آموزشکده فنی پسران آمل- علامه حسن زاده آملی

پایگاه داده ها

کاردانی پیوسته کامپیوتر

حسین اخوان اسکی

کارشناس ارشد نرم افزار

آشنایی با درس پایگاه داده

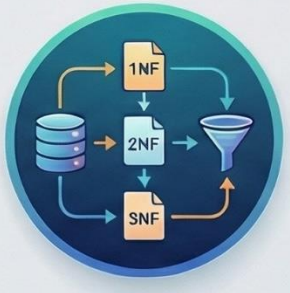
سرفصل‌های کلیدی آموزشی

روش‌های مدل‌سازی داده
شامل مدل‌های سلسله‌مراتبی، شبکه‌ای، رابطه‌ای و شیء‌گرا.

معماری پایگاه داده
آشنایی با معماری چندسطحی (ANSI) و مدل کلاتییت-سرور.

نرمال‌سازی داده‌ها
آموزش اهداف و قوانین فرم‌های نرمال اول، دوم و سوم.

معماری پایگاه داده
آشنایی با معاری چندصطی مدل کلاتنیت-سرور.



فهرست مطالب

عنوان **صفحه**

عنوان **صفحه**

عنوان **صفحه**

فصل اول

آشنایی عمیق با داده، فایل و ضرورت شکل‌گیری پایگاه داده

مقدمه

در دنیای امروز، تقریباً تمام فعالیت‌های انسانی به نوعی با ثبت و پردازش داده‌ها در ارتباط است. برای مثال، وقتی دانشجویی در دانشگاه ثبت‌نام می‌کند، اطلاعاتی مانند نام، شماره دانشجویی، رشته و واحدهای انتخابی او ثبت می‌شود. یا وقتی مشتری از یک فروشگاه خرید می‌کند، اطلاعات خرید، مبلغ و تاریخ در سیستم ذخیره می‌شود.

در گذشته، این اطلاعات معمولاً در قالب فایل‌های جداگانه ذخیره می‌شدند. فرض کنید در یک دانشگاه، اطلاعات دانشجویان در یک فایل، نمرات در فایل دیگر و شهریه در فایل سومی ذخیره شود. با افزایش تعداد دانشجویان، مدیریت این فایل‌ها به‌تدریج دشوار و پرخطا می‌شود. این فصل دقیقاً با هدف توضیح این مسیر نوشته شده است: از داده‌های ساده و فایل‌های ابتدایی تا نیاز به پایگاه داده.

داده(Data)

داده به واقعیت‌های خام و پردازش‌نشده گفته می‌شود که به‌تنهایی معنی مشخصی ندارند. داده‌ها می‌توانند عددی، متنی، تاریخی یا حتی صوتی و تصویری باشند، اما بدون قرار گرفتن در یک چارچوب مشخص، قابل تفسیر نیستند.

مثال:

- عدد «۱۷»
- کلمه «رضا»
- تاریخ «۱۰/۰۳/۱۴۰۳»

این موارد داده هستند، اما مشخص نیست:

- ۱۷نمره است یا سن؟
- رضا دانشجو است یا استاد؟
- این تاریخ مربوط به ثبت‌نام است یا امتحان؟

پس داده‌ها به‌تنهایی برای تصمیم‌گیری کافی نیستند.

اطلاعات(Information)

اطلاعات زمانی ایجاد می‌شود که داده‌ها پردازش شده و در یک زمینه مشخص معنا پیدا کنند. اطلاعات قابل تحلیل است و می‌تواند مبنای تصمیم‌گیری قرار گیرد.

مثال:

- »رضا در درس پایگاه داده نمره ۱۷ گرفته است«

- »دانشجو در تاریخ ۱۰/۰۳/۱۴۰۳ ثبت‌نام کرده است«

در این مثال‌ها:

- داده‌ها معنا پیدا کرده‌اند
- می‌توان تصمیم گرفت (قبولی، عدم قبولی، ثبت‌نام به‌موقع و ...)

📌رابطه داده و اطلاعات:

داده ⟶ پردازش ⟶ اطلاعات ⟶ تصمیم‌گیری

فایل(File)

فایل مجموعه‌ای از داده‌های مرتبط با یک موضوع مشخص است که برای ذخیره‌سازی دائمی استفاده می‌شود. در سیستم‌های اولیه، فایل اصلی‌ترین ابزار ذخیره‌سازی اطلاعات بود و هر برنامه فایل‌های مخصوص خود را داشت.

مثال:

فرض کنید یک فایل به نام Students.txtداریم که شامل اطلاعات دانشجویان است:

- شماره دانشجویی
- نام
- نام خانوادگی
- رشته
- معدل

این فایل برای نگهداری اطلاعات دانشجویان استفاده می‌شود، اما به‌تنهایی هیچ ارتباطی با فایل نمرات یا شهریه ندارد.

اجزای تشکیل‌دهنده فایل

فیلد(Field)

فیلد کوچک‌ترین واحد داده است و یک ویژگی مشخص را نگهداری می‌کند.

مثال فیلدها:

- شماره دانشجویی
- نام
- معدل

هر فیلد فقط یک مقدار مشخص را ذخیره می‌کند.

رکورد(Record)

رکورد مجموعه‌ای از فیلدهای مرتبط است که اطلاعات کامل یک موجودیت را نشان می‌دهد.

مثال یک رکورد دانشجو:

علی | رضایی | کامپیوتر | ۱۶.۸ | ۱۴۰۲۰۱۲۵

این رکورد، اطلاعات کامل یک دانشجو را نشان می‌دهد.

فایل(File)

فایل مجموعه‌ای از رکوردهاست.

مثال:
فایل اطلاعات دانشجویان شامل رکوردهای علی، رضا، مریم و سایر دانشجویان است.

ساختار سلسله‌مراتبی:

Field → Record → File

ذخیره‌سازی و بازیابی اطلاعات

ذخیره‌سازی به معنای ثبت داده‌ها در حافظه‌های دائمی مانند هارد دیسک یا سرور است. بازیابی به معنای دسترسی به این داده‌ها در زمان نیاز است.

مثال:

- وارد کردن نمرات دانشجویان در سیستم آموزش ← ذخیره‌سازی
- مشاهده کارنامه دانشجو در پایان ترم ← بازیابی

اگر ذخیره‌سازی درست انجام نشود یا بازیابی کند باشد، سیستم عملاً ناکارآمد خواهد شد.

سیستم فایل‌محور(File-Based System)

در سیستم فایل‌محور، داده‌ها در فایل‌های جداگانه ذخیره می‌شوند و هر برنامه مدیریت فایل‌های خودش را بر عهده دارد. هیچ سیستم مرکزی برای کنترل داده‌ها وجود ندارد.

مثال:

در یک دانشگاه:

- فایل دانشجویان توسط برنامه ثبت‌نام
- فایل نمرات توسط برنامه آموزش
- فایل شهریه توسط برنامه مالی

هر برنامه فایل خودش را دارد و ارتباط بین آن‌ها ضعیف یا غیرمستقیم است.

مشکلات سیستم فایل‌محور

افزونگی داده‌ها

نام و مشخصات دانشجو در چند فایل مختلف ذخیره می‌شود.

مثال:
نام دانشجو در فایل ثبت‌نام، فایل نمرات و فایل مالی تکرار می‌شود.

ناسازگاری داده‌ها

اگر یکی از فایل‌ها به‌روزرسانی نشود، اطلاعات متفاوت ثبت می‌شود.

مثال:
در فایل ثبت‌نام نام خانوادگی اصلاح شده، ولی در فایل نمرات هنوز قدیمی است.

امنیت پایین

کنترل اینکه چه کسی به کدام اطلاعات دسترسی دارد دشوار است.

مثال:
کارمند مالی ممکن است به نمرات دسترسی داشته باشد، در حالی که نباید.

وابستگی برنامه به داده

تغییر ساختار فایل باعث اختلال در برنامه می‌شود.

مثال:
اضافه شدن فیلد «شماره تماس» ← نیاز به تغییر برنامه

نتیجه‌گیری: چرا پایگاه داده؟

تمام مثال‌ها و مشکلات نشان می‌دهد که سیستم فایل‌محور برای محیط‌های بزرگ و واقعی مناسب نیست. پایگاه داده به‌وجود آمد تا:

- داده‌ها متمرکز شوند
- افزونگی کاهش یابد
- امنیت افزایش یابد
- گزارش‌گیری ساده‌تر شود

این فصل پایه‌ای‌ترین مفاهیم درس پایگاه داده را فراهم می‌کند و مقدمه‌ای برای مباحث تخصصی‌تر فصل‌های بعد است.

تمرین‌های پایان فصل اول

۱. برای داده و اطلاعات، هرکدام یک مثال جدید بزنید.

۲. برای یک فایل «کارمندان»، فیلدها و یک رکورد نمونه بنویسید.

۳. یک مثال واقعی از افزونگی داده‌ها در سیستم فایل‌محور بیان کنید.

٤. توضیح دهید چرا سیستم فایل‌محور برای یک بانک مناسب نیست.

فصل دوم

تعریف پایگاه داده‌ها، عناصر تخصصی، کاربران و سیستم‌های DBMS

تعریف پایگاه داده‌ها(Database)

پایگاه داده مجموعه‌ای سازمان‌یافته، ساخت‌یافته و مرتبط از داده‌هاست که به‌صورت متمرکز ذخیره می‌شوند و تحت کنترل یک سیستم مشخص مدیریت می‌گردند. هدف اصلی پایگاه داده، ذخیره‌سازی مؤثر حجم زیادی از اطلاعات به‌گونه‌ای است که بتوان آن‌ها را به‌سرعت جستجو کرد، به‌روزرسانی نمود، گزارش‌های متنوع تهیه کرد و هم‌زمان چندین کاربر بتوانند بدون تداخل از آن استفاده کنند.

در پایگاه داده، داده‌ها مستقل از برنامه‌های کاربردی ذخیره می‌شوند. این استقلال باعث می‌شود که تغییر در یک برنامه، الزاماً باعث تغییر در داده‌ها نشود و بالعکس. همچنین در پایگاه داده، ارتباط‌های منطقی میان داده‌ها برقرار می‌شود تا اطلاعات معنادار تولید گردد.

مثال کاربردی:

در یک سیستم آموزش دانشگاه:

- اطلاعات دانشجو (نام، شماره دانشجویی، رشته)
- اطلاعات درس (کد درس، نام درس)
- اطلاعات استاد
- نمرات

همگی در یک پایگاه داده ذخیره می‌شوند و از طریق ارتباط‌های مشخص به یکدیگر متصل‌اند.

عناصر تخصصی پایگاه داده

پایگاه داده فقط شامل داده‌های خام نیست، بلکه از اجزای تخصصی مختلفی تشکیل شده است که هرکدام نقش مهمی در صحت و کارایی سیستم دارند.

داده‌ها(Data)

داده‌ها اطلاعات خام و ثبت‌شده هستند، مانند نام، عدد، تاریخ یا کد.

