয়ান টু ওয়ান (one to one)
২. ওয়ান টু মেনি (one to many)
৩. মেনি টু মেনি (many to many)

১. ওয়ান টু ওয়ান রিলেশন

দুটি টেবিলের মধ্যে যদি ওয়ান টু ওয়ান রিলেশন থাকে, তবে একটি টেবিলের একটি রো-এর সঙ্গে অন্য টেবিলের একটিমাত্র রো-এর সম্পর্ক খুঁজে পাওয়া যাবে। ধরা যাক, শিক্ষার্থীদের সাধারণ তথ্য রাখার জন্য একটি টেবিল student_info তৈরি হলো। আবার শিক্ষার্থীদের যোগাযোগের ঠিকানা রাখার জন্য তৈরি করা হলো student_contact টেবিল। টেবিলগুলো নিম্নরূপ :

টেবিলের নাম : student_info

Roll (integer, primary key)	Name (text)	Class (integer)
1	Mizanur Rahman	6
2	Mosharraf Hossain	7
3	Subir Kumar	6

টেবিলের নাম : student_contact

ID (integer, primary key)	Roll (integer)	Phone (text)	Email (text)	Address (text)
1	1	012345678	mizan@email.com	Adabor, Shyamoli, Dhaka
2	2	012345543	mosharraf@email.com	Sector 3, Uttara, Dhaka
3	3	014343678	subir@email.com	College Road, Mymensingh

উপরের টেবিল দুটির মধ্যে ওয়ান টু ওয়ান রিলেশন বিদ্যমান। যেমন– student_info টেবিলের প্রতিটি রো-তে একজন শিক্ষার্থীর তথ্য রয়েছে। এই টেবিলের প্রাইমারি কি হচ্ছে Roll (যদিও প্রাইমারি কি হিসেবে Roll সবসময় সঠিক নয়, তবে এখানে আলোচনার সুবিধার্থে এটি প্রাইমারি কি হিসেবে ঘোষণা করা হয়েছে)। এখন, এই টেবিলে প্রতিটি শিক্ষার্থী (রো)-এর জন্য student_contact টেবিলে একটি রো থাকবে, যেখানে ওই শিক্ষার্থীর যোগাযোগের ঠিকানা (ফোন, ইমেইল, বাসার ঠিকানা) থাকবে। student_info টেবিলে Roll হচ্ছে প্রাইমারি কি, কিন্তু student_contact টেবিলে Roll হচ্ছে ফরেন কি (foreign key)। একটি টেবিলের প্রাইমারি কি অন্য টেবিলে যখন ব্যবহার করা হয়, তখন সেই টেবিলে সেই কি-কে ফরেন কি বলা হয়। এই বিশেষ কি দিয়ে টেবিলদুটি সম্পর্কযুক্ত হয়।

২. ওয়ান টু মেনি রিলেশন

ধরা যাক, শিক্ষার্থীদের পরীক্ষার ফল সংরক্ষণ করার জন্য result নামের একটি টেবিল তৈরি করা হলো। টেবিলের প্রতিটি রো-তে একজন শিক্ষার্থীর একটি বিষয়ে পরীক্ষায় প্রাপ্ত নম্বর থাকবে।

টেবিলের নাম : result

ID (Integer, Primary Key)	Roll (Integer)	Subject (Text)	Marks (Decimal)
1	1	Bangla	70
2	1	English	76
3	2	Bangla	68
4	2	English	81

এখানে দেখা যাচ্ছে যে, student_info টেবিলের একটি রো-এর সঙ্গে result টেবিলের একাধিক রো-এর সম্পর্ক রয়েছে। যেমন- রোল নম্বর 1 যার, তার দুটি বিষয়ে প্রাপ্ত নম্বর result টেবিলে দুটি রো-তে রাখা আছে। student_info ও result টেবিলের মধ্যকার সম্পর্ককে ওয়ান টু মেনি রিলেশন বলা হয়, কারণ প্রথম টেবিলের একটি রো-এর সঙ্গে দ্বিতীয় টেবিলের একাধিক রো-এর সম্পর্ক রয়েছে। result টেবিলের Roll হচ্ছে ফরেন কি।

৩. মেনি টু মেনি

যখন দুটি টেবিল এমনভাবে সম্পর্কযুক্ত হয় যে, প্রথম টেবিলের একটি রো, দ্বিতীয় টেবিলের একাধিক রো-এর সঙ্গে সম্পর্কযুক্ত, আবার দ্বিতীয় টেবিলের একটি রো, প্রথম টেবিলের একাধিক রো-এর সঙ্গে সম্পর্কযুক্ত হয়, তখন তাদের মধ্যকার সম্পর্ককে বলা হয় মেনি টু মেনি রিলেশনশিপ।

ধরা যাক, স্কুলে বিভিন্ন ক্লাব তৈরি করা হয়েছে। যেমন– ফুটবল ক্লাব, ক্রিকেট ক্লাব, দাবা ক্লাব, বিতর্ক ক্লাব, বিজ্ঞান ক্লাব, সাংস্কৃতিক ক্লাব ইত্যাদি। একজন শিক্ষার্থী এক বা একাধিক ক্লাবের সদস্য হতে পারে। আবার একটি ক্লাবে একাধিক শিক্ষার্থী থাকতে পারে। নিচে club টেবিলটি দেওয়া হলো-

টেবিলের নাম : club

Name (Text)	Moderator (Text)	Established (Date)
Cricket Club	Mr. Ruhul Amin	1-1-2000
Football Club	Mr. Shahidul Islam	5-1-1998
Debating Club	Mr. Sumon Kumar	3-7-2002
Chess Club	Ms. Fatema Akhter	1-1-2001

এই টেবিলের প্রাইমারি কি হচ্ছে Name. অর্থাৎ প্রতিটি ক্লাবের অনন্য (unique) নাম থাকবে, একই নামে একাধিক ক্লাব থাকতে পারবে না।

এখন student_info ও club টেবিল দুটির মধ্যে সম্পর্ক স্থাপনের জন্য আমাদেরকে আরেকটি টেবিল তৈরি করতে হবে।

টেবিলের নাম : student_club

Roll (Integer)	club_name (text)
1	Cricket Club
2	Cricket Club
2	Football Club
2	Chess Club
2	Debating Club

মেনি টু মেনি রিলেশনের জন্য এরকম একটি টেবিল তৈরি করতে হয়, যার কাজ হচ্ছে মূল টেবিলদুটি যুক্ত করা।

৬.২.৪ এসকিউএল (SQL)

রিলেশনাল ডেটাবেজে এসকিউএল (SQL : Structured Query Language) নামক প্রোগ্রামিং ভাষার সাহায্যে ডেটাবেজে তথ্য লেখা, পড়া, পরিবর্তন করা ও অন্যান্য কাজ করা হয়। এসকিউএল ভাষার নির্ধারিত নিয়ম-কানুন থাকলেও বিভিন্ন বাণিজ্যিক ডেটাবেজ তাদের নিজস্ব কুয়েরি ভাষা ব্যবহার করে, যা প্রমিত (standard) এসকিউএল-এর বেশ কাছাকাছি।

এসকিউএল-এর সঙ্গে সাধারণ প্রোগ্রামিং ভাষাগুলোর মূল পার্থক্য কোথায়? একজন প্রোগ্রামার যখন একটি নির্দিষ্ট সমস্যা সমাধানের জন্য প্রোগ্রাম লিখেন, তখন আসলে সেই সমস্যাটি সমাধানের যে অ্যালগরিদম, সেটিকেই প্রোগ্রামিং ভাষার মাধ্যমে প্রকাশ করেন। এসব প্রোগ্রামে কম্পিউটারের জন্য সুনির্দিষ্টভাবে বলা থাকে যে, কীভাবে ও কোন ধাপে কী কাজ করা হবে। কম্পিউটার শুধু সেই নির্দেশনা অনুসরণ করে। আর এসকিউএল কুয়েরি লেখার সময় বলে দিতে হয় যে, ডেটাবেজ সিস্টেমের কাছ থেকে কোন তথ্য চাওয়া হচ্ছে বা কোন তথ্য রাখতে হবে। অর্থাৎ কী করতে হবে সেটি বলে দিতে হয়। আর সেই কাজটি কীভাবে করা হবে, সেটি নির্ভর করে ডেটাবেজ সিস্টেমের উপর। এ জন্য এসকিউএল-কে বলা হয় ডিক্লারেটিভ (declarative) প্রোগ্রামিং ভাষা। সি এবং সি-এর মতো অন্যান্য প্রোগ্রামিং ভাষাকে বলা হয় প্রসিডিউরাল (procedural) প্রোগ্রামিং ভাষা। এসকিউএল ভাষাটি এমনভাবে তৈরি করা হয়েছে যেন, কেবল প্রোগ্রামাররাই নন, যারা প্রোগ্রামিং জগতের বাইরের মানুষ, তারাও যেন সহজে এটি শিখে ব্যবহার করতে পারেন।

এসকিউএলকে আবার কয়েক ভাগে ভাগ করা যায়। এর মধ্যে গুরুত্বপূর্ণ দুটি হচ্ছে : ডেটা ডেফিনেশন ল্যাঙ্গুয়েজ (Data Definition Language বা সংক্ষেপে DDL) ও ডেটা ম্যানিপুলেশন ল্যাঙ্গুয়েজ (Data Manipulation Language বা সংক্ষেপে DML)।

