مدیریت پایگاه داده

پایهٔ دوازدهم
دورهٔ دوم متوسطه

شاخهٔ کاردانش

زمینهٔ صنعت

گروه تحصیلی: برق و رایانه

رشتهٔ مهارتی: برنامه‌نویسی پایگاه داده

نام استاندارد مهارتی مبنا: کاربر بانک اطلاعاتی SQL , Access

کد استاندارد متولی : ۰ـ۸۴/۸۰/۱/۳/۲

عنوان و نام پدیدآور : مدیریت پایگاه داده [کتابهای درسی] شاخهٔ کاردانش، زمینهٔ صنعت .../ مؤلف حبیب فروزنده‌دهکردی:
وزارت آموزش و پرورش سازمان پژوهش و برنامه‌ریزی آموزشی.
مشخصات نشر : تهران: شرکت چاپ و نشر کتاب‌های درسی ایران.
مشخصات ظاهری : ۳۳۸ص.
شابک : ۳-۲۲۱۰-۰۵-۹۶۴-۹۷۸
وضعیت فهرست نویسی : فیپا
یادداشت : کتاب حاضر استاندارد مهارت مبنا: کاربر بانک اطلاعاتی SQL , Access کد استاندارد متولی ۰-۸۴/۸۰/۱/۳/۲
موضوع : پایگاه‌های اطلاعاتی
شناسه افزوده : فروزنده، حبیب، ۱۳۶۰ -
شناسه افزوده : سازمان پژوهش و برنامه‌ریزی آموزشی
رده‌بندی کنگره : ۱۳۸۹/QA۷۶ ۳۷ف۲پ/ ۹
رده‌بندی دیویی : ۰۴۹۲ک/۳۷۳
شماره کتابشناسی ملی : ۲۰۸۴۰۵۳

وزارت آموزش و پرورش
سازمان پژوهش و برنامه‌ریزی آموزشی

نام کتاب : مدیریت پایگاه داده ـ ۳۱۲۱۴۹
پدیدآورنده : سازمان پژوهش و برنامه‌ریزی آموزشی
مدیریت برنامه‌ریزی درسی و تألیف : دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش
شناسه افزوده برنامه‌ریزی و تألیف : حبیب فروزنده‌دهکردی (مؤلف)
مدیریت آماده‌سازی هنری : ادارهٔ کلّ نظارت بر نشر و توزیع مواد آموزشی
شناسه افزوده آماده‌سازی : امید باوی (صفحه‌آرا، طراح جلد)
نشانی سازمان : تهران : خیابان ایرانشهر شمالی ـ ساختمان شمارهٔ ۴ آموزش و پرورش (شهیدموسوی)
تلفن : ۹ـ۸۸۸۳۱۱۶۱ ، دورنگار : ۸۸۳۰۹۲۶۶، کد پستی : ۱۵۸۴۷۴۷۳۵۹
وب‌گاه : www.chap.sch.ir و www.irtextbook.ir
ناشر : شرکت چاپ و نشر کتاب‌های درسی ایران : تهران ـ کیلومتر ۱۷ جادهٔ مخصوص کرج ـ خیابان ۶۱ (دارو پخش)
تلفن : ۵ـ۴۴۹۸۵۱۶۱، دورنگار : ۴۴۹۸۵۱۶۰، صندوق پستی : ۱۳۹ـ۳۷۵۱۵
چاپخانه : شرکت چاپ و نشر کتاب‌های درسی ایران «سهامی خاص»
سال انتشار و نوبت چاپ : چاپ اوّل ۱۳۹۷

شابک ۳-۲۲۱۰-۰۵-۹۶۴-۹۷۸ ISBN 978- 964-05-2210-3

از شماست که مردان و زنان بزرگ تربیت می شود. شما باید تحصیل کوشش کنید که برای فضایل اخلاقی، فضایل اعلای مجهز شوید. شما برای آتیه مملکت با جوانان نیرومند تربیت کنید. دامان شما یک مدرسه ای است که در آن جوانان بزرگ تربیت می شوند. شما فضایل تحصیل کنید تا کودکان شما در دامان شما به فضیلت برسند.

امام خمینی (رحمه الله علیه)

فهرست مطالب

فصل ۱

سپاس بیکران خداوند یکتا را که به انسان نعمت اندیشیدن عطا فرمود تا در سایه این نعمت بزرگ، هر روز برگی تازه از کتاب دانستن و آموختن را ورق بزند.