ساختار داده(Data Structure)

ساختار داده مشخص می‌کند داده‌ها چگونه سازمان‌دهی شوند؛ معمولاً به‌صورت جدول، سطر و ستون.

ارتباط‌ها(Relationships)

ارتباط‌ها مشخص می‌کنند که داده‌ها چگونه به هم مرتبط‌اند، مثلاً ارتباط بین دانشجو و نمره.

محدودیت‌ها(Constraints)

قوانینی هستند که برای حفظ صحت داده‌ها تعریف می‌شوند؛ مانند یکتا بودن شماره دانشجویی.

فراداده‌ها(Metadata)

فراداده‌ها اطلاعاتی درباره خود داده‌ها هستند؛ مثل نوع داده (عدد، متن) یا طول فیلد.

مثال:

فیلد «نمره» از نوع عدد صحیح است و فقط مقادیر ۰ تا ۲۰ را می‌پذیرد. این اطلاعات جزو فراداده محسوب می‌شوند.

ویژگی‌های سخت‌افزاری پایگاه داده

پایگاه داده برای عملکرد صحیح نیاز به سخت‌افزار مناسب دارد. هرچه حجم داده‌ها و تعداد کاربران بیشتر باشد، سخت‌افزار قوی‌تری مورد نیاز است.

ویژگی‌های مهم سخت‌افزار پایگاه داده شامل:

- پردازنده قوی برای پردازش سریع درخواست‌ها
- حافظه ذخیره‌سازی مطمئن و پرسرعت
- تجهیزات شبکه پایدار
- سیستم‌های پشتیبان‌گیری و تحمل خطا

مثال:

پایگاه داده یک بانک باید همیشه فعال باشد و حتی در صورت خرابی یک سرور، سرور پشتیبان جایگزین شود.

معرفی انواع نرم‌افزارهای مرتبط با پایگاه داده

در محیط پایگاه داده، چند نوع نرم‌افزار به‌طور هم‌زمان نقش دارند:

سیستم عامل

مدیریت منابع سخت‌افزاری مانند حافظه و پردازنده را انجام می‌دهد.

سیستم مدیریت پایگاه داده(DBMS)

مدیریت کامل داده‌ها، امنیت، ذخیره و بازیابی اطلاعات را بر عهده دارد.

نرم‌افزارهای کاربردی

برنامه‌هایی هستند که کاربران از طریق آن‌ها با پایگاه داده کار می‌کنند.

مثال:

سامانه آموزش دانشگاه یک نرم‌افزار کاربردی است که به DBMS متصل می‌شود.

انواع کاربران پایگاه داده

کاربران پایگاه داده بر اساس نقش و سطح دسترسی به گروه‌های مختلف تقسیم می‌شوند.

مدیر پایگاه داده(DBA)

مهم‌ترین نقش را دارد و مسئول:

- طراحی پایگاه داده
- تعیین سطح دسترسی کاربران
- پشتیبان‌گیری
- بهینه‌سازی عملکرد

برنامه‌نویسان

نرم‌افزارهایی را طراحی می‌کنند که با پایگاه داده در ارتباط هستند.

کاربران نهایی

افرادی که از سیستم استفاده می‌کنند.

مثال:

دانشجو فقط مشاهده می‌کند، اما کارمند آموزش ثبت و ویرایش انجام می‌دهد.

ویژگی‌های داده‌ها در پایگاه داده

داده‌ها در پایگاه داده باید دارای ویژگی‌های زیر باشند:

- صحت (درست بودن)
- کامل بودن
- به‌روز بودن

- سازگاری
- امنیت

پایگاه داده با قوانین مشخص از ورود داده‌های نادرست جلوگیری می‌کند.

مثال:

سیستم اجازه ثبت دو دانشجو با یک شماره دانشجویی را نمی‌دهد.

انواع پایگاه‌های داده و کاربردهای آن‌ها

پایگاه‌های داده بر اساس کاربرد و ساختار به انواع مختلفی تقسیم می‌شوند.

کاربردها:

- بانکداری
- آموزش
- پزشکی
- تجارت الکترونیک
- شبکه‌های اجتماعی

مثال:

در فروشگاه اینترنتی، سفارش‌ها و پرداخت‌ها در پایگاه داده ذخیره می‌شوند.

سیستم مدیریت پایگاه داده(DBMS)

DBMS نرم‌افزاری است که تمام عملیات مربوط به پایگاه داده را مدیریت می‌کند. این سیستم امکان تعریف ساختار، ذخیره و بازیابی داده‌ها، کنترل امنیت و پشتیبان‌گیری را فراهم می‌کند.

بدونDBMS ، استفاده مؤثر از پایگاه داده امکان‌پذیر نیست.

سیستم مدیریت پایگاه داده رابطه‌ای(RDBMS)

RDBMS نوعی DBMS است که داده‌ها را به‌صورت جدول ذخیره می‌کند و ارتباط‌ها از طریق کلیدها برقرار می‌شوند.

بیشتر پایگاه‌های داده امروزی از این نوع هستند.

مثال:

ارتباط جدول دانشجویان با جدول نمرات از طریق شماره دانشجویی.

سیستم مدیریت پایگاه داده شیء–رابطه‌ای(ORDBMS)

ORDBMSترکیبی از پایگاه داده رابطه‌ای و شیءگرایی است و برای سیستم‌های پیچیده‌تر استفاده می‌شود.

مثال:

ذخیره تصاویر پزشکی یا فایل‌های چندرسانه‌ای.

جمع‌بندی نهایی

در این جزوه به‌صورت کامل با تعریف پایگاه داده‌ها، عناصر تخصصی، ویژگی‌های سخت‌افزاری، نرم‌افزارهای مرتبط، انواع کاربران، ویژگی‌های داده‌ها و انواع سیستم‌هایDBMS ، RDBMSو ORDBMS آشنا شدیم. این مطالب پایه اصلی درس پایگاه داده هستند.

تمرین‌های پایان فصل دوم

۱. پایگاه داده را تعریف کرده و سه مزیت آن نسبت به فایل بیان کنید.

۲. عناصر تخصصی پایگاه داده را نام ببرید و توضیح دهید.

۳. تفاوتDBMS ، RDBMSو ORDBMS را با مثال توضیح دهید.

٤. نقش DBA را به‌طور کامل شرح دهید.

٥. یک سیستم واقعی را انتخاب کرده و اجزای پایگاه داده آن را توضیح دهید.

فصل سوم

معماری پایگاه داده‌ها و دیدگاه‌های مختلف کاربران

مقدمه

در فصل‌های قبل با مفهوم پایگاه داده، سیستم مدیریت پایگاه داده، کاربران و انواع DBMS آشنا شدیم. اما تاکنون بیشتر تمرکز ما روی چیستی پایگاه داده بود. در این فصل می‌خواهیم به این سؤال پاسخ دهیم که **پایگاه داده چگونه طراحی و سازمان‌دهی می‌شود و اجزای آن چگونه با یکدیگر ارتباط دارند.**

معماری پایگاه داده مشخص می‌کند که داده‌ها در چه لایه‌هایی قرار می‌گیرند، کاربران چگونه به داده‌ها دسترسی دارند و سیستم مدیریت پایگاه داده چگونه این ارتباط را کنترل می‌کند. همچنین در این فصل با معماری‌های رایج مانند **کلاینت–سرور** و **معماری سه‌لایه‌ای ANSI/SPARC** آشنا می‌شویم و نقش دیدهای مختلف (داخلی، ادراکی و خارجی) را بررسی می‌کنیم. درک این فصل برای فهم استقلال داده، امنیت و عملکرد پایگاه داده بسیار ضروری است.

معماری پایگاه داده‌ها(Database Architecture)

معماری پایگاه داده به ساختار کلی سیستم پایگاه داده و نحوه ارتباط اجزای مختلف آن با یکدیگر گفته می‌شود. این معماری مشخص می‌کند که داده‌ها چگونه ذخیره می‌شوند، کاربران چگونه به داده‌ها دسترسی دارند و پردازش درخواست‌ها چگونه انجام می‌شود.

هدف اصلی معماری پایگاه داده، **افزایش کارایی، امنیت، استقلال داده‌ها و سهولت توسعه سیستم** است. با استفاده از معماری مناسب، می‌توان سیستم‌هایی طراحی کرد که هم پاسخ‌گوی تعداد زیادی کاربر باشند و هم در برابر تغییرات مقاوم باقی بمانند.

معماری کلاینت–سرور(Client–Server Architecture)

یکی از رایج‌ترین معماری‌های پایگاه داده، معماری **کلاینت–سرور** است. در این معماری، سیستم به دو بخش اصلی تقسیم می‌شود:

- **کلاینت :(Client)** بخشی که کاربر با آن کار می‌کند (مانند کامپیوتر کاربر یا نرم‌افزار کاربردی)
- **سرور :(Server)** بخشی که پایگاه داده و DBMS روی آن قرار دارد

در این معماری، کاربر درخواست خود را از طریق نرم‌افزار کلاینت ارسال می‌کند. این درخواست به سرور منتقل می‌شود، سرور آن را پردازش کرده و نتیجه را به کلاینت بازمی‌گرداند.

مثال:

در یک سامانه آموزشی دانشگاه:

- مرورگر یا نرم‌افزار دانشجو = کلاینت
- سرور دانشگاه که پایگاه داده روی آن نصب است = سرور

مزایا:

- متمرکز بودن داده‌ها
- امنیت بیشتر
- مدیریت آسان‌تر

معایب:

- وابستگی شدید به سرور
- کاهش کارایی در صورت افزایش بیش از حد کاربران

معماری ANSI/SPARC (معماری سه‌لایه‌ای)

برای حل بسیاری از مشکلات معماری‌های ساده، معماری **سه‌لایه‌ای ANSI/SPARC** معرفی شد. این معماری پایگاه داده را به سه دید یا سطح مستقل تقسیم می‌کند تا **استقلال داده‌ها** فراهم شود.

سه دید اصلی در این معماری عبارت‌اند از:

- دید داخلی(Internal View)
- دید ادراکی یا مفهومی(Conceptual View)
- دید خارجی(External View)

این معماری باعث می‌شود تغییر در یک سطح، کمترین تأثیر را بر سایر سطوح داشته باشد.

دید داخلی(Internal View)

دید داخلی پایین‌ترین سطح معماری پایگاه داده است و به نحوه ذخیره‌سازی فیزیکی داده‌ها مربوط می‌شود. در این دید، جزئیاتی مانند محل ذخیره داده‌ها روی دیسک، نوع فایل‌ها، ایندکس‌ها و روش‌های دسترسی به داده‌ها مشخص می‌شود.

این دید معمولاً فقط توسط سیستم مدیریت پایگاه داده و مدیر پایگاه داده مورد استفاده قرار می‌گیرد و کاربران عادی هیچ آگاهی‌ای از آن ندارند.

مثال:

اینکه اطلاعات دانشجویان روی چه هاردی، با چه فرمت فایلی و چه نوع ایندکسی ذخیره شده‌اند، مربوط به دید داخلی است.