ডেটা ডেফিনেশন ল্যাঙ্গুয়েজ

ডেটাবেজের টেবিল তৈরি করা, টেবিল মুছে ফেলা, ইনডেক্স তৈরি করা ইত্যাদি কাজ করার জন্য ডেটা ডেফিনেশন ল্যাঙ্গুয়েজ ব্যবহার করা হয়। যেমন– একটি টেবিল তৈরি করতে গেলে টেবিলের নাম, টেবিলের বিভিন্ন কলামের নাম ও সেখানে কী ধরনের ডেটা থাকবে, ইনডেক্স ইত্যাদি বলে দিতে হয়।

ডেটা ম্যানিপুলেশন ল্যাঙ্গুয়েজ

ডেটা ম্যানিপুলেশন ল্যাঙ্গুয়েজের সাহায্যে একটি টেবিলের ডেটার উপর বিভিন্ন ধরনের কুয়েরি চালানো হয়, যেমন– ডেটা পড়া, ডেটা পরিবর্তন করা, ডেটা মুছে ফেলা ইত্যাদি।

৬.৩ ডেটাবেজ তৈরি (Creating Database)

ডেটাবেজ তৈরি সংক্রান্ত ৬.৩ সেকশনটি পুরোপুরি ব্যাবহারিক। প্রোগ্রামিং করার ব্যবস্থা আছে (কম্পিউটারে কিংবা স্মার্টফোনে) শুধু সেরকম পরিবেশে পরের অংশটুকু শিক্ষার্থীর জন্য অর্থপূর্ণ বলে বিবেচিত হবে।

SQLite

এসকিউলাইট একটি ফ্রি ও ওপেন সোর্স ডেটাবেজ। ওয়েব, ডেস্কটপ ও মোবাইল অ্যাপ্লিকেশনে এই ডেটাবেজ ব্যবহার করা হয়। প্রচলিত অনেক ডেটাবেজের তুলনায় এর ব্যবহার অপেক্ষাকৃত সহজ বলে ডেটাবেজ শেখার জন্যও এটি বেশ জনপ্রিয়।

ইনস্টল করার প্রক্রিয়া

এসকিউলাইট-এর অফিশিয়াল ওয়েবসাইট (https://www.sqlite.org/download.html) থেকে এটি ডাউনলোড করা যাবে। ডাউনলোড করার পরে ইনস্টল করতে হবে। তাহলে কমান্ড লাইন অথবা টার্মিনাল থেকে বিভিন্ন কমান্ড দিয়ে এসকিউলাইট ব্যবহার করা যাবে। এছাড়া বেশ কিছু গ্রাফিক্যাল ইউজার ইন্টারফেস সমৃদ্ধ সফটওয়্যার রয়েছে যার মাধ্যমে এসকিউলাইট সহজে ব্যবহার করা যায়। যেমন - *DB Browser for SQLite, SQLiteStudio* ইত্যাদি। ইন্টারনেট থেকে সফটওয়্যারগুলো বিনামূল্যে ডাউনলোড করা যাবে। এছাড়া অ্যান্ড্রয়েড অপারেটিং সিস্টেমচালিত মোবাইল ফোনেও এসকিউলাইট ইনস্টল করা থাকে। বিভিন্ন অ্যাপের মাধ্যমে এটি ব্যবহার করা যায়।

এসকিউলাইটে নতুন ডেটাবেজ তৈরি করতে হলে, টার্মিনাল চালু করে সেখানে কমান্ড লিখতে হবে sqlite 3 তারপর একটি স্পেস দিয়ে ডেটাবেজের নাম।

```
$ sqlite3 school.db
```

আবার school.db ডেটাবেজ ইতোমধ্যে তৈরি করা হয়ে থাকলেও একই কমান্ড ব্যবহার করে এসকিউলাইট সফটওয়্যার চালু করতে হবে। সফটওয়্যার চালু হওয়ার পরে সেখানে বিভিন্ন কমান্ড দেওয়া যাবে। যেমন এসকিউলাইট বন্ধ করতে হলে লিখতে হবে .quit।

```
sqlite> .quit
```

৬.৩.১ কুয়েরি ব্যবহার

টেবিল তৈরি

স্কুলের ডেটাবেজে আমরা বিভিন্ন টেবিল তৈরি করব। প্রথমেই ধরা যাক শিক্ষার্থীর টেবিল। শিক্ষার্থীর টেবিলে কী কী তথ্য থাকতে পারে? (আমরা তথ্যগুলোর পাশে ব্র্যাকেটে ইংরেজি শব্দটিও লিখব যেন এসকিউলাইটে টেবিল তৈরির সময় আমরা সেটি ব্যবহার করতে পারি)।

- শিক্ষার্থীর নাম (name)
- কোন শ্রেণিতে পড়ে (class)
- শিক্ষার্থীর রোল নম্বর (roll)
- কোন শাখার অন্তর্ভুক্ত (দিবা, বা প্রভাতি, কিংবা ক, খ, গ ইত্যাদি) (section)

টেবিল তৈরি করার জন্য CREATE TABLE কুয়েরি ব্যবহার করতে হবে। এই কুয়েরি লেখার নিয়ম (সিনট্যাক্স / syntax) হচ্ছে এরকম–

```
CREATE TABLE table_name (column_name column_type, ...);
```

এখানে table_name-এর জায়গায় যেই টেবিল তৈরি করা হবে, তার নাম লিখতে হবে। আর প্রথম বন্ধনীর ভেতরে প্রতিটি কলামের নাম ও একটি স্পেস দিয়ে সেই কলামের ডেটা টাইপ লিখতে হবে। আর একাধিক কলামের তথ্য কমা দিয়ে পৃথক করা থাকবে।

উপরে পরিকল্পিত student টেবিল তৈরি করতে হলে লিখতে হবে–

```
CREATE TABLE student (name TEXT, class INTEGER, roll
INTEGER, section TEXT);
```

কোনো টেবিল মুছে ফেলতে হলে DROP TABLE কুয়েরি ব্যবহার করতে হবে–

```
DROP TABLE [টেবিলের নাম];
```

যেমন– student টেবিলটি মুছে ফেলতে হলে লিখতে হবে,

```
DROP TABLE student;
```

এখন আবার CREATE TABLE কুয়েরি ব্যবহার করে টেবিলটি আবার তৈরি করা যাবে।

নোট : এসকিউএল ভাষার কমান্ডগুলো ইংরেজি বড়হাতের অক্ষরে বা ছোট হাতের অক্ষরে উভয়ভাবেই লেখা যায়। অর্থাৎ, CREATE বা create দুটিই একই কমান্ড বোঝায়। তবে, প্রচলিত রীতি হচ্ছে ইংরেজি বড় হাতের অক্ষর ব্যবহার করে লেখা।

টেবিলে ডেটা রাখা ও টেবিল থেকে ডেটা পড়া

কোনো টেবিলে ডেটা রাখতে চাইলে, INSERT কুয়েরি ব্যবহার করতে হবে। এই কুয়েরি লেখার নিয়ম হচ্ছে,

```
INSERT INTO টেবিলের নাম (প্রথম কলামের নাম, দ্বিতীয় কলামের নাম, তৃতীয় কলামের নাম
...) VALUES (প্রথম কলামের ডেটা, দ্বিতীয় কলামের ডেটা, তৃতীয় কলামের ডেটা ...);
```

যেমন– student টেবিলে Mizanur Rahman নামের একজন শিক্ষার্থী, যে কি না নবম শ্রেণির morning শাখায় পড়ে এবং রোল নম্বর 3, তার তথ্য রাখতে হলে নিচের মতো করে কুয়েরি লিখতে হবে–

```
INSERT INTO student (name, class, roll, section) VALUES
('Mizanur Rahman', 9, 3, 'morning');
```

টেবিলে ডেটা ঠিকমতো রাখা হলো কি না, সেটি দেখার জন্য এখন টেবিলের ডেটা পড়া হবে। সেজন্য SELECT কুয়েরি লিখতে হবে। এই কুয়েরি লেখার নিয়ম হচ্ছে–

```
SELECT [কলামের নাম] কিংবা * FROM [টেবিলের নাম]; (একাধিক কলামের জন্য প্রতিটি
কলামের নাম কমা দিয়ে পৃথক করে দিতে হবে)
SELECT * FROM student;
```

নোট : এসকিউলাইট কমান্ড লাইনে SELECT কুয়েরি-এর আউটপুট সুন্দর করে দেখতে চাইলে নিচের কমান্ড দুটি আগে দিতে হবে,

```
sqlite> .mode column
sqlite> .headers on
sqlite> select * from student;
```

আবার যদি শুধু নাম আর শ্রেণি দেখতে চাই, তাহলে লিখতে হবে—

```
sqlite> SELECT name, class FROM student;
```

পরবর্তী কিছু কাজের সুবিধার জন্য student টেবিলে আরো কিছু ডেটা রাখতে হবে।

```
INSERT INTO student (name, class, roll, section) VALUES
('Mosharraf Hossain', 9, 4, 'morning');
INSERT INTO student (name, class, roll, section) VALUES
('David Pandey', 9, 2, 'morning');
INSERT INTO student (name, class, roll, section) VALUES
('Promila Gosh', 8, 2, 'day');
INSERT INTO student (name, class, roll, section) VALUES
('Bazlur Rahman', 8, 1, 'day');
INSERT INTO student (name, class, roll, section) VALUES
('Sourav Das', 9, 1, 'day');
INSERT INTO student (name, class, roll, section) VALUES
('Tamanna Nishat', 10, 1, 'morning');
INSERT INTO student (name, class, roll, section) VALUES
('Maysha', 10, 1, 'day');
```