دورانـــی را که در آن زندگی می‌کنیم، «عصر ارتباطات و اطلاعات» نام‌گذاری کرده‌اند چون میزان تولید و مبادله اطلاعات آن‌قدر زیاد شـــده که تصور دنیا بدون ابزارهای تولیـــد، مدیریت و انتقال اطلاعات تقریباً غیرممکن است. در گذشته‌ای نه چندان دور، همه اطلاعات موردنیاز انسان‌ها، در یک محل مشخص تولید و نگه‌داری می‌شـــد؛ افراد فقط با مراجعه به بانکی که در آن حساب باز کرده بودند می‌توانستند وجه موردنظر را برداشـــت یا واریز کنند و انجام کاری غیر از این بســـیار دشوار بود. اما امروزه به لطف گسترش شبکه‌های رایانه‌ای و تولید نرم‌افزارهای قدرتمند، همه این کارها با چند کلیک ساده انجام می‌شود.

در پشـــت این کار ساده، دانشمندانی از رشـــته‌های گوناگون مثل الکترونیک، مخابرات، رایانه، ریاضی و ... ســـال‌ها وقت خود را صرف کرده‌اند تا دوران ما را به عصر ارتباطات و اطلاعات تبدیل کنند؛ روزگاری که شـــما پیغامی را تایپ می‌کنید و با یک کلیک آن را به دست گیرنده‌ای می‌رسانید که هزاران کیلومتر دورتر از شما نشسته است.

در این پیشـــرفت های خیره کننده، قطعاً نقش «سیســـتم‌های مدیریت پایگاه داده» حیاتی و برجسته اســـت. پایگاه داده، بانک اطلاعاتی و بانک داده همگی عنوان‌هایی تقریباً مشـــابه هستند که کارِ مهم آن‌ها، ذخیره کردن انبوهی از داده‌ها، بازگرداندن نتایج موردنظر کاربر در سریع‌ترین زمان ممکن و در برخی موارد، انجام پردازش‌هایی روی داده‌های ذخیره شده است.

شـــما با خواندن این کتاب، قدم در مسیری می‌گذارید که در انتهای آن به یک کاربرِ بانک‌های اطلاعاتی Access و SQL Server تبدیل خواهید شد و با کامل‌کردن دانسته‌های خود می‌توانید به عنوان یک «طراح» یا «مدیر» پایگاه داده در بازار وسیع فن‌آوری اطلاعات مشغول به کار شوید.

ســـعی کنید توضیحات تئوری را به دقت دنبال کنید، کارهای عملـــی را حداقل یک‌بار روی رایانه خود امتحـــان نمایید و با گســـترش دادن مثال‌های این کتاب، توانایی خود را در کار با این دو سیســـتم معروف مدیریت پایگاه داده به نقطه مطلوبی برسانید.

از آن‌جا که تقویت زبان انگلیسی، کمک فراوانی به شما در جهت یافتن پاسخ سؤالات و مراجعه به منابع و مراجع آموزشی خواهد کرد، توصیه می‌کنم متون انگلیسی انتهای هر فصل را به دقت مطالعه کرده و معنی کلمات کلیدی آن را به خاطر بسپارید.

از همه اســـاتید محترم و دانش آموزان عزیز تقاضا دارم نظرات، انتقادات و پیشنهادهای خود را راجع به محتوای این کتاب به نشـــانی habibfd@yahoo.com ارســـال نمایند تا در ویرایش‌های بعدی این کتاب از آن‌ها بهره ببرم.

موفق باشید

حبیب فروزنده – بهار ۸۹

فصل اول

فصل اول : مفاهیم پایگاه داده

پیش از شروع کار با هر برنامه‌ای، باید ضرورت استفاده از نرم‌افزار و مراحل طی شده در ایجاد و تکامل آن را فرابگیرید تا بتوانید برای انجام هر کاری، ابزارِ مناسب را انتخاب و از همه قابلیت‌های آن استفاده نمایید.

در این فصل با مفاهیم مقدماتی پایگاه داده، تاریخچه شکل‌گیری و استفاده از آن و نیز روش‌های طراحی یک پایگاه داده آشنا خواهید شد. مطالب این فصل در مورد هر دو سیستم مدیریت پایگاه داده یعنی Access و SQL Server صدق می‌کند.