دید ادراکی یا مفهومی(Conceptual View)

دید ادراکی، دید میانی در معماری ANSI/SPARC است و نمای کلی و منطقی از پایگاه داده ارائه می‌دهد. در این دید مشخص می‌شود که پایگاه داده شامل چه موجودیت‌هایی است، هر موجودیت چه ویژگی‌هایی دارد و ارتباط بین آن‌ها چگونه است.

این دید مستقل از نحوه ذخیره‌سازی فیزیکی داده‌هاست و معمولاً توسط طراحان پایگاه داده تعریف می‌شود.

مثال:

در دید ادراکی یک دانشگاه، موجودیت‌هایی مانند «دانشجو»، «درس» و «استاد» و ارتباط بین آن‌ها تعریف می‌شود، بدون توجه به اینکه داده‌ها چگونه ذخیره شده‌اند.

دید خارجی(External View)

دید خارجی بالاترین سطح معماری پایگاه داده است و نمایی است که هر کاربر از پایگاه داده می‌بیند. هر کاربر یا گروهی از کاربران می‌تواند دید خارجی مخصوص به خود را داشته باشد.

در این دید، فقط بخشی از داده‌ها که برای کاربر لازم است نمایش داده می‌شود و بقیه داده‌ها پنهان می‌مانند.

مثال:

- دانشجو فقط نمرات و اطلاعات شخصی خود را می‌بیند
- استاد فقط اطلاعات دانشجویان درس‌های خود را مشاهده می‌کند

ارتباط بین دیدها

دیدهای داخلی، ادراکی و خارجی به‌صورت سلسله‌مراتبی به هم مرتبط‌اند. دید خارجی به دید ادراکی متصل است و دید ادراکی به دید داخلی ارتباط دارد. این ارتباط باعث می‌شود تغییر در یک سطح، بدون تأثیر مستقیم بر سطوح دیگر انجام شود.

این ویژگی را **استقلال داده‌ها** می‌نامند که یکی از مهم‌ترین مزایای معماری پایگاه داده است.

زبان میزبان(Host Language)

زبان میزبان، زبان برنامه‌نویسی‌ای است که برنامه‌های کاربردی با استفاده از آن نوشته می‌شوند و از طریق آن به پایگاه داده متصل می‌شوند. زبان‌هایی مانندJava ، #C، Pythonو PHP نمونه‌هایی از زبان‌های میزبان هستند.

این زبان‌ها به‌تنهایی برای کار با پایگاه داده کافی نیستند و نیاز به زبان‌های خاص پایگاه داده دارند.

مثال:

یک برنامه Java که به پایگاه داده MySQL متصل می‌شود.

زبان فرعی داده(Data Sublanguage)

زبان فرعی داده زبانی است که برای تعریف، مدیریت و پرس‌وجو از داده‌ها استفاده می‌شود. مهم‌ترین زبان فرعی داده، **SQL**است.

زبان فرعی داده معمولاً شامل:

- زبان تعریف داده‌ها(DDL)
- زبان دستکاری داده‌ها(DML)
- زبان کنترل داده‌ها(DCL)

مثال:

دستور SELECTبرای بازیابی اطلاعات از پایگاه داده.

مدیر پایگاه داده(DBA)

مدیر پایگاه داده شخصی است که مسئول مدیریت کلی پایگاه داده است DBA .نقش بسیار مهمی در عملکرد، امنیت و سلامت پایگاه داده دارد.

وظایف مدیر پایگاه داده

وظایف مدیر پایگاه داده شامل:

- طراحی ساختار پایگاه داده
- تعیین سطح دسترسی کاربران
- پشتیبان‌گیری و بازیابی اطلاعات
- نظارت بر عملکرد سیستم
- حفظ امنیت داده‌ها

مثال:

DBAمشخص می‌کند چه کاربری اجازه حذف اطلاعات را دارد.

دیکشنری داده‌ها(Data Dictionary)

دیکشنری داده‌ها پایگاه داده‌ای است که اطلاعات مربوط به ساختار پایگاه داده را در خود نگهداری می‌کند. این اطلاعات شامل نام جداول، ستون‌ها، نوع داده‌ها، محدودیت‌ها و مجوزهاست.

دیکشنری داده‌ها برای DBMS و DBA بسیار حیاتی است.

مثال:

اطلاعات مربوط به اینکه جدول «دانشجو» چه ستون‌هایی دارد، در دیکشنری داده‌ها ذخیره می‌شود.

جمع‌بندی

در این فصل با معماری پایگاه داده‌ها، معماری کلاینت–سرور، معماریANSI/SPARC ، دیدهای داخلی، ادراکی و خارجی، ارتباط بین دیدها، زبان میزبان، زبان فرعی داده، نقش مدیر پایگاه داده و دیکشنری داده‌ها آشنا شدیم. این مفاهیم پایه طراحی و پیاده‌سازی سیستم‌های پایگاه داده هستند.

تمرین‌های پایان فصل سوم

۱. معماری پایگاه داده چیست و چرا اهمیت دارد؟

۲. معماری کلاینت–سرور را با مثال توضیح دهید.

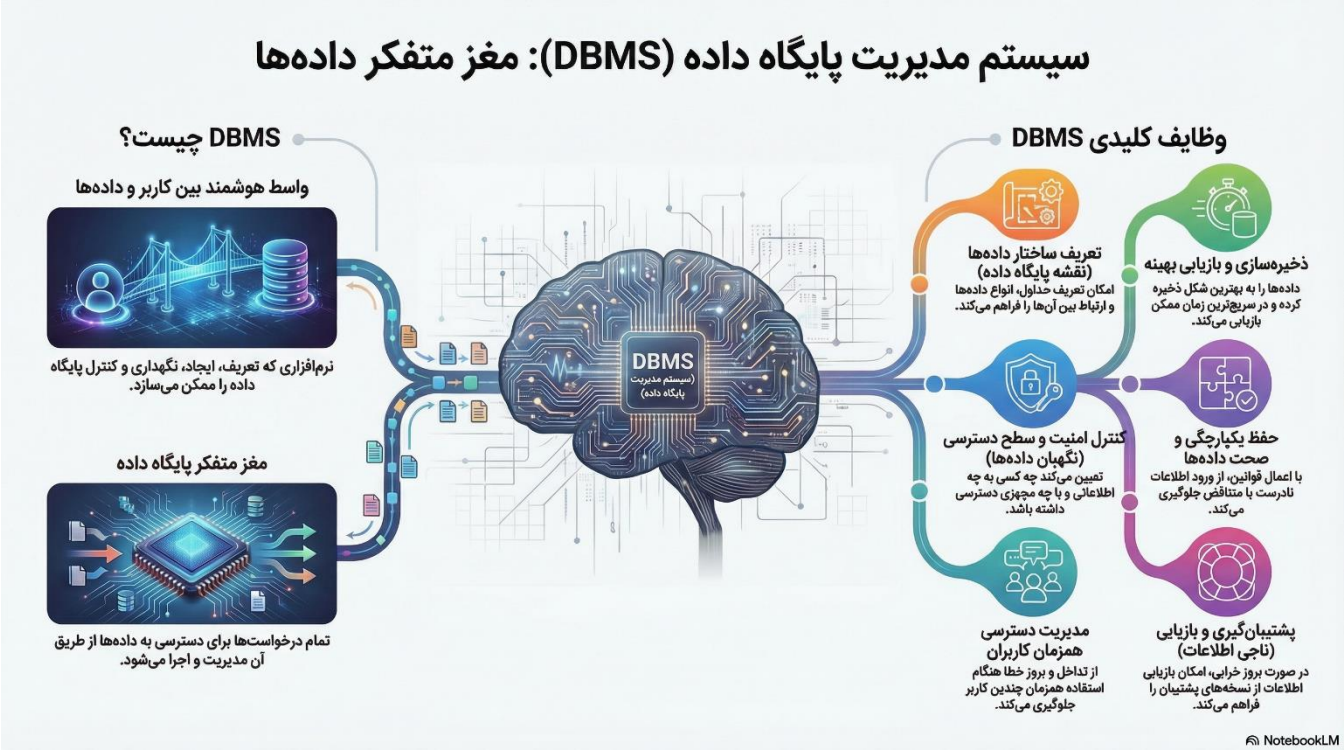
۳. سه دید معماری ANSI/SPARC را نام برده و توضیح دهید.

٤. تفاوت دید داخلی و دید خارجی چیست؟

٥. نقش DBA و دیکشنری داده‌ها را شرح دهید.

فصل چهارم

سیستم مدیریت پایگاه داده (DBMS) و ارتباط آن با سطوح معماری پایگاه داده



مقدمه

در فصل‌های گذشته با مفهوم پایگاه داده، انواع معماری‌های پایگاه داده و دیدگاه‌های مختلف کاربران آشنا شدیم. اما در تمام این مباحث، یک عنصر کلیدی به‌صورت غیرمستقیم حضور داشت و آن **سیستم مدیریت پایگاه داده (DBMS)** بود. در واقع، بدونDBMS ، پایگاه داده فقط مجموعه‌ای از فایل‌هاست که مدیریت آن‌ها بسیار دشوار و پرخطاست.

سیستم مدیریت پایگاه داده نقش واسط و هماهنگ‌کننده بین کاربران، برنامه‌های کاربردی و داده‌های ذخیره‌شده را بر عهده دارد. این سیستم مسئول اجرای دستورات کاربران، کنترل امنیت، حفظ صحت داده‌ها و مدیریت همزمانی دسترسی کاربران است. در این فصل، ابتدا DBMS را به‌طور دقیق معرفی می‌کنیم، سپس وظایف اصلی آن را بررسی کرده و در ادامه ارتباط DBMS با سطوح مختلف معماری پایگاه داده را توضیح می‌دهیم.

سیستم مدیریت پایگاه داده (DBMS) چیست؟

سیستم مدیریت پایگاه داده (Database Management System) نرم‌افزاری است که امکان تعریف، ایجاد، نگهداری و کنترل پایگاه داده را فراهم می‌کند DBMS .بین کاربران یا برنامه‌های کاربردی و داده‌های فیزیکی ذخیره‌شده قرار می‌گیرد و تمام درخواست‌ها برای دسترسی به داده‌ها از طریق آن انجام می‌شود.

به بیان ساده، DBMSمغز متفکر پایگاه داده است. کاربران مستقیماً با فایل‌های داده سروکار ندارند، بلکه از طریق دستورات یا نرم‌افزارهای کاربردی با DBMS ارتباط برقرار می‌کنند و DBMS مسئول اجرای صحیح این درخواست‌هاست.

مثال:

وقتی یک دانشجو نمرات خود را در سامانه آموزش مشاهده می‌کند، درخواست او ابتدا به DBMS ارسال می‌شود DBMS . داده‌های موردنظر را از پایگاه داده استخراج کرده و نتیجه را به کاربر نمایش می‌دهد.

وظایف اصلی سیستم مدیریت پایگاه داده

سیستم مدیریت پایگاه داده وظایف متعددی دارد که همگی در جهت حفظ صحت، امنیت و کارایی داده‌ها انجام می‌شوند. این وظایف باعث می‌شوند پایگاه داده به‌صورت پایدار و قابل اعتماد عمل کند.