এরকম কিছু ডেটা টেবিলে রাখার পরে আবার SELECT কুয়েরি ব্যবহার করে দেখতে হবে যে টেবিলে ডেটা আছে কি না। টার্মিনালে অনেক সময় কোনো ফিল্ডের নাম বেশি বড় হলে পুরোটা নাও দেখাতে পারে।

টেবিল থেকে ডেটা পড়া বা দেখার জন্য SELECT কুয়েরিতে বিভিন্ন শর্তও জুড়ে দেওয়া যায়। সেজন্য WHERE লিখে তারপরে শর্ত লিখতে হবে। নিচে বিভিন্ন উদাহরণ দেখানো হলো।
কেবল নবম শ্রেণির শিক্ষার্থীদের তথ্য দেখতে হলে লিখতে হবে—

```
SELECT * FROM student WHERE class = 9;
```

এখানে শর্ত লেখা হয়েছে class = 9, অর্থাৎ class কলামের মান হতে হবে 9। SQL-এ দুটি মান তুলনা করার জন্য নিচের অপারেটরগুলো ব্যবহার করা হয়—

অপারেটর	বর্ণনা
=	সমান
<>	সমান নয়
>	বড় (বামপক্ষ ডানপক্ষের চেয়ে বড়)
>=	বড় কিংবা সমান (বামপক্ষ ডানপক্ষের চেয়ে বড় বা সমান)
<	ছোট (বামপক্ষ ডানপক্ষের চেয়ে ছোট)
<=	ছোট কিংবা সমান (বামপক্ষ ডানপক্ষের চেয়ে ছোট বা সমান)

আবার যেসব শিক্ষার্থী morning শাখার, তাদের তথ্য দেখতে হলে লিখতে হবে।

```
SELECT * FROM student WHERE section = 'morning';
```

নোট : টেক্সট টাইপের ডেটা নিয়ে কাজ করার সময় শুরুতে ও শেষে কোটেশন চিহ্ন (' চিহ্ন) ব্যবহার করতে হবে। অর্থাৎ নিচের মতো কুয়েরি লিখলে হবে না।

```
sqlite> SELECT * FROM student WHERE section = morning;
Error: no such column: morning
```

একাধিক শর্ত একসঙ্গে জুড়ে দিতে চাইলে AND অথবা OR ব্যবহার করে কাজটি করা যায়।
নবম শ্রেণি এবং morning শাখার শিক্ষার্থীদের তথ্য পেতে চাইলে লিখতে হবে–

```
SELECT * FROM student WHERE class = 9 AND section =
'morning';
```

নবম শ্রেণি অথবা morning শাখার শিক্ষার্থীদের তথ্য পেতে চাইলে লিখতে হবে–

```
SELECT * FROM student WHERE class = 9 OR section ='morning';
```

নবম শ্রেণিতে পড়ে না এবং morning শাখার শিক্ষার্থীদের তথ্য দেখার জন্য নিচের কুয়েরি লিখতে হবে–

```
SELECT * FROM student WHERE class <> 9 AND section =
'morning';
```

অষ্টম, নবম কিংবা দশম শ্রেণির শিক্ষার্থীদের তথ্য পাওয়ার জন্য কুয়েরি লিখতে হবে–

```
SELECT * FROM student WHERE class = 8 OR class = 9 OR class
= 10;
```

উপরের কুয়েরিটি অন্যভাবেও লেখা যায়–

```
SELECT * FROM student WHERE class IN (8, 9, 10);
```

ডেটা মুছে ফেলা ও পরিবর্তন করা

ধরা যাক, student টেবিলে একটি নতুন রেকর্ড যোগ করা হলো–

```
INSERT INTO student (name, class, roll, section) VALUES
('Fardeem Munir', 10, 1, 'day');
```

এখন টেবিলে day শাখার দশম শ্রেণির রোল নম্বর এক, এরকম দুইজন শিক্ষার্থী দেখা যাচ্ছে।

```
sqlite> SELECT name FROM student WHERE class = 10 AND roll =
1 AND section = 'day';
Maysha
Fardeem Munir
```

তাহলে নতুন যোগ করা রেকর্ডটি সঠিক নয়, কিংবা আগের রেকর্ডটি সঠিক নয়। নতুন রেকর্ডটি অর্থাৎ Fardeem Munir নামের শিক্ষার্থীর রেকর্ডটি মুছে ফেলতে হলে, DELETE কুয়েরি ব্যবহার করতে হবে। এটি লেখার নিয়ম হচ্ছে–

```
DELETE FROM [টেবিলের নাম] WHERE [শর্ত]
```

নিচের কুয়েরি চালালে Fardeem Munir-এর রেকর্ড মুছে যাবে–

```
DELETE FROM student WHERE name = 'Fardeem Munir';
```

তবে উপরের কুয়েরিটি চালালে student টেবিলে name কলামের যতগুলো রেকর্ড Fardeem Munir হবে, সব রেকর্ড মুছে যাবে। তাই কোনো নির্দিষ্ট রেকর্ড মুছে ফেলার জন্য একটু সতর্কতা অবলম্বন করতে হয়। যেমন, এই কুয়েরিতে নামের সঙ্গে আরো শর্ত জুড়ে দেওয়া যায়–

```
DELETE FROM student WHERE name = 'Fardeem Munir' AND class =
10 AND roll = 1 AND section = 'day';
```

অনেক সময় কোনো রেকর্ড পরিবর্তন বা হালনাগাদ করার প্রয়োজন হয়। এই কাজটি করা যায় UPDATE কুয়েরি ব্যবহার করে। এই কুয়েরি লেখার নিয়ম হচ্ছে–

```
UPDATE [টেবিলের নাম] SET [কলামের নাম] = নতুন ডেটা (একাধিক কলাম হলে কমা দিয়ে
তারপরে আবার [কলামের নাম] = নতুন ডেটা) WHERE [শর্ত]
```

ধরা যাক, ডেটাবেজে নিচের তথ্য রাখা হলো–

```
INSERT INTO student (name, class, roll, section) VALUES
('Fardeem Munir', 1, 1, 'day');
```

তারপর দেখা গেল, Fardeem Munir আসলে দশম শ্রেণির শিক্ষার্থী এবং তার রোল নম্বর 3। তখন রেকর্ডটি মুছে না ফেলে আপডেট করা যায়।

```
UPDATE student SET class = 10, roll = 3 WHERE name =
'Fardeem Munir';
```

একাধিক টেবিল জয়েন করা

রিলেশনাল ডেটাবেজে ডেটা বিভিন্ন টেবিলে রাখা হয় এবং প্রয়োজন হলে একটি কুয়েরিতে একাধিক টেবিল থেকে ডেটা পড়া যায়। এই বিষয়টিকে বলে জয়েন (join) করা, যা রিলেশনাল ডেটাবেজের একটি গুরুত্বপূর্ণ বৈশিষ্ট্য।

ধরা যাক, student_info ও result নামে দুটি টেবিল তৈরি করা হলো।

```
CREATE TABLE student_info (roll INTEGER, name TEXT;
CREATE TABLE result (roll INTEGER, subject TEXT, marks REAL;
```

এই টেবিল দুটির মধ্যে roll কলাম দিয়ে একটি রিলেশন দেখা যাচ্ছে। student_info টেবিলে প্রতিটি শিক্ষার্থীর roll ও name রয়েছে। আবার result টেবিলে, শিক্ষার্থীর রোল নম্বর ও বিভিন্ন বিষয়ে পরীক্ষায় প্রাপ্ত নম্বর (marks) রয়েছে। student_info টেবিলের সঙ্গে result টেবিলের রিলেশন হচ্ছে ওয়ান টু মেনি রিলেশন।

এখন টেবিলে কিছু ডেটা ইনসার্ট করা হবে—

```
INSERT INTO student_info (roll, name) VALUES (1, 'Mizanur
Rahman');
INSERT INTO student_info (roll, name) VALUES (10, 'Mosharraf
Hossain');
INSERT INTO student_info (roll, name) VALUES (2,  'Maysha');
INSERT INTO result (roll, subject, marks) VALUES (1,
'Bangla', 79.0);
INSERT INTO result (roll, subject, marks) VALUES (1,
'English', 76.0);
INSERT INTO result (roll, subject, marks) VALUES (1,
'Mathematics', 74.0);
INSERT INTO result (roll, subject, marks) VALUES (10,
'Bangla', 82.0);
```

```
INSERT INTO result (roll, subject, marks) VALUES (10,
'English', 70.0);
INSERT INTO result (roll, subject, marks) VALUES (10,
'Mathematics', 98.0);
INSERT INTO result (roll, subject, marks) VALUES (2,
'Bangla', 75.0);
INSERT INTO result (roll, subject, marks) VALUES (2,
'English', 80.0);
INSERT INTO result (roll, subject, marks) VALUES (2,
'Mathematics', 100.0);
```