۱-۱ آشنایی با انواع و ساختار پایگاه داده

۱-۱-۱ معرفی پایگاه داده

پایگاه داده[1] را می‌توان یکی از انواع «سیستم‌های ذخیره و بازیابی اطلاعات» محسوب کرد. در این نوع سیستم‌ها به کاربر اجازه داده می‌شود داده‌های موردنظر خود را ذخیره نموده و ضمن انجام پردازش روی آن‌ها، عملیات بازیابی، یعنی استفاده از اطلاعات ذخیره شده را انجام دهد.

سال‌ها پیش و قبل از تولید مفهومی به نام پایگاه داده، برای ذخیره کردن اطلاعات از سیستم فایلینگ[2] استفاده می‌شد. در این سیستم، اطلاعات وارد شده در برنامه، درون یک یا چند فایل نوشته می‌شد و برنامه‌نویس مجبور بود همه الگوریتم‌های موردنیاز برای مرتب‌سازی یا جستجوی داده‌ها را درون نرم‌افزار پیاده‌سازی کند.

برای درک دلایلی که منجر به کنار گذاشتن سیستم فایلینگ و عرضه سیستم‌های مدیریت پایگاه داده شد ابتدا باید معایب فایلینگ را بررسی کنیم.

الف) در سیستم فایلینگ، برنامه‌نویس مجبور است برای هر کاربرد خاص، فایل‌های مجزایی را طراحی و پیاده‌سازی کند و از آن‌جا که تمام عملیات ذخیره‌سازی و بازیابی توسط کدهای برنامه کنترل می‌شود، هر تغییر کوچکی در برنامه مستلزم بازنویسی کدهای نرم‌افزار است. به این ترتیب تولید یک نرم‌افزار به حجم بالایی از کدنویسی نیاز دارد.

ب) سیستم دچار افزونگی در ذخیره‌سازی اطلاعات می‌شود؛ به این معنی که مثلاً نشانی دانشجو یک‌بار در فایل‌های نرم‌افزار «اداره رفاه» و بار دیگر در فایل‌های نرم‌افزار «اداره آموزش» ثبت می‌شود و همین مسأله «تکرار در ذخیره‌سازی» را به دنبال خواهد داشت.

ج) جدا بودن سیستم‌های ذخیره‌سازی اطلاعات از هم، مشکل دیگری را به دنبال دارد که آن را احتمال «ناسازگاری داده‌ها» می‌نامیم. برای مثال اگر فردی در سیستم پرسنلی با نام «امیر محمودی» و در سیستم آموزشی به اسم «امیر محمودی دهکردی» معرفی شده باشد، داده‌های مربوط به این فرد در دو زیرسیستم، ناسازگار تلقی خواهند شد.

د) به دلیل وجود فایل‌های متعدد، اعمال استاندارد در شیوه ذخیره‌سازی داده‌ها بسیار دشوار است و به عنوان مثال ممکن است برای ذخیره‌سازی نام خانوادگی در فایل‌های مختلف، از اندازه‌های متفاوتی استفاده شود.

1-DataBase

2 . Filing

با وجود این که هنوز هم برای تولید برنامه‌های کوچک از سیستم فایلینگ استفاده می‌شود اما تولید نرم‌افزارهای یکپارچه که همه نیازهای یک سازمان را رفع کند و برای مثال شامل سیستم پرسنلی، مالی و آموزشی یک دانشگاه باشد نیازمند استفاده از پایگاه داده است. بهره‌گیری از یک «سیستم مدیریت پایگاه داده» باعث ایجاد تمرکز در ذخیره‌سازی اطلاعات می‌شود و در صورت طراحی صحیح، افزونگی و ناسازگاری اطلاعات را از بین می‌برد. به عنوان مثال، مشخصات پرسنلی تنها یک‌بار و با رعایت استانداردی مشخص ذخیره خواهد شد. علاوه بر این، ذخیره‌سازی و بازیابی اطلاعات بر عهده این سیستم گذاشته می‌شود و برنامه‌نویس مجبور نیست مانند روش فایلینگ، برای ایجاد یا تغییر فایل‌ها، اضافه کردن رکورد و جستجو و مرتب‌سازی، کدهای اضافی بنویسد.