تعریف ساختار پایگاه داده

DBMSامکان تعریف ساختار پایگاه داده را فراهم می‌کند. این ساختار شامل جداول، ستون‌ها، نوع داده‌ها و ارتباط بین آن‌هاست. بدونDBMS ، تعریف چنین ساختاری به‌صورت منظم امکان‌پذیر نیست.

مثال:

تعریف جدول «دانشجو» با ستون‌هایی مانند شماره دانشجویی، نام و رشته تحصیلی.

ذخیره‌سازی و بازیابی داده‌ها

یکی از مهم‌ترین وظایفDBMS ، ذخیره داده‌ها در حافظه دائمی و بازیابی سریع آن‌ها در زمان نیاز است DBMS . تصمیم می‌گیرد داده‌ها چگونه روی دیسک ذخیره شوند و با چه روشی سریع‌تر بازیابی شوند.

مثال:

نمایش لیست نمرات یک دانشجو در چند ثانیه، حتی اگر پایگاه داده شامل هزاران رکورد باشد.

کنترل امنیت و سطح دسترسی

DBMS امکان تعریف سطوح مختلف دسترسی برای کاربران را فراهم می‌کند. هر کاربر فقط به داده‌هایی دسترسی دارد که مجاز به مشاهده یا ویرایش آن‌هاست.

مثال:

دانشجو فقط اجازه مشاهده نمرات خود را دارد، اما استاد می‌تواند نمرات را ثبت یا اصلاح کند.

حفظ یکپارچگی داده‌ها

سیستم مدیریت پایگاه داده با اعمال محدودیت‌ها و قوانین، از صحت و اعتبار داده‌ها محافظت می‌کند. این قوانین مانع ورود داده‌های نادرست به سیستم می‌شوند.

مثال:

جلوگیری از ثبت نمره خارج از بازه صفر تا بیست.

مدیریت همزمانی کاربران(Concurrency Control)

در سیستم‌های واقعی، چندین کاربر به‌صورت همزمان به پایگاه داده دسترسی دارند DBMS .تضمین می‌کند که این دسترسی‌ها باعث تداخل یا ناسازگاری داده‌ها نشوند.

مثال:

دو کارمند به‌طور همزمان اطلاعات یک دانشجو را ویرایش می‌کنند؛ DBMS از بروز خطا جلوگیری می‌کند.

پشتیبان‌گیری و بازیابی اطلاعات

DBMSامکان تهیه نسخه پشتیبان از داده‌ها و بازیابی آن‌ها در صورت بروز خطا یا خرابی سیستم را فراهم می‌کند.

مثال:

در صورت قطع برق یا خرابی سخت‌افزار، اطلاعات پایگاه داده از نسخه پشتیبان بازیابی می‌شود.

مدیریت کارایی سیستم

سیستم مدیریت پایگاه داده با استفاده از روش‌هایی مانند ایندکس‌گذاری و بهینه‌سازی پرس‌وجوها، کارایی سیستم را افزایش می‌دهد.

مثال:
جستجوی سریع اطلاعات یک دانشجو با استفاده از شماره دانشجویی به‌عنوان کلید.

ارتباط سیستم مدیریت پایگاه داده و سطوح معماری پایگاه داده

در فصل سوم با معماری سه‌لایه‌ای ANSI/SPARC آشنا شدیم که شامل دید داخلی، دید ادراکی و دید خارجی است. سیستم مدیریت پایگاه داده نقش اصلی را در برقراری ارتباط بین این سطوح ایفا می‌کند.

DBMS و دید خارجی

DBMSدرخواست‌های کاربران را که از طریق دید خارجی مطرح می‌شوند، دریافت می‌کند. هر کاربر بر اساس سطح دسترسی خود، نمای خاصی از داده‌ها را مشاهده می‌کند.

مثال:
دانشجو فقط اطلاعات شخصی و نمرات خود را می‌بیند، نه اطلاعات سایر دانشجویان.

DBMS و دید ادراکی

در دید ادراکی، ساختار منطقی کل پایگاه داده تعریف می‌شود DBMS .این ساختار را مدیریت کرده و ارتباط بین موجودیت‌ها را کنترل می‌کند.

مثال:
ارتباط بین موجودیت‌های «دانشجو»، «درس» و «نمره» توسط DBMS مدیریت می‌شود.

DBMS و دید داخلی

در دید داخلی، نحوه ذخیره‌سازی فیزیکی داده‌ها مطرح است DBMS .مسئول نگاشت ساختار منطقی به ساختار فیزیکی و مدیریت فایل‌ها و ایندکس‌هاست.

مثال:
اینکه داده‌ها روی چه دیسکی و با چه فرمت ذخیره شوند، توسط DBMS تعیین می‌شود.

نقش DBMS در استقلال داده‌ها

یکی از مهم‌ترین نتایج ارتباط DBMS با سطوح معماری، **استقلال داده‌ها**ست. به این معنا که تغییر در نحوه ذخیره‌سازی فیزیکی داده‌ها، تأثیری بر برنامه‌های کاربردی ندارد.

مثال:
تغییر محل ذخیره داده‌ها از یک دیسک به دیسک دیگر بدون تغییر در نرم‌افزار کاربران.

جمع‌بندی

در این فصل با مفهوم سیستم مدیریت پایگاه داده، وظایف اصلی آن و نقش حیاتی DBMS در مدیریت، امنیت، صحت و کارایی داده‌ها آشنا شدیم. همچنین ارتباط DBMS با سطوح معماری پایگاه داده بررسی شد و دیدیم که چگونه DBMS باعث ایجاد استقلال داده‌ها و تسهیل دسترسی کاربران می‌شود. این مفاهیم پایه‌ای برای ورود به مباحث پیشرفته‌تر مانند طراحی پایگاه داده و زبان SQL هستند.

تمرین‌های پایان فصل چهارم

۱. سیستم مدیریت پایگاه داده چیست و چرا وجود آن ضروری است؟
۲. پنج وظیفه اصلی DBMS را نام ببرید و توضیح دهید.
۳. نقش DBMS در حفظ امنیت داده‌ها را تشریح کنید.
٤. ارتباط DBMS با دیدهای داخلی، ادراکی و خارجی را توضیح دهید.
٥. استقلال داده‌ها چیست و چگونه DBMS آن را فراهم می‌کند؟

فصل پنجم

روند اجرای درخواست کاربر در سیستم پایگاه داده

نحوه ارتباط کاربر با سیستم و

نحوه اجرای درخواست‌ها

مقدمه

در فصل‌های قبل با پایگاه داده، سیستم مدیریت پایگاه داده، معماری پایگاه داده و نقش DBMS آشنا شدیم. اما هنوز به‌طور دقیق بررسی نکرده‌ایم که **وقتی یک کاربر یک درخواست را ثبت می‌کند، این درخواست چگونه در سیستم پردازش می‌شود و در نهایت نتیجه آن چگونه به کاربر بازگردانده می‌شود.**

درک روند اجرای درخواست کاربر، به ما کمک می‌کند بفهمیم DBMS چگونه دستورات را تحلیل می‌کند، چگونه امنیت و صحت داده‌ها را کنترل می‌نماید و چگونه اطلاعات را از پایگاه داده استخراج می‌کند. این فصل پلی است میان مباحث نظری پایگاه داده و کار عملی با زبان.SQL

ارتباط کاربر با سیستم پایگاه داده

کاربران معمولاً به‌صورت مستقیم با پایگاه داده ارتباط برقرار نمی‌کنند، بلکه از طریق **نرم‌افزارهای واسط** یا **برنامه‌های کاربردی** با سیستم پایگاه داده در ارتباط هستند. این برنامه‌ها می‌توانند نرم‌افزارهای تحت وب، نرم‌افزارهای دسکتاپ یا حتی ابزارهای مدیریت پایگاه داده باشند.

در این ارتباط، کاربر درخواست خود را در قالب یک عملیات مشخص (مانند جستجو، ثبت یا ویرایش اطلاعات) به سیستم ارسال می‌کند. این درخواست به‌صورت یک دستور قابل فهم برای) DBMS معمولاً دستور (SQL تبدیل می‌شود.

مثال:

دانشجو در سامانه آموزشی روی گزینه «مشاهده نمرات» کلیک می‌کند. این عمل باعث ارسال یک درخواست به سیستم می‌شود که در پشت‌صحنه به یک دستور SQL تبدیل شده است.

اجزای درگیر در اجرای درخواست کاربر

در روند اجرای یک درخواست، اجزای مختلفی نقش دارند که هماهنگی آن‌ها باعث اجرای صحیح درخواست می‌شود. این اجزا شامل **کاربر، برنامه کاربردی، سیستم مدیریت پایگاه داده و پایگاه داده فیزیکی** هستند.

کاربر درخواست را ارسال می‌کند، برنامه کاربردی آن را دریافت کرده و به DBMS منتقل می‌کند .DBMS پس از بررسی و اجرای درخواست، نتیجه را از پایگاه داده دریافت کرده و به کاربر نمایش می‌دهد.

مراحل کلی اجرای درخواست کاربر

فرآیند اجرای یک درخواست در سیستم پایگاه داده معمولاً شامل چند مرحله متوالی است. این مراحل به‌گونه‌ای طراحی شده‌اند که هم صحت داده‌ها حفظ شود و هم امنیت و کارایی سیستم تضمین گردد.

در مرحله اول، **دریافت درخواست** انجام می‌شود DBMS .درخواست را از برنامه کاربردی یا کاربر دریافت می‌کند. در مرحله بعد، **تحلیل درخواست** صورت می‌گیرد تا مشخص شود درخواست از نظر ساختار و نحو (Syntax) صحیح است یا خیر.

سپس، DBMS**مجوزهای دسترسی** کاربر را بررسی می‌کند تا مطمئن شود کاربر اجازه انجام این عملیات را دارد. پس از آن، درخواست وارد مرحله **بهینه‌سازی** می‌شود تا بهترین و سریع‌ترین روش اجرای آن انتخاب گردد. در نهایت، درخواست اجرا شده و نتیجه به کاربر بازگردانده می‌شود.

تحلیل و تفسیر درخواست(Query Parsing)

در این مرحله، DBMSدستور دریافتی را بررسی می‌کند تا از نظر قواعد زبانی صحیح باشد. اگر دستور دارای خطا باشد، اجرای آن متوقف شده و پیام خطا به کاربر بازگردانده می‌شود.

مثال:

اگر کاربر دستور SQL را اشتباه بنویسد، مانند نوشتن نام نادرست یک جدول، DBMSدر این مرحله خطا را تشخیص می‌دهد.

بررسی مجوزها و امنیت درخواست

پس از تحلیل نحوی، DBMSبررسی می‌کند که آیا کاربر مجوز انجام درخواست موردنظر را دارد یا خیر. این مرحله نقش بسیار مهمی در حفظ امنیت پایگاه داده دارد.

مثال:

اگر یک دانشجو تلاش کند اطلاعات دانشجویان دیگر را مشاهده کند، DBMSاین درخواست را رد می‌کند.