রোল নম্বর 1 যে শিক্ষার্থীর, তার পরীক্ষার ফল জানতে নিচের কুয়েরিটি চালাতে হবে—

```
SELECT roll, subject, marks FROM result WHERE roll = 1;
sqlite> SELECT roll, subject, marks FROM result WHERE roll =
1;
roll           subject              marks
----------     ----------           ----------
1               Bangla              79.0
1               English             76.0
1               Mathematics         74.0
```

এখন, রোল নম্বরের পাশাপাশি শিক্ষার্থীর নামও যদি দেখানোর প্রয়োজন হয়, তখন student_info টেবিলেও কুয়েরি করতে হবে, কারণ result টেবিলে তো শিক্ষার্থীর নাম নেই। কুয়েরিটি হবে এমন—

```
sqlite> SELECT name, result.roll, subject, marks FROM
result, student_info WHERE result.roll = 1 AND result.roll =
student_info.roll;
name            roll           subject       marks
------------    ----------     ----------    ----------
Mizanur Rahman   1             Bangla         79.0
Mizanur Rahman   1             English        76.0
Mizanur Rahman   1             Mathematics    74.0
```

লক্ষ করতে হবে, কুয়েরিতে roll না লিখে result.roll লেখা হয়েছে, কারণ roll নামের কলাম দুটি টেবিলেই আছে। আর কুয়েরির result.roll = student_info.roll অংশ দিয়ে টেবিল দুটি যুক্ত (join) করা হয়েছে। সব শিক্ষার্থীর তথ্য চাইলে result.roll = 1 শর্তটি বাদ দিতে হবে। তখন কুয়েরিটি হবে এমন—

```
SELECT name, result.roll, subject, marks FROM result,
student_info WHERE result.roll = student_info.roll;
```

৬.৩.২ সর্টিং (Sorting)

সর্ট করা মানে হচ্ছে একটি নির্দিষ্ট ক্রমে সাজানো। এই ক্রমটি হতে পারে সংখ্যার ক্রম, নামের ক্রম বা অন্যকিছু। যেমন– আমরা শিক্ষার্থীদের তালিকা তৈরি করলে তাদেরকে সাজাতে পারি–নামের ক্রমানুসারে, শ্রেণির ক্রমানুসারে কিংবা রোল নম্বরের ক্রমানুসারে। আবার ছোট থেকে বড় ক্রমে যেমন সাজানো যায় (ইংরেজিতে একে বলা হয় Ascending Order), তেমনি বড় থেকে ছোট ক্রমেও সাজানো যায় (ইংরেজিতে একে বলে Descending Order)। সর্টিং করার জন্য কম্পিউটার বিজ্ঞানে বিভিন্ন ধরনের অ্যালগরিদম বা পদ্ধতি রয়েছে। কোনোটি সহজ, কোনোটি জটিল, কোনোটি বেশ দ্রুত গতির, আবার কোনোটি ধীর গতির। তবে ডেটাবেজে সর্টিং ব্যবহার করার সময় ডেটাবেজ সফটওয়্যার আসলে কীভাবে সর্টিংয়ের কাজটি করবে, তা নিয়ে মাথা ঘামাতে হয় না, বরং কীসের ভিত্তিতে সাজাতে হবে, আর কোন ক্রমে (ছোট থেকে বড়, নাকি বড় থেকে ছোট) সেটি বলে দিলেই হয়। সিলেক্ট কুয়েরির শেষে ORDER BY লিখে তারপরে কলামের নাম লিখলে সেই কলামের ডেটা অনুসারে ছোট থেকে বড় ক্রমে ডেটা আসবে। আর উল্টো ক্রমে (অর্থাৎ, বড় থেকে ছোট) ডেটা পেতে চাইলে শেষে DESC লিখতে হবে (Descending শব্দের প্রথম চারটি অক্ষর)।

student টেবিলে ডেটাগুলো বিভিন্নভাবে সাজানো যায়। নিচে কিছু উদাহরণ দেখানো হলো। শিক্ষার্থীদের প্রতিটি কুয়েরি নিজে নিজে এসকিউলাইটে চালিয়ে দেখার পরামর্শ দেওয়া হলো।

শিক্ষার্থীদের শ্রেণি অনুসারে ছোট থেকে বড় ক্রমে সাজাতে চাইলে–

```
SELECT * FROM student ORDER BY class;
```

শিক্ষার্থীদের শ্রেণি অনুসারে বড় থেকে ছোট ক্রমে সাজাতে চাইলে

```
SELECT * FROM student ORDER BY class DESC;
```

শিক্ষার্থীদের শ্রেণি অনুসারে বড় থেকে ছোট ক্রমে এবং একই শ্রেণির শিক্ষার্থীদের রোল নম্বর অনুযায়ী ছোট থেকে বড় ক্রমে সাজাতে চাইলে–

```
SELECT * FROM student ORDER BY class DESC, roll;
```

শিক্ষার্থীদের শাখা অনুসারে ছোট থেকে বড় ক্রমে (day, morning-এর চাইতে ছোট, কারণ ইংরেজিতে d অক্ষরটি m-এর আগে আসে), একই শাখার শিক্ষার্থীদের শ্রেণি অনুসারে বড় থেকে ছোট, আর সবশেষে একই শ্রেণির শিক্ষার্থীদের রোল নম্বর অনুসারে ছোট থেকে বড় ক্রমে সাজাতে চাইলে–

```
SELECT * FROM student ORDER BY section, class DESC, roll;
```

নিজে কর : নিচের কুয়েরিটি চালালে কোন ক্রমে ডেটা পাওয়া যাবে?

```
SELECT * FROM student ORDER BY class DESC, section, roll;
```

৬.৩.৩ ইনডেক্সিং (Indexing)

এসকিউএল একটি ডিক্লারেটিভ প্রোগ্রামিং ভাষা, এ কথা পূর্বেই বলা হয়েছে। এখানে কী করতে হবে, এটিই গুরুত্বপূর্ণ। কীভাবে করতে হবে, সেটি ডেটাবেজ সফটওয়্যারের দায়িত্ব। এখন, কোনো টেবিলে যখন অনেক অনেক বেশি ডেটা থাকে (যেমন– পাঁচ লক্ষ, দশ লক্ষ কিংবা আরো বেশি), তখন বিভিন্ন কুয়েরির গতি অনেক কমে যায়, অর্থাৎ কুয়েরি চলতে অনেক বেশি সময় লাগে। যেমন– একটি টেবিলে যদি এ বছরের মাধ্যমিক পরীক্ষায় ঢাকা বোর্ডের সব শিক্ষার্থীর ডেটা থাকে, তাহলে সেই টেবিলে পাঁচ থেকে দশ লক্ষের মতো রেকর্ড বা রো থাকবে। এখন যদি সেখান থেকে নাম কিংবা রোল নম্বর (বা রেজিস্ট্রেশন নম্বর) দিয়ে একজন শিক্ষার্থীর তথ্য বের করার চেষ্টা করা হয়, তাহলে সেই কুয়েরি চলতে বেশ সময় লাগবে। কারণ তখন লিনিয়ার সার্চের মাধ্যমে এক এক করে সবার তথ্য পরীক্ষা করা হবে এবং যাদের সঙ্গে মিল পাওয়া যাবে, তাদের তথ্য দেখানো হবে। লিনিয়ার সার্চ কীভাবে কাজ করে সেটি আগের অধ্যায়ে আলোচনা করা হয়েছে। টেবিলে যত বেশি রেকর্ড থাকবে, তত বেশি সময় লাগবে।

ইনডেক্সিং হচ্ছে একটি বিশেষ পদ্ধতি, যার মাধ্যমে ডেটা সহজে ও দ্রুত খুঁজে পাওয়া যায়। যেমন– ডিকশনারি বা অভিধান ব্যবহার করার সময় কোনো শব্দ খুঁজতে কিন্তু বেশি সময় লাগে না। কারণ শব্দগুলো একটি ক্রমে সাজানো থাকে, এবং অভিধানের কোনো পাতা খুললে কাঙ্ক্ষিত শব্দটি ওই পাতায়, নাকি তার আগে না পরে আছে, সেটি সহজেই বোঝা যায়। ডেটা যদি সর্ট করা থাকে তাহলে বাইনারি সার্চ ব্যবহার করে খুব দ্রুতে খুঁজে পাওয়া যায়। তেমনি কোনো বিশেষ কলামের উপর ইনডেক্স তৈরি করলে সেই কলামের মান দিয়ে ডেটা খুঁজলে ডেটাবেজ সফটওয়্যার খুব দ্রুত সেটি বের করে দিতে পারে। ডেটাবেজ কীভাবে ইনডেক্স তৈরি করার কাজটি করবে, সেটি ব্যবহারকারীর জানতে হয় না, কেবল কোন কলামের উপর ইনডেক্স তৈরি করতে হবে, সেটি বলে দিতে হয়। প্রয়োজন হলে একাধিক কলামের উপরও ইনডেক্স তৈরি করা যায়।

ইনডেক্স তৈরির কুয়েরি লেখার নিয়ম হচ্ছে–

```
CREATE INDEX [ইনডেক্সের নাম] ON [টেবিলের নাম] ([কলামের নাম], একাধিক কলামের ক্ষেত্রে নামগুলো কমা দিয়ে পৃথক করা থাকতে হবে);
```

যেমন– class কলামের উপর ইনডেক্স তৈরি করতে চাইলে নিচের কুয়েরি লিখতে হবে–

```
CREATE INDEX student_class_idx ON student (class);
```