با توجه به مطالب ذکر شده می‌توان پایگاه داده را به این صورت تعریف کرد : «مجموعه‌ای سازمان‌یافته و بدون افزونگی از داده‌های مرتبط با هم که در چارچوب یک مدل داده‌ای و تحت کنترل یک سیستم متمرکز قرار دارند».

۲-۱-۱ سیستم مدیریت پایگاه داده

در تعریف پایگاه داده از یک «سیستم متمرکز» نام بردیم که پایگاه داده را تحت کنترل دارد. این سیستم متمرکز را «سیستم مدیریت پایگاه داده» یا به اختصار DBMS[1] می‌نامیم؛ یعنی دروازه‌ای که هرگونه دسترسی به داده‌ها و تغییر اطلاعات باید از طریق آن صورت گیرد. در واقع، پایگاه داده و فایل‌های آن تحت کنترل نرم‌افزار DBMS هستند و حتی برنامه‌نویس، درخواست‌های خود را از طریق کد به آن ارسال نموده و تنها این سیستم است که می‌داند چگونه داده‌ها را درون فایل‌های پایگاه داده ذخیره و بازیابی کند و در صورت نیاز تغییر دهد.

در مجموع، سیستم مدیریت پایگاه داده نسبت به سیستم‌های ذخیره و بازیابی پیش از خود دارای مزایا و معایب زیر است:

مزایا :

الف) ذخیره‌سازی و پردازش داده‌ها به صورت متمرکز انجام می‌شود.

ب) تاحد زیادی از افزونگی داده‌ها جلوگیری می‌شود؛ برای نمونه مشخصات پرسنلی افراد مانند نام، نام خانوادگی و ... تنها در یک بخش از پایگاه داده ذخیره خواهد شد.

ج) امکان به اشتراک گذاشتن داده‌ها و استفاده هم‌زمان چند برنامه کاربردی از پایگاه داده وجود دارد.

---

1 . DataBase Management System

د) اعمال محدودیت‌های امنیتی برای کنترل دسترسی‌ها امکان‌پذیر است.

ه) صحت داده‌ها (مستقل از برنامه‌های کاربردی) تضمین می‌شود. برای مثال اگر در برنامه کاربردی روشی برای صحت ورود تاریخ درنظر نگرفته شده باشد و تاریخ ۱۳۸۹/۱۰/۳۲ قابل ورود باشد، پایگاه داده از ثبت آن جلوگیری خواهد کرد.

و) پیاده‌سازی برنامه‌های کاربردی جدید ساده‌تر است.

معایب

الف) طراحی سیستم‌های پایگاه داده پیچیده و زمان‌بر است.

ب) هزینه قابل توجهی صرف خرید نرم‌افزار و نصب تجهیزات سخت‌افزاری می‌شود.

درحال حاضر Access، MySQL ، SQL Server و Oracle از سیستم‌های مدیریت پایگاه داده پرکاربرد در دنیا محسوب می‌شوند و افراد و سازمان‌ها بسته به میزان کارایی و هزینه‌ای که می‌خواهند پرداخت کنند، یکی از آن‌ها را به عنوان منبع ذخیره‌سازی و مدیریت داده‌های خود انتخاب می‌نمایند.

این نرم‌افزارهای قدرتمند و پیچیده روزبه‌روز در حال تکامل هستند و در برخی از آن‌ها علاوه بر امکانات مدیریت داده‌ها، ابزارهایی برای ساخت فرم‌های ورود اطلاعات و نیز تولید گزارش وجود دارد. هر یک از DBMSهای مذکور دارای ویژگی‌هایی به صورت زیر هستند :

Microsoft Access : برای حجم اطلاعات متوسط و تعداد کاربران کم طراحی شده و امکانات خوبی برای طراحی فرم‌های ورود اطلاعات و ساخت گزارش دارد.

Microsoft SQL Server : این محصول شرکت مایکروسافت می‌تواند از تعدادی زیادی کاربر و حجم بسیار بالایی از اطلاعات پشتیبانی کند و به دلیل هماهنگی با سایر محیط‌های تولید شده توسط مایکروسافت (مثل پلت‌فرم دات‌نت[1]) طرفداران زیاد دارد. در نسخه‌های جدید این نرم‌افزار، سرویس گزارش‌سازی قدرتمندی هم تعبیه شده است.