بهینه‌سازی درخواست(Query Optimization)

در این مرحله، DBMSبهترین روش اجرای درخواست را انتخاب می‌کند. ممکن است چندین روش مختلف برای اجرای یک دستور وجود داشته باشد، اما DBMS روشی را انتخاب می‌کند که سریع‌تر و کم‌هزینه‌تر باشد.

مثال:

اگر برای یک ستون ایندکس وجود داشته باشد، DBMSاز آن برای جستجوی سریع‌تر استفاده می‌کند.

اجرای درخواست و دسترسی به داده‌ها

پس از انتخاب بهترین روش، DBMSدرخواست را اجرا می‌کند. در این مرحله، داده‌ها از پایگاه داده فیزیکی خوانده یا تغییر داده می‌شوند.

مثال:

دریافت نمرات یک دانشجو از جدول نمرات.

بازگرداندن نتیجه به کاربر

در آخرین مرحله، نتیجه اجرای درخواست به برنامه کاربردی ارسال شده و به‌صورت قابل فهم به کاربر نمایش داده می‌شود. این نتیجه می‌تواند شامل داده‌ها یا پیام موفقیت عملیات باشد.

مثال:

نمایش لیست نمرات در صفحه کاربری دانشجو.

مثال کامل از روند اجرای یک درخواست

فرض کنید یک دانشجو می‌خواهد نمرات خود را مشاهده کند:

۱. دانشجو درخواست را در سامانه ثبت می‌کند.

۲. برنامه کاربردی درخواست را به دستور SQL تبدیل می‌کند.

۳. DBMSدستور را دریافت و تحلیل می‌کند.

٤. مجوزهای دسترسی بررسی می‌شود.

٥. بهترین روش اجرای دستور انتخاب می‌شود.

٦. داده‌ها از پایگاه داده استخراج می‌شوند.

۷. نتیجه به کاربر نمایش داده می‌شود.

جمع‌بندی

در این فصل با روند کامل اجرای درخواست کاربر در سیستم پایگاه داده آشنا شدیم. دیدیم که یک درخواست ساده کاربر شامل مراحل متعددی مانند تحلیل، بررسی امنیت، بهینه‌سازی و اجراست. شناخت این مراحل به درک بهتر عملکرد DBMSو کار عملی با پایگاه داده کمک زیادی می‌کند.

تمرین‌های پایان فصل پنجم

۱. مراحل اجرای یک درخواست کاربر در سیستم پایگاه داده را توضیح دهید.

۲. نقش DBMS در اجرای درخواست‌ها چیست؟

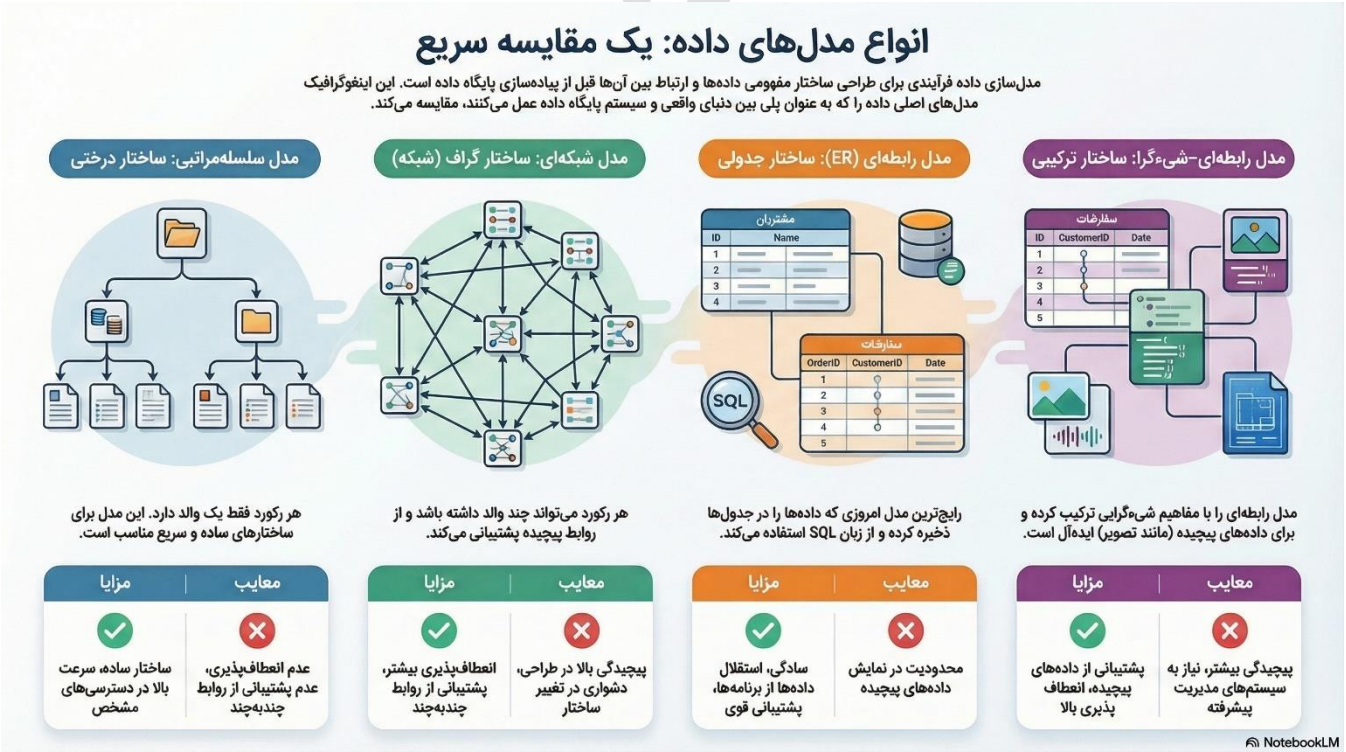
۳. چرا بررسی مجوزهای دسترسی قبل از اجرای درخواست ضروری است؟

٤. بهینه‌سازی درخواست چیست و چه اهمیتی دارد؟

٥. روند اجرای درخواست «مشاهده نمرات دانشجو» را به‌صورت مرحله‌به‌مرحله توضیح دهید.

فصل ششم

انواع روش‌های مدل‌سازی داده



مقدمه

در فصل‌های گذشته با پایگاه داده، سیستم مدیریت پایگاه داده، معماری آن و نحوه اجرای درخواست‌های کاربران آشنا شدیم. اما قبل از آنکه بتوان یک پایگاه داده را پیاده‌سازی کرد، لازم است داده‌ها و ارتباط میان آن‌ها به‌درستی **مدل‌سازی** شوند. مدل‌سازی داده به ما کمک می‌کند ساختار داده‌ها را به‌صورت مفهومی و قابل فهم طراحی کنیم، بدون آنکه وارد جزئیات فنی ذخیره‌سازی شویم.

در این فصل با مفهوم مدل داده و انواع روش‌های مدل‌سازی داده آشنا می‌شویم. سپس مهم‌ترین مدل‌های داده شامل **مدل سلسله‌مراتبی، مدل شبکه‌ای، مدل رابطه‌ای (ER) و مدل رابطه‌ای–شیءگرا** را به‌صورت کامل بررسی می‌کنیم و مزایا و معایب هرکدام را توضیح می‌دهیم.

مدل‌سازی داده چیست؟

مدل‌سازی داده فرآیندی است که در آن ساختار داده‌ها، نوع داده‌ها، ارتباط میان آن‌ها و محدودیت‌های موجود مشخص می‌شود. هدف از مدل‌سازی داده، ارائه تصویری ساده و قابل درک از دنیای واقعی است که بتوان بر اساس آن پایگاه داده را طراحی و پیاده‌سازی کرد.

مدل داده به‌عنوان یک پل ارتباطی بین دنیای واقعی و سیستم پایگاه داده عمل می‌کند. طراح پایگاه داده ابتدا داده‌ها را مدل‌سازی می‌کند و سپس بر اساس این مدل، پایگاه داده واقعی ایجاد می‌شود.

مثال:

قبل از طراحی پایگاه داده یک دانشگاه، ابتدا موجودیت‌هایی مانند «دانشجو»، «درس» و «استاد» شناسایی و ارتباط آن‌ها مشخص می‌شود.

مدل داده سلسله‌مراتبی(Hierarchical Data Model)

مدل داده سلسله‌مراتبی یکی از قدیمی‌ترین مدل‌های داده است که در آن داده‌ها به‌صورت یک ساختار درختی سازمان‌دهی می‌شوند. در این مدل، هر رکورد می‌تواند فقط **یک والد** داشته باشد، اما می‌تواند چندین **فرزند** داشته باشد.

در این ساختار، ارتباط‌ها از بالا به پایین تعریف می‌شوند و دسترسی به داده‌ها معمولاً از ریشه درخت آغاز می‌شود.

مثال:

در یک سازمان:

- سازمان (ریشه)
- بخش‌ها (فرزندان)
- کارمندان (زیرمجموعه بخش‌ها)

مزایا:

- ساختار ساده و قابل فهم
- سرعت بالا در دسترسی‌های مشخص و تکراری

معایب:

- عدم انعطاف‌پذیری
- عدم پشتیبانی مناسب از ارتباط‌های چندبه‌چند
- وابستگی شدید به ساختار فیزیکی داده‌ها

مدل داده شبکه‌ای(Network Data Model)

مدل شبکه‌ای توسعه‌یافته مدل سلسله‌مراتبی است. در این مدل، هر رکورد می‌تواند **چند والد** و **چند فرزند** داشته باشد. ارتباط‌ها به‌صورت یک شبکه از گره‌ها تعریف می‌شوند.

این مدل انعطاف‌پذیری بیشتری نسبت به مدل سلسله‌مراتبی دارد و می‌تواند ارتباط‌های پیچیده‌تری را نمایش دهد.

مثال:

در سیستم دانشگاهی:

- یک دانشجو می‌تواند چند درس داشته باشد
- هر درس می‌تواند چند دانشجو داشته باشد

مزایا:

- پشتیبانی از ارتباط‌های چندبه‌چند
- انعطاف‌پذیری بیشتر نسبت به مدل سلسله‌مراتبی

معایب:

- پیچیدگی بالا در طراحی و پیاده‌سازی
- دشواری در تغییر ساختار داده‌ها
- وابستگی زیاد به مسیرهای دسترسی

مدل داده رابطه‌ای و(Relational & ER Model) ER

مدل رابطه‌ای رایج‌ترین و پرکاربردترین مدل داده در سیستم‌های پایگاه داده امروزی است. در این مدل، داده‌ها در قالب **جدول‌ها (Relation)** ذخیره می‌شوند. هر جدول شامل سطرها (رکوردها) و ستون‌ها (ویژگی‌ها) است.

مدل (Entity-Relationship) ER یک مدل مفهومی است که برای طراحی پایگاه داده رابطه‌ای استفاده می‌شود. در این مدل، موجودیت‌ها، ویژگی‌ها و ارتباط‌ها به‌صورت نمودار نمایش داده می‌شوند.