ইনডেক্স তৈরি করলে কলামের উপর নির্ভর করে ডেটা খুঁজে পেতে যেমন সহজ হয়, তেমনি যেই কলামের উপর ইনডেক্স করা হয়েছে, সেটি অনুসারে সর্ট করলেও সর্ট করার কাজটি দ্রুত হয়। উপরের ইনডেক্স তৈরি করার ফলে SELECT * FROM student ORDER BY class; কুয়েরিটি অনেক দ্রুত কাজ করবে।

আবার অনেক সময় অনন্য (unique) ইনডেক্স তৈরি করা যায়, যেন একই ডেটা ডেটাবেজে একাধিকবার না ঢোকে। যেমন– নিচের ইনডেক্স তৈরি করা হলে student টেবিলে একই শাখার একই ক্লাসের একই রোল নম্বরের কেবল একজন শিক্ষার্থী থাকবে–

```
CREATE UNIQUE INDEX unique_student_idx ON student (section,
class, roll);
```

কাজটি টেবিল তৈরি করার পরে যে কোনো সময় করা যায়। এমনকি টেবিলে ডেটা রাখার পরেও করা যায়। আবার টেবিল তৈরি করার সময়ও এক বা একাধিক ফিল্ডকে ইউনিক ঘোষণা করা যায়।

দশম শ্রেণির day শাখার রোল নম্বর 1 ইতোমধ্যে student টেবিলে আছে। এখন যদি এরকম আরেকটি ডেটা রাখার চেষ্টা করা হয়, তাহলে ডেটাবেজ সেটি হতে দেবে না। ইতিপূর্বে একটি উদাহরণে দেখানো হয়েছিল যে একই ক্লাসের একই রোল নম্বরে দুজন শিক্ষার্থীর ডেটা টেবিলে ঢুকে গিয়েছে। ইউনিক ইনডেক্স ব্যবহার করলে বিষয়টি এড়ানো যায়।

```
sqlite> INSERT INTO student (name, class, roll, section)
VALUES ('Maysha', 10, 1, 'day');
Error: UNIQUE constraint failed: student.section,
student.class, student.roll
```

উল্লেখ্য যে, কোনো টেবিলে কোনো কলামকে প্রাইমারি কি হিসেবে ঘোষণা করলে ডেটাবেজ আপনা-আপনি ওই কলামের উপর ইনডেক্স তৈরি করে নেয়, আলাদাভাবে ইনডেক্স তৈরি করার প্রয়োজন হয় না।

ডেটাবেজে ইনডেক্স তৈরির বিভিন্ন সুবিধার পাশাপাশি কিছু অসুবিধাও আছে। প্রথমত, ইনডেক্স তৈরির পরে সেই টেবিলে কোনো নতুন ডেটা যোগ করা, মুছে ফেলা বা পরিবর্তন করা হলে ইনডেক্স যেখানে তৈরি করা হয়েছে, সেখানেও পরিবর্তন হয়। কাজটি ডেটাবেজ নিজেই করে, তবে এর ফলে INSERT, UPDATE, DELETE কুয়েরি চলতে আগের চেয়ে বেশি সময় লাগে। এছাড়া ইনডেক্স তৈরি করতে অতিরিক্ত জায়গার প্রয়োজন হয়, অর্থাৎ হার্ড ডিস্কের অতিরিক্ত জায়গা খরচ হয়।

কোনো ইনডেক্স মুছে ফেলতে হলে DROP INDEX লিখে তারপরে ইনডেক্সের নাম লিখতে হবে। যেমন–

```
DROP INDEX student_class_idx;
```

৬.৪ ডেটা সিকিউরিটি (Data Security)

ডেটাবেজের নিরাপত্তা খুবই গুরুত্বপূর্ণ বিষয়। কারণ ডেটাবেজে ব্যক্তিগত কিংবা গোপনীয় তথ্য থাকতে পারে, ব্যবসায়িক তথ্য থাকতে পারে কিংবা সরকারি গুরুত্বপূর্ণ তথ্যও থাকতে পারে। ডেটাবেজের নিরাপত্তার বিষয়টি একাধিক দৃষ্টিকোণ থেকে দেখা হয়।

প্রথমত, ডেটার নিরাপত্তা দিতে হবে যেন ডেটা হারিয়ে না যায়, বা ডেটা লস (data loss) না ঘটে। এ জন্য নিয়মিত ডেটার ব্যাকআপ নিতে হয়, অর্থাৎ ডেটার কপি তৈরি করা হয়। ডেটার কপি তৈরি করে একই হার্ড ডিস্কে রাখলে কোনো কারণে আসল ডেটাতে কোনো সমস্যা হলে (যাকে ডেটা করাপশন– data corruption বলা হয়) ব্যাকআপ থেকে সেই ডেটা পুনরুদ্ধার করা যায়। কিন্তু সম্পূর্ণ হার্ড ডিস্ক ক্র্যাশ করতে পারে বা নষ্ট হয়ে যেতে পারে সেই সম্ভাবনার কথা বিবেচনা করে পৃথক হার্ড ডিস্কে ডেটা ব্যাকআপ রাখা হয়। আবার ডেটার গুরুত্ব বিবেচনা করে, আলাদা ডেটা সেন্টারেও ডেটার ব্যাকআপ রাখা হয়, যেন কোনোরকম দুর্যোগ, যেমন– অগ্নিকান্ড, ভূমিকম্প ইত্যাদি ঘটলেও ডেটা পুনরুদ্ধার করা যায়। তাই ডেটা সেন্টারগুলো আলাদা শহরে হয়।

আবার অনাকাঙ্ক্ষিত ব্যক্তি বা সিস্টেম যেন ডেটা দেখতে কিংবা ডেটা পরিবর্তন করতে না পারে, ডেটাবেজে সেই ব্যবস্থাও থাকে। এটিও ডেটার নিরাপত্তার গুরুত্বপূর্ণ দিক। যেমন–পাসওয়ার্ড ছাড়া কেউ ডেটাবেজ সিস্টেমে ঢুকতে পারবে না। আবার কোনো কোনো ব্যবহারকারী কেবল ডেটাবেজের নির্দিষ্ট কিছু টেবিল নিয়ে কাজ করতে পারবে। আবার কোনো কোনো ব্যবহারকারী কেবল ডেটা দেখতে পারবে (SELECT), কিন্তু পরিবর্তন (INSERT, UPDATE, DELETE) করতে পারবে না। এজন্য বিভিন্ন রকমের পারমিশন (permission) ঠিক করে দেওয়া যায়। এসকিউলাইট ডেটাবেজে এই বৈশিষ্ট্য না থাকলেও ওরাকল, পোস্টগ্রেস, মাইসিক্যুয়েল, এসকিউএল সার্ভার ইত্যাদি ডেটাবেজে এ ধরনের নিরাপত্তার ব্যবস্থা রয়েছে।

৬.৪.১ ডেটা এনক্রিপশন (Data encryption)

হার্ড ডিস্কে যখন ডেটা সংরক্ষণ করা হয়, কিংবা নেটওয়ার্কের মাধ্যমে ডেটা আদান-প্রদান করা হয়, তখন সেই ডেটার গোপনীয়তা রক্ষা করতে হলে ডেটা এনক্রিপ্ট (encrypt) করতে হয়। তা না হলে অনাকাঙ্ক্ষিত ব্যক্তি কিংবা সিস্টেম সেই ডেটা পড়ে ফেলতে পারে। ডেটা এনক্রিপ্ট করার ধারণা কিন্তু নতুন নয়, বা এটা যে কেবল কম্পিউটারের সঙ্গে সম্পর্কিত, এমনটি নয়। হাজার বছর আগেও মানুষ ডেটা এনক্রিপ্ট করত, যেন যাকে ডেটা পাঠানো হচ্ছে সে ছাড়া অন্য কেউ সেই ডেটার মর্মোদ্ধার করতে না পারে। রোমান সম্রাট জুলিয়াস সিজার একটি পদ্ধতিতে তার চিঠিপত্র লিখতেন, যেটি এনক্রিপ্ট করা থাকত এবং যার কাছে চিঠি যাবে, সেই কেবল ডিক্রিপ্ট (decrypt) করতে পারত বা চিঠির অর্থ উদ্ধার করতে পারত। আবার প্রথম ও দ্বিতীয় বিশ্বযুদ্ধের সময় তো অনেক গণিতবিদ এই এনক্রিপশন পদ্ধতি নিয়ে কাজ করেছেন, যেন তারা শত্রুপক্ষের নিজেদের মধ্যে পাঠানো বার্তার মর্মোদ্ধার করতে পারেন, সেই সঙ্গে মিত্রপক্ষের মধ্যে নিরাপদে ডেটা এনক্রিপ্ট করে পাঠাতে পারেন। কম্পিউটার বিজ্ঞানের যেই শাখায় ডেটা এনক্রিপশন নিয়ে গবেষণা ও কাজ করা হয়, তাকে বলা হয় ক্রিপ্টোগ্রাফি (cryptography)।

এনক্রিপশন পদ্ধতির মূলনীতি হচ্ছে মূল ডেটাকে প্রথমে এনক্রিপ্ট করা। যে ডেটা পাঠাবে, এটি তার কাজ। মূল ডেটাকে বলা হয় প্লেইন টেক্সট (plane text) আর এনক্রিপ্ট করার পরে সেই ডেটাকে বলে সাইফার টেক্সট (cipher text)। তারপর আরেকটি সিস্টেমের কাজ হচ্ছে সাইফার টেক্সট থেকে মূল ডেটা উদ্ধার করা। ডেটা এনক্রিপশন পদ্ধতি মূলত দুই ধরনের হয়–

১. সিমেট্রিক কি ক্রিপ্টোগ্রাফি (symmetric key cryptography)
২. অ্যাসিমেট্রিক কি ক্রিপ্টোগ্রাফি (asymmetric key cryptography)

সিমেট্রিক কি ক্রিপ্টোগ্রাফি

এই পদ্ধতিতে একটি বিশেষ কি (key) ব্যবহার করে ডেটা এনক্রিপ্ট করা হয় এবং প্রেরক ও গ্রাহক উভয়পক্ষের কাছেই এই কি (key) থাকতে হয়। প্রেরক এই কি (key) ব্যবহার করে ডেটা এনক্রিপ্ট করে এবং গ্রাহক এই কি (key) ব্যবহার করে ডেটা ডিক্রিপ্ট করে।

চিত্র 6.2 : সিমেট্রিক কি ক্রিপ্টোগ্রাফি

এই পদ্ধতিটি বেশ কার্যকর হলেও যখন দুটি আলাদা পক্ষের মধ্যে ডেটা আদান-প্রদান করা হয়, তখন দুটি বিশেষ কারণে অসুবিধা হয়। প্রথমত, যেই কি (key) ব্যবহার করা হয়, সেই কি যেন অন্য কেউ জানতে না পারে, সেটি নিশ্চিত করতে হয়। এটি আপাতদৃষ্টিতে সহজ মনে হলেও, আসলে অত্যন্ত কঠিন কাজ। দ্বিতীয়ত, একপক্ষ যদি অনেকের সঙ্গে ডেটা আদান-প্রদান করে, সেই ক্ষেত্রে প্রতিটি পক্ষের জন্যই আলাদা কি (key) ব্যবহার করতে হয়। এখন, ধরা যাক, একটি ই-কমার্স সাইটে দশ লক্ষ গ্রাহক, তাদের প্রত্যেকের সঙ্গে ডেটা এনক্রিপ্ট করার জন্য পৃথক কি ব্যবহার করা বাস্তবসম্মত নয়।

অ্যাসিমেট্রিক কি ক্রিপ্টোগ্রাফি

প্রত্যেক সিস্টেম একটি বিশেষ অ্যালগরিদম ব্যবহার করে একজোড়া কি তৈরি করে, যাদের একটি হচ্ছে পাবলিক কি ও অপরটি হচ্ছে প্রাইভেট কি। এখন প্রত্যেক সিস্টেম তার পাবলিক কি সবাইকে জানিয়ে দেয়।

চিত্র 6.3 : অ্যাসিমেট্রিক কি ক্রিপ্টোগ্রাফি

তাহলে A-এর কাছে B ও C-এর পাবলিক কি আছে। এখন A যদি B-কে কোনো ডেটা পাঠাতে চায়, তাহলে B-এর পাবলিক কি দিয়ে সেই ডেটা এনক্রিপ্ট করে পাঠায়। এই ডেটা ডিক্রিপ্ট করতে হলে B-এর প্রাইভেট কি ব্যবহার করতে হবে, তাই অন্য কেউ এই ডেটা ডিক্রিপ্ট করতে পারবে না। তেমনি C-এর কাছে ডেটা পাঠাতে হলে C-এর পাবলিক কি ব্যবহার করে ডেটা পাঠাতে হবে যা কেবল C-এর পক্ষেই ডিক্রিপ্ট করা সম্ভব। C-এর পাবলিক কি ব্যবহার না করে এনক্রিপ্ট করা হলে সেই ডেটা C-এর পক্ষে ডিক্রিপ্ট করা সম্ভব নয়। অনুরূপভাবে, A-এর কাছে ডেটা পাঠাতে হলে A-এর পাবলিক কি ব্যবহার করে এনক্রিপ্ট করে ডেটা পাঠাতে হবে, যা A তার প্রাইভেট কি ব্যবহার করে ডিক্রিপ্ট করতে পারবে।

৬.৪.২ RDBMS -এর বৈশিষ্ট্য

রিলেশনাল ডেটাবেজের ধারণা প্রবর্তন করেন এডগার ফ্র্যাঙ্ক কড (Edgar Frank Codd)। সেই সময় তিনি ১২টি বৈশিষ্ট্যের কথা উল্লেখ করেন, যেগুলো রিলেশনাল ডেটাবেজ সিস্টেমে থাকতে হবে। বিভিন্ন ডেটাবেজ নির্মাতা প্রতিষ্ঠান তাদের নিজেদের মতো ডেটাবেজ তৈরির সময় বৈশিষ্ট্যগুলো মেনে চলার চেষ্টা করে। রিলেশনাল ডেটাবেজের কিছু সাধারণ বৈশিষ্ট্য হচ্ছে–

- একটি রিলেশনাল ডেটাবেজ সিস্টেম কেবল বিভিন্ন টেবিল ও তাদের মধ্যকার সম্পর্ক ব্যবহার করেই সব ধরনের কাজ করতে পারবে। ডেটাবেজের সমস্ত ডেটা টেবিলে সংরক্ষিত হবে। যে কোনো ডেটাই কোনো একটি টেবিলের একটি ঘরের (নির্দিষ্ট রো ও কলামে) মান হিসেবে প্রকাশিত হবে।
- ডেটাবেজের যে কোনো ডেটা সুনির্দিষ্টভাবে টেবিলের নাম, প্রাইমারি কি (কিংবা রো-এর মান) এবং ক্ষেত্রবিশেষে অন্যান্য কলামের মান ব্যবহার করে পাওয়া যাবে। উদাহরণস্বরূপ, দশম শ্রেণির রোল নম্বর 1 যেই শিক্ষার্থীর, তার নাম পেতে হলে কুয়েরি লিখতে হয়,

```
SELECT name FROM student WHERE roll = 1 AND class = 10;
```

- ডেটাবেজে এক বা একাধিক রো ইনসার্ট, আপডেট ও ডিলিট করার ব্যবস্থা থাকতে হবে। যেমন– নবম শ্রেণির সব শিক্ষার্থীকে দশম শ্রেণিতে নিতে চাইলে, এরকম কুয়েরি লেখা যায়–

```
UPDATE student SET class = 10 WHERE class = 9;
```

  এতে student টেবিলের সব শিক্ষার্থী যাদের class এর মান 9, তাদের ক্ষেত্রে সেই মানটি 10 হয়ে যাবে।
- ডেটাবেজের অভ্যন্তরীণ কোনো পরিবর্তন হলে সেটি ডেটাবেজ যারা ব্যবহার করে, তাদের উপর কোনো প্রভাব ফেলবে না। যেমন ডেটা ডিস্কে যেই ফরম্যাটে সংরক্ষিত হয়, সেই ফরম্যাট হয়তো পরিবর্তন করা হতে পারে, কিন্তু যেসব ব্যবহারকারী ওই ডেটাবেজ ব্যবহার করবে, এটি তাদের জানতে হবে না, কিংবা এ নিয়ে মাথা ঘামাতে হবে না। তারা আগের মতোই ডেটা অ্যাকসেস করতে পারবে।

- ডেটাবেজ প্রদত্ত ইন্টারফেস ব্যবহার করে বিভিন্ন সফটওয়্যার অ্যাপ্লিকেশন ডেটাবেজ ব্যবহার করতে পারবে। ইন্টারফেস পরিবর্তন না করে ডেটাবেজে প্রয়োজন হলে অভ্যন্তরীণ পরিবর্তন করা যাবে।
- ডেটাবেজের ডেটা যদি একাধিক ডিস্কে কিংবা একাধিক কম্পিউটারে সংরক্ষণ করা হয়, সেটি নিয়ে ব্যবহারকারীর মাথা ঘামাতে হবে না। ব্যবহারকারীর কাছে মনে হবে ডেটাবেজ একটি জায়গাতেই ডেটা সংরক্ষণ করছে।

৬.৪.৩ RDBMS -এর ব্যবহার

RDBMS-এর ব্যবহার অনেক ব্যাপক ও বিস্তৃত। যদিও বাংলাদেশে এখন সব পর্যায়ে তথ্যপ্রযুক্তির ছোঁয়া লাগেনি, তাই এখানে অনেক কিছু করার আছে।

সরকারি-বেসরকারি প্রতিষ্ঠানের বিভিন্ন ধরনের তথ্য সংরক্ষণ করার জন্য ডেটাবেজ ব্যবহার করা হয়। যেমন—জাতীয় পরিচয়পত্রে নাগরিকদের যে তথ্য থাকে, সেগুলো ডেটাবেজে সংরক্ষণ করা হয়। তারপরে পাসপোর্ট, ড্রাইভিং লাইসেন্স, চিকিৎসা, কৃষি, জমিজমার হিসেব ইত্যাদি নানান তথ্য ডেটাবেজে সংরক্ষণ করা হয়।