مثال:

- موجودیت: دانشجو
- ویژگی‌ها: شماره دانشجویی، نام، رشته
- ارتباط: انتخاب درس

مزایا:

- سادگی و قابل فهم بودن
- استقلال داده‌ها از برنامه‌ها
- پشتیبانی قوی توسط DBMS ها
- امکان استفاده از زبان SQL

معایب:

- محدودیت در نمایش داده‌های پیچیده
- نیاز به تبدیل مدل ER به جدول‌های رابطه‌ای

مدل داده رابطه‌ای–شیءگرا(Object-Relational Data Model)

مدل رابطه‌ای–شیءگرا ترکیبی از مدل رابطه‌ای و مفاهیم برنامه‌نویسی شیءگراست. این مدل امکان تعریف نوع داده‌های پیچیده، اشیاء و متدها را در کنار جدول‌ها فراهم می‌کند.

این مدل برای سیستم‌هایی مناسب است که داده‌های پیچیده‌ای مانند تصویر، ویدئو یا داده‌های چندرسانه‌ای دارند.

مثال:

ذخیره اطلاعات پزشکی شامل تصویر MRI در کنار اطلاعات متنی بیمار.

مزایا:

- پشتیبانی از داده‌های پیچیده
- انعطاف‌پذیری بالا
- سازگاری با سیستم‌های شیءگرا

معایب:

- پیچیدگی بیشتر نسبت به مدل رابطه‌ای
- نیاز به DBMS های پیشرفته‌تر
- یادگیری دشوارتر

مقایسه کلی مدل‌های داده

مدل سلسله‌مراتبی و شبکه‌ای بیشتر جنبه تاریخی دارند و امروزه کمتر استفاده می‌شوند، در حالی که مدل رابطه‌ای و مدل رابطه‌ای–شیءگرا پایه اصلی پایگاه داده‌های مدرن را تشکیل می‌دهند. انتخاب مدل داده به نوع سیستم، حجم داده‌ها و نیازهای کاربردی بستگی دارد.

جمع‌بندی

در این فصل با مفهوم مدل‌سازی داده و انواع مدل‌های داده شامل سلسله‌مراتبی، شبکه‌ای، رابطه‌ای (ER) و رابطه‌ای–شیءگرا آشنا شدیم. همچنین مزایا و معایب هر مدل بررسی شد. این فصل مقدمه‌ای مهم برای یادگیری طراحی پایگاه داده و رسم نمودار ER در فصل‌های بعدی است.

تمرین‌های پایان فصل ششم

۱. مدل‌سازی داده چیست و چرا اهمیت دارد؟

۲. مدل داده سلسله‌مراتبی را با مثال توضیح دهید.

۳. تفاوت مدل شبکه‌ای و سلسله‌مراتبی چیست؟

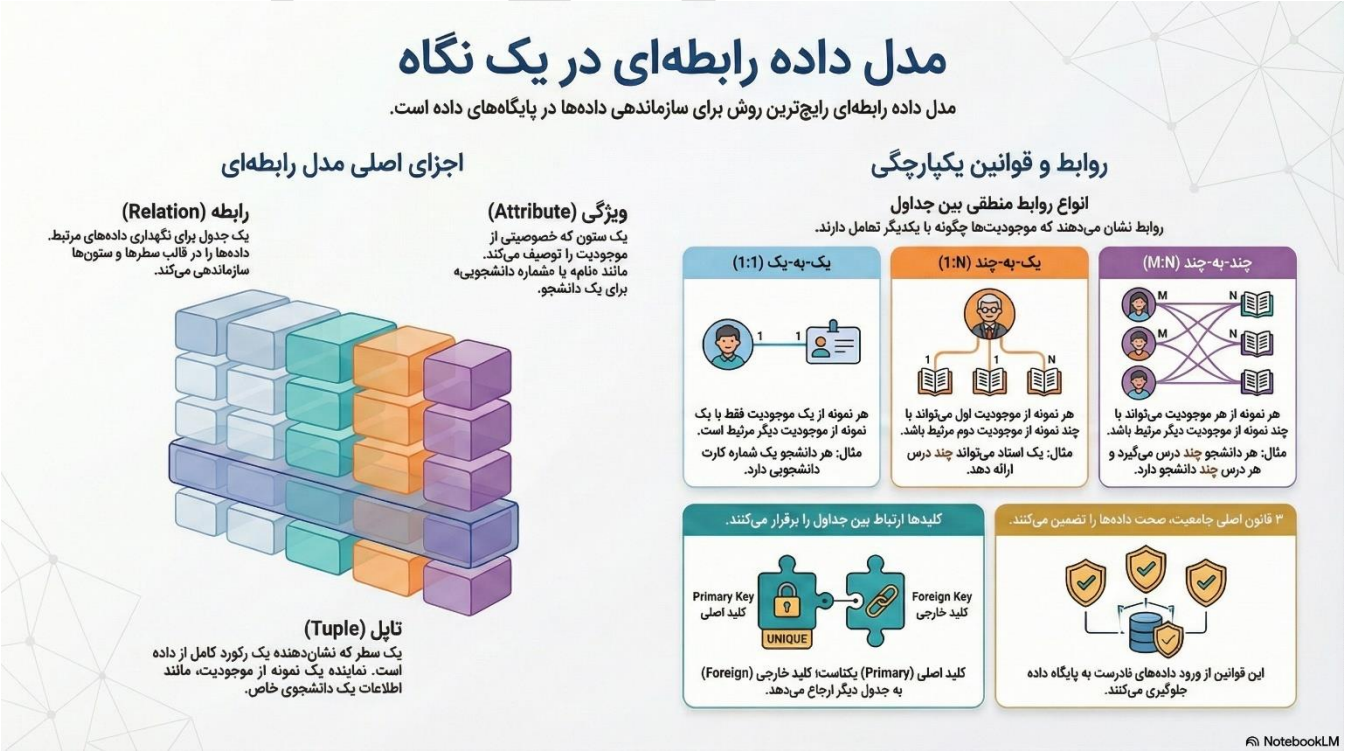
٤. چرا مدل رابطه‌ای پرکاربردترین مدل داده است؟

٥. مزایا و معایب مدل رابطه‌ای–شیءگرا را بیان کنید.

فصل هفتم

مدل داده رابطه‌ای

(Relational Data Model)



مقدمه

در فصل قبل با انواع روش‌های مدل‌سازی داده آشنا شدیم و دیدیم که مدل رابطه‌ای رایج‌ترین و پرکاربردترین مدل داده در سیستم‌های پایگاه داده امروزی است. برای آنکه بتوانیم پایگاه داده‌ای دقیق، منسجم و قابل توسعه طراحی کنیم، لازم است مفاهیم مدل داده رابطه‌ای را به‌صورت عمیق و اصولی بشناسیم.

در این فصل، ابتدا عناصر اصلی مدل رابطه‌ای مانند رابطه، ویژگی، تاپل، بسط و کاردینالیتی را بررسی می‌کنیم. سپس به مفاهیم موجودیت و انواع آن، انواع رابطه‌ها، ویژگی‌ها و در نهایت مفهوم جامعیت و قواعد جامعیت در مدل داده رابطه‌ای می‌پردازیم. این فصل پایه اصلی طراحی منطقی پایگاه داده و پیش‌نیاز یادگیری SQL است.

مدل داده رابطه‌ای چیست؟

مدل داده رابطه‌ای مدلی است که در آن داده‌ها به‌صورت مجموعه‌ای از **رابطه‌ها (Relation)** ذخیره می‌شوند. هر رابطه در عمل یک جدول است که از سطرها و ستون‌ها تشکیل شده است. این مدل بر پایه نظریه ریاضی مجموعه‌ها و روابط بنا شده و به همین دلیل از دقت و انسجام بالایی برخوردار است.

در مدل رابطه‌ای، هر داده فقط یک‌بار ذخیره می‌شود و ارتباط بین داده‌ها از طریق کلیدها برقرار می‌گردد. همین ویژگی باعث کاهش افزونگی و افزایش یکپارچگی داده‌ها می‌شود.

عناصر اصلی مدل داده رابطه‌ای

رابطه(Relation)

رابطه در مدل داده رابطه‌ای به‌معنای یک جدول است که شامل مجموعه‌ای از داده‌های مرتبط می‌باشد. هر رابطه دارای نام مشخصی است و داده‌ها در قالب سطر و ستون در آن ذخیره می‌شوند.

مثال:

جدول «دانشجو» یک رابطه است که اطلاعات مربوط به دانشجویان را نگهداری می‌کند.

ویژگی(Attribute)

ویژگی یا صفت، مشخص‌کننده یک خصوصیت از موجودیت است و به‌صورت ستون در جدول نمایش داده می‌شود. هر ویژگی دارای نام و نوع داده مشخصی است.

مثال:

نام، شماره دانشجویی و رشته تحصیلی ویژگی‌های موجودیت دانشجو هستند.

تاپل(Tuple)

تاپل یک سطر از جدول است که مجموعه‌ای از مقادیر مربوط به ویژگی‌های یک رابطه را در بر دارد. هر تاپل نماینده یک نمونه از موجودیت است.

مثال:

یک سطر شامل اطلاعات یک دانشجو خاص، یک تاپل محسوب می‌شود.

بسط(Extension)

بسط به مجموعه تمام تاپل‌های موجود در یک رابطه در یک زمان مشخص گفته می‌شود. به عبارت دیگر، محتوای فعلی جدول را بسط می‌نامند.

مثال:

تمام رکوردهای جدول دانشجو در یک ترم تحصیلی خاص، بسط آن جدول است.

کاردینالیتی(Cardinality)

کاردینالیتی به تعداد تاپل‌های موجود در یک رابطه گفته می‌شود. این مفهوم نشان می‌دهد که یک جدول در یک زمان مشخص چند رکورد دارد.

مثال:

اگر جدول دانشجو شامل ۵۰۰ رکورد باشد، کاردینالیتی آن ۵۰۰ است.

مفاهیم موجودیت(Entity)

موجودیت به هر شیء یا مفهومی از دنیای واقعی گفته می‌شود که دارای وجود مستقل بوده و اطلاعاتی درباره آن ذخیره می‌شود.

مثال:

دانشجو، استاد، درس

موجودیت ضعیف(Weak Entity)

موجودیت ضعیف موجودیتی است که بدون وابستگی به موجودیت دیگر معنا ندارد و کلید مستقل ندارد.

مثال:

نمره بدون دانشجو معنا ندارد، بنابراین موجودیت نمره یک موجودیت ضعیف است.

فرا موجودیت(Super Entity)

فرا موجودیت موجودیتی کلی است که ویژگی‌های مشترک چند موجودیت دیگر را در بر می‌گیرد.

مثال:

»شخص» می‌تواند فرا موجودیت «دانشجو» و «استاد» باشد.

مفاهیم رابطه(Relationship)

رابطه به ارتباط منطقی بین دو یا چند موجودیت گفته می‌شود. این ارتباط‌ها نشان می‌دهند که موجودیت‌ها چگونه با یکدیگر تعامل دارند.