ই-কমার্স ওয়েবসাইটে বিভিন্ন ধরনের পণ্য ক্রয় করার ব্যবস্থা থাকে। এক্ষেত্রে পণ্যের তথ্য রাখা, গ্রাহকদের তথ্য রাখা, গ্রাহকদের পণ্য সরবরাহ ব্যবস্থা—এই পুরো প্রক্রিয়াটি পরিচালনা করার জন্য যে সফটওয়্যার ব্যবহার করা হয়, তার মূলে রয়েছে ডেটাবেজ। ব্যাংক, বিমা ও বিভিন্ন আর্থিক প্রতিষ্ঠানেও রিলেশনাল ডেটাবেজ ব্যবহার করা হয়। এক্ষেত্রে গ্রাহকদের তথ্য ব্যবস্থাপনা, লেনদেন ইত্যাদি পরিচালনা করার জন্য ডেটাবেজের প্রয়োজন হয়। শিক্ষাপ্রতিষ্ঠানে শিক্ষার্থীদের তথ্য, শিক্ষকদের তথ্য, শিক্ষার্থী ভর্তি, তাদের হাজিরার তথ্য, পরীক্ষার ফলাফল ইত্যাদি ব্যবস্থাপনা করার জন্য ডেটাবেজ ব্যবহার করা হয়।

৬.৪.৪ কর্পোরেট ডেটাবেজ (Corporate Database)

বড় বড় প্রতিষ্ঠান তথা কর্পোরেশন (corporation)-এ অনেক ধরনের ডেটা নিয়ে কাজ করতে হয়। এর মধ্যে অনেক কাজই আবার পরস্পরের উপর নির্ভরশীল—একটি না ঘটলে অন্যটি ঘটানো যায় না। যেমন—কোনো পণ্য যদি স্টকে না থাকে, তাহলে সেটি বিক্রি করা যায় না। এখন এই কর্পোরেশন পরিচালনা করার জন্য এক ধরনের সফটওয়্যার রয়েছে, যেগুলোকে বলা হয় ইআরপি (ERP : Enterprise Resource Planner)। ইআরপি সফটওয়্যারের বিভিন্ন মডিউল থাকে, বিভিন্ন প্রতিষ্ঠান তাদের প্রয়োজনমতো বিভিন্ন মডিউল ব্যবহার করে। কিছু সাধারণ মডিউল হচ্ছে, অ্যাকাউন্টস (accounts —সব ধরনের হিসাব-নিকাশের জন্য), ইনভেন্টরি (inventory—পণ্যের মজুদ ব্যবস্থাপনা), পে-রোল (payroll—কর্মচারীদের বেতন-ভাতা সংক্রান্ত হিসাব-নিকাশ), কাস্টমার রিলেশনশিপ ম্যানেজমেন্ট (customer relationship management) ইত্যাদি। এই সবকিছুর মূলেই রয়েছে ডেটা আর তাই ডেটার সঠিক ব্যবস্থাপনা অত্যন্ত গুরুত্বপূর্ণ। আবার বিভিন্ন বড় বড় প্রতিষ্ঠানের অফিস একাধিক শহরে, এমনকি একাধিক দেশেও থাকতে পারে। সব অফিসের সব তথ্য একই সিস্টেমের আওতায় আনা কর্পোরেট ডেটাবেজের প্রধান চ্যালেঞ্জ। সেই সঙ্গে সেসব ডেটার নিরপত্তা নিশ্চিত করাও একটি গুরুত্বপূর্ণ বিষয়।

৬.৪.৫ সরকারি প্রতিষ্ঠানে ডেটাবেজ (Database in Government Organization)

সরকারি প্রতিষ্ঠানে ডেটাবেজের ব্যবহার অত্যন্ত গুরুত্বপূর্ণ। সরকারি বিভিন্ন প্রতিষ্ঠান দেশের নাগরিকদের বিভিন্ন তথ্য নিয়ে কাজ করে। কিন্তু ডেটাবেজ ব্যবহার না করলে কিংবা সঠিকভাবে ব্যবহার না করলে সেসব প্রতিষ্ঠানের মধ্যে কোনো সমন্বয় থাকে না। এ কারণে নাগরিকদের যথেষ্ট ভোগান্তি পোহাতে হয়, সরকারি প্রতিষ্ঠানগুলোতেও লাখ লাখ কর্মঘণ্টা নষ্ট হয়। উদাহরণস্বরূপ, বাংলাদেশের সব নাগরিকদের তথ্য ও আঙ্গুলের ছাপ জাতীয় পরিচয়পত্র তৈরির সময় সংগ্রহ করা হয়। সরকারি কোনো একটি বিশেষ প্রতিষ্ঠান সেই তথ্য সংরক্ষণ ও ব্যবস্থাপনার কাজ করে। এখন পাসপোর্ট তৈরির সময়, সবাইকে আবার সব তথ্য পূরণ করতে হয়। দুটি প্রতিষ্ঠানের সফটওয়্যারের মধ্যে সমন্বয় করে ডেটাবেজের সঠিক ব্যবহার করলে এই কাজটি সহজেই এড়ানো যায়।

আরেকটি উদাহরণ দেওয়া যাক। এইচএসসি পরীক্ষা পাশ করার পরে অনেকেই বিভিন্ন বিশ্ববিদ্যালয়ে ভর্তি পরীক্ষা দিতে যায়। সেখানে তাকে রেজিস্ট্রেশন ফর্ম পূরণ করা সহ নানা ঝামেলার মধ্যে দিয়ে যেতে হয়। শিক্ষাবোর্ডের কাছে কিন্তু একজন শিক্ষার্থীর তথ্য ও তার মাধ্যমিক ও উচ্চ মাধ্যমিক পরীক্ষার ফলাফল ডেটাবেজে সংরক্ষণ করা আছে। এই ডেটাবেজের সঠিক ব্যবহার করলে আর আলাদা রেজিস্ট্রেশন ফর্মে একই তথ্য দেওয়ার কোনো প্রয়োজন নেই। ইতোমধ্যে কয়েকটি বিশ্ববিদ্যালয় এই কাজটি করে তাদের ভর্তি প্রক্রিয়া সহজ করেছে। এরকম শত শত সরকারি প্রতিষ্ঠানে অনেক কাজ হয়, যেখানে ডেটাবেজ ব্যবহার ও সঠিকভাবে সমন্বয় করলে অনেক কাজ অনেক কম সময়ে ও কম ঝামেলা করে সম্পন্ন করা যায়।

সরকারি প্রতিষ্ঠানে ডেটাবেজের আরেকটি ভালো ব্যবহার হতে পারে ডেটা-ভিত্তিক সিদ্ধান্ত গ্রহণ প্রক্রিয়া চালুর মাধ্যমে। ঠিকভাবে বিভিন্ন রকম ডেটা সংরক্ষণ করলে, সেই ডেটা ব্যবহার করে ভবিষ্যতে কোন কাজটি কখন করতে হবে, সেই সিদ্ধান্ত নেওয়া সহজ হয়ে যায়। শিক্ষা, স্বাস্থ্য, কৃষি—এসব ক্ষেত্রে বিগত বছরের ডেটা ব্যবহার ও বিশ্লেষণ করে অনেক তথ্য বের করা সম্ভব, যা পরবর্তী বছরের করণীয় নির্ধারণ করতে সহায়তা করে। যেমন— বিভিন্ন সরকারি-বেসরকারি হাসপাতালের রোগীদের তথ্য যদি একটি কেন্দ্রীয় ডেটাবেজে থাকে, তাহলে কোন সময়ে, কোন অঞ্চলে কোন রোগের প্রকোপ বেশি হয়, তা সহজেই নির্ণয় করা সম্ভব। সেক্ষেত্রে আগে থেকেই প্রতিরোধের ব্যবস্থা গ্রহণ, প্রয়োজনীয় ওষুধের সরবরাহ নিশ্চিতকরণ ইত্যাদি কাজ করে ফেলা সম্ভব।

সরকারি প্রতিষ্ঠানে ডেটাবেজ ব্যবহারের মূল চ্যালেঞ্জগুলো হচ্ছে ডেটার নিরাপত্তা নিশ্চিত করা, বিপুল পরিমাণ ডেটার ব্যবস্থাপনার জন্য দক্ষ লোকের সরবরাহ নিশ্চিত করা, বিভিন্ন প্রতিষ্ঠান যেন একই ডেটা আলাদাভাবে ব্যবহার না করে (বরং নিজেদের মধ্যে ডেটা আদান-প্রদান করতে পারে), সেটির ব্যবস্থা করা ইত্যাদি।

অনুশীলনী

বহুনির্বাচনি প্রশ্ন

১. মূল ডেটাকে অন্য ফরমেটে পরিবর্তনের পদ্ধতি কোনটি?

ক. ম্যানিপুলেশন খ. ভ্যালিডেশন
গ. এনক্রিপশন ঘ. ডিক্রিপশন

২. সর্টিংয়ের জন্য ব্যবহৃত ফিল্ডের ডেটা টাইপ -

i. Text
ii. Currency
iii. OLE Objects

নিচের কোনটি সঠিক?

ক. i ও ii খ. i ও iii
গ. ii ও iii ঘ. i, ii ও iii

নিচের উদ্দীপকটি পড় এবং ৩ ও ৪ নং প্রশ্নের উত্তর দাও-

একটি কলেজের অধ্যক্ষ প্রতিষ্ঠানের সব ধরনের তথ্য ডেটাবেসের মাধ্যমে সংরক্ষণের সিদ্ধান্ত নেন। সিদ্ধান্তটি বাস্তবায়নের পর ফলাফলের ভিত্তিতে দুর্বল শিক্ষার্থীদের তালিকা আলাদাভাবে প্রদর্শণের ব্যবস্থা নিলেন।

৩. তালিকা প্রদর্শনের পদ্ধতি কোনটি?

ক. সর্টিং খ. ইনডেক্সিং
গ. কুয়েরি ঘ. এনক্রিপশন

৪. অধ্যক্ষের সিদ্ধান্ত বাস্তবায়নের ফলে-

i. তথ্যসমূহের সব ধরনের নিরাপত্তা দেয়া যাবে
ii. তথ্যসমূহের যেকোনো ধরনের বিন্যাস সম্ভব হবে
iii. অতিদ্রুত শিক্ষার্থীদের ডেটা উপস্থাপন করা যাবে

নিচের কোনটি সঠিক?

ক. i ও ii খ. i ও iii
গ. ii ও iii ঘ. i, ii ও iii

৫.

Roll	Name	GPA
01	X	5.00
02	Y	4.50
03	Z	5.00

উদ্দীপকের টেবিল হতে যাদের GPA = 5.00 তাদের নাম দেখতে SQL কমান্ড "SELECT NAME FROM Student" এর পরের অংশ কোনটি?

ক. WHERE "GPA", = "5.00"; খ. WHERE "GPA", "5.00";
গ. WHERE GPA = "5.00"; ঘ. WHERE "GPA", = "5.00"

৬. Primary Key এর সাথে Foreign Key এর রিলেশন কীরূপ?

i. one to one ii. one to many
iii. many to many

নিচের কোনটি সঠিক?

ক. i ও ii খ. i ও iii

গ. ii ও iii ঘ. i, ii ও iii

৭. ডেটাবেস ম্যানেজমেন্ট সিস্টেম (DBMS) এর প্রধান কাজ হচ্ছে-

i. ডেটাবেস তৈরি করা

ii. ডেটা এন্ট্রি ও সংরক্ষণ করা

iii. রিপোর্ট তৈরি ও প্রিন্ট করা

নিচের কোনটি সঠিক?

ক. i ও ii খ. i ও iii

গ. ii ও iii ঘ. i, ii ও iii

সৃজনশীল প্রশ্ন

১.

TID	T NAME	Subject
101	Mr. Rayhan	English
102	Mr. Kaiser	ICT
10.	Mr. Yaqub	Biology

Teacher's table

TID	Group	Time
101	Science	10:00
101	Humanities	10:45
102	Science	10:45
102	B. Studies	10:00
103	Science	11:30

Routine table

ক. ডেটাবেস কী?

খ. কুয়েরি কমান্ডের অন্যতম কাজটি ব্যাখ্যা কর।

গ. Teacher's table এর ফিল্ডগুলোর ডেটা টাইপ ব্যাখ্যা কর।

ঘ. উদ্দীপকের টেবিল দু'টির মধ্যে রিলেশন তৈরির সম্ভাব্যতা যাচাই কর।

২. একটি কলেজের ফলাফলের ডেটাবেস থেকে একজন শিক্ষার্থীর তথ্য খোঁজার জন্য তিনজন শিক্ষার্থীকে নির্দেশ দেয়া হলো। 1 শিক্ষার্থী শর্ত সাপেক্ষে কমান্ড দিয়ে, 2 শিক্ষার্থী ডেটাবেসের টেবিলে তথ্য সাজিয়ে এবং 3 শিক্ষার্থী 2 শিক্ষার্থীর চেয়ে দ্রুততর কৌশল প্রয়োগ করে তথ্য খুঁজে বের করে।

ক. ডেটা এনক্রিপশন কী?

খ. জাতীয় পরিচয়পত্রের তথ্য সম্বলিত ডেটাবেসের ধরন ব্যাখ্যা কর।

গ. তথ্য খোঁজার ক্ষেত্রে 2 শিক্ষার্থীর কৌশল বর্ণনা কর।

ঘ. 1 ও 3 শিক্ষার্থীর কৌশল দু'টির মধ্যে কোনটি উত্তম? বিশ্লেষণপূর্বক মতামত দাও।

৩. সংশ্লিষ্ট কর্তৃপক্ষ নির্বাচন অনুষ্ঠানের জন্য 'ক' এলাকার ভোটার তালিকা হালনাগাদ করার পরিকল্পনা করেছে। এ জন্য প্রয়োজনীয় তথ্যগুলো সরবরাহ করার জন্য তথ্য সংগ্রহকারীকে একজন ভোটারের নাম, পিতার নাম, বয়স, ধর্ম, জন্ম তারিখ, জন্মস্থান সংগ্রহ করার জন্য বললেন। উক্ত তথ্যগুলো দিয়ে একটি ডেটাবেস ফাইল তৈরি করার হলো। অন্যদিকে, নাম, বয়স ও জন্ম তারিখ ব্যবহার করে পরিসংখ্যান করার জন্য অপর একটি ফাইল তৈরি করা হলো।

ক. SQL কী?

খ. 'প্রাইমারি কি ও ফরেন কি এক নয়' ব্যাখ্যা কর।

গ. উদ্দীপকে বর্ণিত নির্বাচন অনুষ্ঠানের জন্য ডেটাবেস ফাইলের ফিল্ডের ডেটা টাইপের বর্ণনা দাও।

ঘ. উদ্দীপকে বর্ণিত দু'টি ফাইলের মধ্যে কীভাবে রিলেশন তৈরি করা যায়?- তোমার মতামত দাও।

৪. উদ্দীপকটি পড় এবং প্রশ্নগুলোর উত্তর দাও-

টেবিল ১

ID	Name	Address
1001	Ariful Haque	Khulna
1002	Shajeda Jannat	Dhaka
1003	Tahmid Salehin	Jamalpur

টেবিল ২

Sl.No.	Designations	Salary
1	Manager	45,000
2	Officer	30,000
3	Accountant	25,000

উক্ত টেবিল দু'টি থেকে যাদের বেতন 30,000 বা তার চেয়ে বেশি তাদের নাম ও পদবী দেখাতে বলা হলো। 'খ' নামক ব্যক্তি শর্তসাপেক্ষে কমান্ড দিয়েই উক্ত কাজটি করে দিল কিন্তু এই প্রক্রিয়ায় একটু বেশি সময় নিচ্ছিল। 'গ' নামক ব্যক্তি বলল, একটি গুরুত্বপূর্ণ ফাইল তৈরি করলে উক্ত কাজটি অনেকটা দ্রুত হবে তবে ডেটা এন্ট্রিতে একটু বেশি সময় নিবে।

ক. RDBMS কী?

খ. SQL-কে ডেটাবেসের অন্যতম হাতিয়ার বলার কারণ ব্যাখ্যা কর।

গ. উক্ত টেবিল দু'টিতে প্রয়োজনীয় কলাম যুক্ত করে ডেটা রিলেশন তৈরি কর।

ঘ. 'গ' ব্যক্তি যা বললো, তার সাথে তুমি কি একমত? উত্তর দাও।

৫. উদ্দীপকটি পড় এবং প্রশ্নগুলোর উত্তর দাও-

Name	Roll	DOB	Tution Fee
A	1011	02-2-2002	3500/-
B	1012	15-5-2003	4000/-
X	1013	22-8-2002	4200/-
Y	1014	27-3-2001	4100/-

চিত্র-1

Roll	Subject	Number	GPA
1011	ICT	70	A
1012	ICT	85	A^+
1013	ICT	90	A^+
1014	ICT	75	A

চিত্র-2

ক. কুয়েরি কী?

খ. গোপনীয়তাই ডেটার নিরাপত্তার প্রধান হাতিয়ার- ব্যাখ্যা কর।

গ. উদ্দীপকে ব্যবহৃত চিত্র-1 টেবিলে Roll ও DOB ফিল্ডের মাঝে Address ফিল্ড সংযোজন প্রক্রিয়া বর্ণনা কর।

ঘ. উদ্দীপকে দু'টি টেবিলের মধ্যে কী ধরনের Relation সম্ভব তা তোমার মতামতসহ ব্যাখ্যা কর।

৬. বাংলাদেশ পরিসংখ্যান ব্যুরো ও কৃষি সম্প্রসারন অধিদপ্তর যৌথভাবে দেশের যাবতীয় কৃষকদের একটি তালিকা তৈরি করেছে। এখানে তারা কৃষকদের নাম, জাতীয় পরিচয়পত্রের নম্বর, জন্মতারিখ, কৃষি খাতের নাম (যেমন, পোল্ট্রি, গবাদি পশুর খামার, চাষাবাদ ইত্যাদি), পরিবারের সদস্য সংখ্যা সহ আরো বিভিন্ন তথ্য সংগ্রহ করেছে।

ক. সাইফার টেক্সট কী?

খ. ডেটা সিকিউরিটির প্রয়োজনীয়তা ব্যাখ্যা কর?

গ. সরকারি প্রতিষ্ঠানের ডেটাবেজ তৈরির সময় কি কি বিষয় বিবেচনা করা উচিত?

ঘ) "ইনডেক্স তৈরি করার পর INSERT, UPDATE, DELETE কুয়েরি করতে বেশি সময় লাগে" – যুক্তিসহ ব্যাখ্যা কর।

সমাপ্ত