رابطه یک‌به‌یک(1:1)

در این نوع رابطه، هر نمونه از یک موجودیت فقط با یک نمونه از موجودیت دیگر مرتبط است.

مثال:

هر دانشجو یک شماره کارت دانشجویی دارد.

رابطه یک‌به‌چند(1:N)

در این رابطه، هر نمونه از موجودیت اول می‌تواند با چند نمونه از موجودیت دوم مرتبط باشد.

مثال:

یک استاد می‌تواند چند درس ارائه دهد.

رابطه چندبه‌چند(M:N)

در این نوع رابطه، هر نمونه از هر موجودیت می‌تواند با چند نمونه از موجودیت دیگر مرتبط باشد.

مثال:

دانشجویان و دروس (هر دانشجو چند درس، هر درس چند دانشجو)

مفاهیم ویژگی(Attribute)

صفت کلید تخصصی(Primary Key)

کلید اصلی یا تخصصی، ویژگی‌ای است که هر تاپل را به‌طور یکتا مشخص می‌کند.

مثال: شماره دانشجویی

صفت کلید خارجی(Foreign Key)

کلید خارجی ویژگی‌ای است که به کلید اصلی جدول دیگر اشاره می‌کند و ارتباط بین جداول را برقرار می‌سازد.

مثال:

شماره دانشجویی در جدول نمرات

صفت استنتاجی(Derived Attribute)

صفتی است که مقدار آن از سایر ویژگی‌ها محاسبه می‌شود.

مثال:

سن دانشجو از تاریخ تولد محاسبه می‌شود.

صفت چندگانه(Multivalued Attribute)

ویژگی‌ای است که می‌تواند بیش از یک مقدار داشته باشد.

مثال:

شماره تلفن‌های یک دانشجو

جامعیت (Integrity) در مدل داده رابطه‌ای

جامعیت به مجموعه قوانینی گفته می‌شود که صحت، سازگاری و اعتبار داده‌ها را تضمین می‌کند. این قوانین مانع ورود داده‌های نادرست به پایگاه داده می‌شوند.

قواعد جامعیت

جامعیت موجودیت:

هر جدول باید دارای کلید اصلی باشد و مقدار آن نباید تکراری یا تهی باشد.

جامعیت ارجاعی:

کلید خارجی باید به یک کلید اصلی معتبر در جدول دیگر اشاره کند.

جامعیت دامنه:

مقادیر هر ویژگی باید در دامنه مشخص‌شده قرار گیرند.

مثال:

نمره باید بین ۰ تا ۲۰ باشد.

جمع‌بندی

در این فصل با مفاهیم پایه مدل داده رابطه‌ای شامل رابطه، ویژگی، تاپل، کاردینالیتی، موجودیت‌ها، انواع رابطه‌ها، انواع ویژگی‌ها و قواعد جامعیت آشنا شدیم. این فصل اساس طراحی منطقی پایگاه داده و پیش‌نیاز مستقیم پیاده‌سازی پایگاه داده و نوشتن دستورات SQL است.

تمرین‌های پایان فصل هفتم

۱. رابطه، ویژگی و تاپل را با مثال توضیح دهید.

۲. تفاوت موجودیت قوی و ضعیف چیست؟

۳. انواع رابطه‌ها را با مثال توضیح دهید.

٤. تفاوت کلید اصلی و کلید خارجی چیست؟

٥. قواعد جامعیت را تعریف کرده و اهمیت آن‌ها را توضیح دهید.

فصل هشتم

پیاده‌سازی عملیات روی رابطه‌ها

زبان SQL استاندارد و دستورات

تعریف داده، دستکاری داده و مدیریت داده

مقدمه

تا فصل هفتم، با مفاهیم نظری پایگاه داده رابطه‌ای، ساختار جداول، موجودیت‌ها، روابط و قواعد جامعیت آشنا شدیم. اما طراحی پایگاه داده بدون پیاده‌سازی عملی، کاربردی نخواهد داشت. برای آنکه بتوانیم داده‌ها را در یک پایگاه داده واقعی ایجاد، ذخیره، ویرایش، حذف و بازیابی کنیم، به یک زبان استاندارد نیاز داریم.

این زبان، **SQL (Structured Query Language)**است SQL .زبان استاندارد کار با پایگاه‌های داده رابطه‌ای است و تقریباً در تمامDBMS های معروف مانندMySQL ، SQL Server، Oracleو PostgreSQL پشتیبانی می‌شود. در این فصل یاد می‌گیریم چگونه با استفاده از SQL عملیات مختلف را روی رابطه‌ها (جداول) انجام دهیم؛ از ایجاد جدول گرفته تا ثبت داده و استخراج اطلاعات.

زبان SQL چیست؟

SQLمخفف Structured Query Language به‌معنای «زبان پرس‌وجوی ساخت‌یافته» است. این زبان برای ارتباط بین کاربران و سیستم مدیریت پایگاه داده طراحی شده است و به ما اجازه می‌دهد ساختار پایگاه داده را تعریف کنیم، داده‌ها را تغییر دهیم و اطلاعات مورد نیاز را استخراج کنیم.

SQLیک زبان **غیررویه‌ای** است؛ یعنی کاربر مشخص می‌کند «چه چیزی» می‌خواهد، نه اینکه «چگونه» سیستم آن را انجام دهد. این ویژگی باعث ساده‌تر شدن کار با پایگاه داده می‌شود.

مثال:

کاربر می‌گوید:
»اطلاعات دانشجویان رشته کامپیوتر را نمایش بده«
و DBMS خودش بهترین روش اجرا را انتخاب می‌کند.

دسته‌بندی دستوراتSQL

دستورات SQL به‌طور کلی به سه دسته اصلی تقسیم می‌شوند:

۱. **دستورات تعریف داده(DDL)**
۲. **دستورات دستکاری داده(DML)**
۳. **دستورات مدیریت داده(DCL)**

هر یک از این دسته‌ها نقش مشخصی در پیاده‌سازی عملیات روی رابطه‌ها دارند.

دستورات تعریف داده(DDL)

دستورات DDL برای **ایجاد، تغییر و حذف ساختار پایگاه داده** به‌کار می‌روند. این دستورات مستقیماً روی ساختار رابطه‌ها (جداول) اثر می‌گذارند.

مهم‌ترین دستورات DDL عبارت‌اند از:

- CREATE
- ALTER
- DROP

دستور CREATE

دستور CREATE برای ایجاد پایگاه داده یا جدول جدید استفاده می‌شود.

مثال: ایجاد جدول دانشجو

```
CREATE TABLE Student (
  StudentID INT PRIMARY KEY,
  Name VARCHAR(50),
  Major VARCHAR(30),
  Age INT
);
```

در این مثال:

- StudentIDکلید اصلی است
- Nameو Major ویژگی‌های متنی هستند
- Ageیک ویژگی عددی است

دستور ALTER

دستور ALTER برای تغییر ساختار یک جدول موجود استفاده می‌شود؛ مثلاً اضافه یا حذف یک ستون.

مثال: اضافه کردن ستون معدل

```
ALTER TABLE Student
ADD GPA FLOAT;
```

دستور DROP

دستور DROP برای حذف کامل یک جدول یا پایگاه داده به‌کار می‌رود.

مثال:

```
DROP TABLE Student;
```

این دستور جدول و تمام داده‌های آن را به‌طور کامل حذف می‌کند.

دستورات دستکاری داده(DML)

دستورات DML برای **کار با داده‌های داخل جداول** استفاده می‌شوند، بدون اینکه ساختار جدول تغییر کند.

مهم‌ترین دستورات:DML

- INSERT
- UPDATE
- DELETE
- SELECT

دستور INSERT

برای افزودن داده جدید به جدول استفاده می‌شود.

مثال:

```
INSERT INTO Student
VALUES (1, 'علی احمدی', 'کامپیوتر', 20);
```

دستور UPDATE

برای ویرایش داده‌های موجود به‌کار می‌رود.

مثال: تغییر سن دانشجو

```
UPDATE Student
SET Age = 21
WHERE StudentID = 1;
```

دستور DELETE

برای حذف داده‌ها از جدول استفاده می‌شود.

مثال:

```
DELETE FROM Student
WHERE StudentID = 1;
```

دستور SELECT (مهم‌ترین دستور SQL)

دستور SELECT برای **بازیابی و نمایش اطلاعات** از پایگاه داده استفاده می‌شود.

مثال: نمایش همه دانشجویان

```
SELECT * FROM Student;
```

مثال: نمایش دانشجویان رشته کامپیوتر

```
SELECT Name, Age
FROM Student
WHERE Major = 'کامپیوتر';
```

دستورات مدیریت داده(DCL)

دستورات DCL برای **کنترل دسترسی کاربران** به پایگاه داده استفاده می‌شوند.

مهم‌ترین دستورات DCL:

- GRANT
- REVOKE

دستور GRANT

برای دادن مجوز به کاربران استفاده می‌شود.

مثال:

```
GRANT SELECT ON Student TO User1;
```

دستور REVOKE

برای گرفتن مجوز از کاربران استفاده می‌شود.

مثال:

```
REVOKE SELECT ON Student FROM User1;
```

ایجاد پرس‌وجوهای نمونه(Query)

پرس‌وجو به عملیاتی گفته می‌شود که برای استخراج اطلاعات خاص از پایگاه داده انجام می‌شود. پرس‌وجوها معمولاً با دستور SELECT نوشته می‌شوند.

مثال ۱: دانشجویان بالای ۲۰ سال

```
SELECT Name
FROM Student
WHERE Age > 20;
```

مثال ۲: مرتب‌سازی نتایج

```
SELECT Name, GPA
FROM Student
ORDER BY GPA DESC;
```

مثال ۳: شمارش تعداد دانشجویان

```
SELECT COUNT(*)
FROM Student;
```

اهمیت SQL در پیاده‌سازی پایگاه داده

SQLزبان اصلی ارتباط با پایگاه داده است و تقریباً تمام عملیات مدیریتی و کاربردی از طریق آن انجام می‌شود. بدون تسلط برSQL ، استفاده عملی از پایگاه داده امکان‌پذیر نیست.

در محیط‌های واقعی مانند دانشگاه، بانک یا فروشگاه اینترنتی، تمام ثبت‌ها، گزارش‌ها و تحلیل‌ها مبتنی بر دستورات SQL هستند.

جمع‌بندی

در این فصل با زبان SQL استاندارد آشنا شدیم و دیدیم که چگونه می‌توان با استفاده از دستورات DDL ساختار پایگاه داده را ایجاد کرد، با DML داده‌ها را مدیریت نمود و با DCL امنیت و سطح دسترسی کاربران را کنترل کرد. همچنین با نمونه‌هایی از پرس‌وجوهای کاربردی، نحوه استخراج اطلاعات از پایگاه داده را یاد گرفتیم.

تمرین‌های پایان فصل هشتم

۱. SQL چیست و چه مزیتی نسبت به زبان‌های برنامه‌نویسی دارد؟

۲. یک جدول برای «درس» طراحی کرده و آن را با دستور CREATE ایجاد کنید.

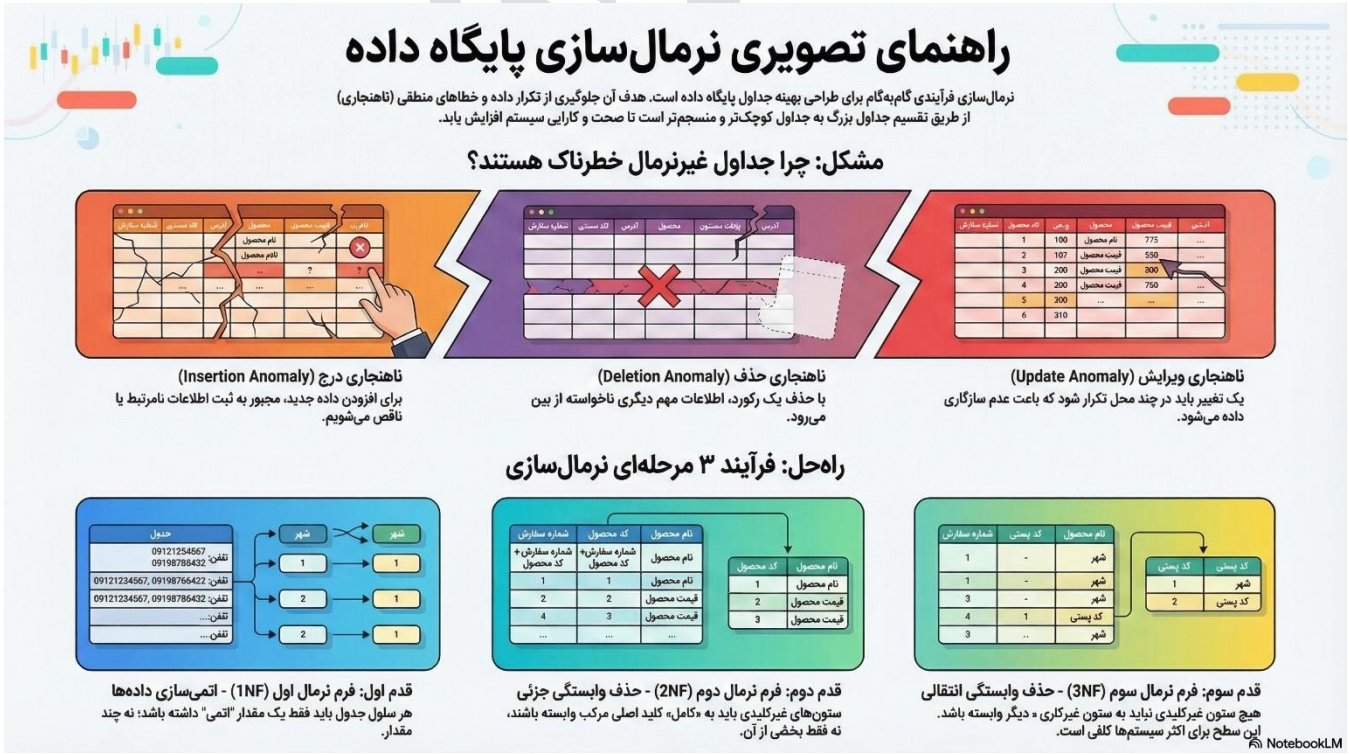
۳. دو رکورد به جدول دانشجو اضافه کنید.

٤. پرس‌وجویی بنویسید که دانشجویان با معدل بالاتر از ۱۷ را نمایش دهد.

٥. تفاوت INSERT و UPDATE را توضیح دهید.

فصل نهم

نرمال‌سازی پایگاه داده
(Normalization)



مقدمه

در طراحی پایگاه داده، تنها دانستن SQL یا ایجاد جدول کافی نیست. اگر ساختار جداول به‌درستی طراحی نشده باشد، پایگاه داده در عمل دچار مشکلات جدی خواهد شد. این مشکلات ممکن است در ابتدا قابل مشاهده نباشند، اما با افزایش حجم داده‌ها و تعداد کاربران، به‌تدریج باعث بروز خطا، کندی سیستم و حتی از بین رفتن صحت اطلاعات می‌شوند.

نرمال‌سازی روشی علمی و مرحله‌به‌مرحله برای **طراحی منطقی جداول** است که کمک می‌کند داده‌ها به شکلی ذخیره شوند که کمترین تکرار، بیشترین انسجام و بالاترین صحت را داشته باشند. در واقع نرمال‌سازی پلی است بین **مدل مفهومی (ER)** و **پیاده‌سازی واقعی پایگاه داده.**

نرمال‌سازی چیست؟

نرمال‌سازی فرآیندی است که طی آن یک جدول بزرگ و پیچیده به چند جدول کوچک‌تر، ساده‌تر و منطقی‌تر تقسیم می‌شود؛ به‌گونه‌ای که:

- هر داده فقط در یک محل ذخیره شود
- هر جدول یک مفهوم مشخص را نمایش دهد
- تغییرات داده باعث بروز خطا در سایر بخش‌ها نشود

نرمال‌سازی بر اساس **وابستگی‌های تابعی** بین داده‌ها انجام می‌شود و هدف آن حذف وابستگی‌های نادرست یا غیرضروری است.

ناهنجاری‌های داده – (Data Anomalies)

اگر نرمال‌سازی انجام نشود، جدول‌ها دچار ناهنجاری می‌شوند. ناهنجاری یعنی وضعیتی که انجام یک عملیات ساده باعث بروز مشکل منطقی در داده‌ها شود.

ناهنجاری درج(Insertion Anomaly)

گاهی برای درج یک داده جدید، مجبور می‌شویم اطلاعاتی را وارد کنیم که هنوز وجود ندارند یا معنی ندارند.

مثال:
اگر اطلاعات دانشجو و درس در یک جدول باشند، برای ثبت یک درس جدید باید حتماً یک دانشجو هم وجود داشته باشد.

ناهنجاری حذف(Deletion Anomaly)

با حذف یک داده، ممکن است اطلاعات مهم دیگری نیز به‌طور ناخواسته از بین برود.

مثال:
اگر آخرین دانشجویی که یک درس را گرفته حذف شود، اطلاعات آن درس نیز از بین می‌رود.

ناهنجاری ویرایش(Update Anomaly)

برای تغییر یک داده، باید آن را در چندین محل ویرایش کنیم و اگر یکی از آن‌ها فراموش شود، ناسازگاری ایجاد می‌شود.

مثال:
نام یک درس در چند سطر تکرار شده و تغییر آن باید در همه سطرها انجام شود.

هدف از نرمال‌سازی

نرمال‌سازی با اهداف زیر انجام می‌شود:

- حذف تکرارهای غیرضروری
- جلوگیری از ناسازگاری اطلاعات
- افزایش دقت و اعتماد به داده‌ها
- ساده‌سازی عملیات درج، حذف و ویرایش
- افزایش عمر و توسعه‌پذیری پایگاه داده

به بیان ساده، نرمال‌سازی کمک می‌کند پایگاه داده **منطقی‌تر فکر کند.**

فرم اول نرمال – (1NF)

مفهوم فرم اول نرمال

فرم اول نرمال روی **اتمی بودن داده‌ها** تمرکز دارد. اتمی بودن یعنی هر داده به کوچک‌ترین جزء معنی‌دار خود شکسته شود.

در 1NF:

- هیچ سلولی نباید شامل چند مقدار باشد
- ستون‌های تکرارشونده مجاز نیستند
- هر ستون یک نوع داده مشخص دارد

مثال واقعی

(قبل از ۱NF)

دروس	نام	شماره دانشجو
پایگاه داده، برنامه‌سازی	علی	۱

مشکل: ستون «دروس» شامل چند مقدار است.

بعد از تبدیل به ۱NF

درس	نام	شماره دانشجو
پایگاه داده	علی	۱
برنامه‌سازی	علی	۱

اکنون جدول در فرم اول نرمال قرار دارد.

فرم دوم نرمال – (2NF)

مفهوم فرم دوم نرمال

فرم دوم نرمال زمانی مطرح می‌شود که **کلید مرکب** وجود داشته باشد. هدف آن حذف **وابستگی جزئی** است.

وابستگی جزئی یعنی:

یک ویژگی غیرکلیدی فقط به بخشی از کلید وابسته باشد، نه به کل کلید.

مثال جدول مشکل‌دار

نام درس	نام دانشجو	کد درس	شماره دانشجو

کلید مرکب: (شماره دانشجو، کد درس)

- نام دانشجو ← وابسته به شماره دانشجو
- نام درس ← وابسته به کد درس

این جدول در ۲NF نیست.

تجزیه به فرم دوم نرمال

جدول دانشجو

|شماره دانشجو | نام دانشجو|

جدول درس

|کد درس | نام درس|

جدول انتخاب واحد

|شماره دانشجو | کد درس|

فرم سوم نرمال – (3NF)

مفهوم فرم سوم نرمال

فرم سوم نرمال به دنبال حذف **وابستگی انتقالی** است.

وابستگی انتقالی یعنی:

ویژگی غیرکلیدی به ویژگی غیرکلیدی دیگر وابسته باشد.

مثال وابستگی انتقالی

|شماره دانشجو | نام | کد رشته | نام رشته|

وابستگی‌ها:

- کد رشته ⟵ شماره دانشجو
- نام رشته ⟵ کد رشته

پس:

نام رشته ⟵ شماره دانشجو (به‌صورت غیرمستقیم)

تبدیل به فرم سوم نرمال

جدول دانشجو

|شماره دانشجو | نام | کد رشته|

جدول رشته

|کد رشته | نام رشته|

تا کجا نرمال‌سازی کنیم؟

در اغلب سیستم‌های واقعی:

- نرمال‌سازی تا **فرم سوم نرمال (3NF)** کاملاً کافی است
- فرم‌های بالاتر BCNF)، ۴NF و (... بیشتر جنبه تخصصی دارند

در سطح کاردانی، تسلط کامل بر **۱NF**، **۲NF** و **۳NF** کاملاً کافی و ضروری است.

جمع‌بندی نهایی

نرمال‌سازی یکی از مهم‌ترین مباحث طراحی پایگاه داده است که مستقیماً بر کیفیت، دقت و عملکرد سیستم تأثیر می‌گذارد. با استفاده از فرم‌های نرمال، می‌توان جداولی طراحی کرد که از تکرار داده، خطاهای منطقی و ناهنجاری‌های عملیاتی جلوگیری کنند.

تمرین‌های تکمیلی فصل نهم

۱. چرا نرمال‌سازی قبل از پیاده‌سازی SQL اهمیت دارد؟

۲. تفاوت وابستگی جزئی و انتقالی را با مثال توضیح دهید.

۳. یک جدول غیرنرمال طراحی کرده و آن را تا ۳ NF تبدیل کنید.

٤. چرا فرم اول نرمال پایه تمام فرم‌های بعدی است؟

٥. آیا همیشه نرمال‌سازی کامل بهترین انتخاب است؟ چرا؟