Oracle : مدیریت تعداد زیادی کاربر و حجم فوق‌العاده بالایی از اطلاعات در این نرم‌افزار امکان‌پذیر است اما قیمت بالا و پیچیدگی‌های نصب و راه‌اندازی، استفاده از آن را محدود به سازمان‌های بزرگ کرده است.

1 . NET Platform

۳-۱-۱ ساختار پایگاه داده

وظیفه یک سیستم مدیریت پایگاه داده این است که داده‌ها را از کاربر گرفته و به روش صحیحی روی حافظه رایانه ذخیره کند؛ به گونه‌ای که بازیابی آن‌ها در سریع‌ترین شکل امکان‌پذیر باشد. علاوه بر این برای کنترل دسترسی به داده‌ها باید پیش‌بینی‌های لازم را صورت دهد تا افراد غیرمجاز نتوانند داده‌ها را مشاهده نموده یا تغییری در آن‌ها ایجاد کنند. این پیچیدگی، زمانی بیش‌تر می‌شود که صدها کاربر بخواهند به طور هم‌زمان و از طریق شبکه، عملیات موردنظرشان را روی داده‌ها انجام دهند.

اغلب DBMSها برای مدیریت داده‌ها و پاسخ‌گویی به درخواست‌های کاربران از معماری سه سطحی استفاده می‌کنند. منظور از معماری، توصیفی از یک سیستم است که در آن اجزاء تشکیل دهنده سیستم، ارتباط میان آن‌ها و قواعد حاکم بر طراحی نشان داده می‌شود. این سطوح ضمن مستقل بودن، به گونه‌ای با هم تعامل دارند که هدف نهایی سیستم یعنی دسترسی کاربران به داده‌های موردنیاز و مجاز محقق شود. این سطوح سه‌گانه عبارتند از :

الف) سطح خارجی[1]

ب) سطح مفهومی[2]

ج)سطح داخلی[3]

الف) سطح خارجی : کاربران پایگاه داده و نرم‌افزارهای کاربردی با این سطح سروکار دارند و هر کدام بسته به نقشی که در سیستم برای آن‌ها تعریف شده، مجاز به مشاهده و تغییر بخشی از اطلاعات هستند. اطلاعاتی که کاربر به آن‌ها دسترسی دارد، اصطلاحاً «دیدِ کاربر»[4] گفته می‌شود.

ب) سطح مفهومی : این سطح، ساختار طراحی شده برای پایگاه داده و ازتباطات بین عناصر بانک اطلاعاتی را شامل می‌شود.

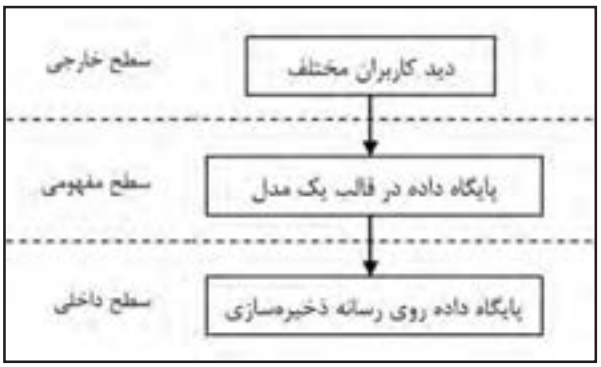
ج) سطح داخلی : نحوه ذخیره‌سازی داده‌ها روی سخت‌افزار، در این سطح تعیین می‌شود. سطح داخلی با جزییات ذخیره‌سازی فیزیکی و روش‌های فایلینگ سروکار دارد.

سطح‌بندی سیستم به این دلیل انجام شده تا به عنوان مثال، کاربری که می‌خواهد یک گزارش ساده از سیستم بگیرد، درگیر جزییات مربوط به نحوه ذخیره‌سازی داده‌ها درون فایل‌ها نشود. علاوه بر این اگر

---

1 . External Level
2 . Conceptual Level
3 . Internal Level
4 . User View

لازم شد تغییری در نحوه ذخیره‌سازی داده‌ها (سطح داخلی) ایجاد شود، دیدِ کاربران (سطح خارجی) بدون تغییر باقی بماند.



شکل ۱-۱

۱-۱-۴ مدل داده‌ای

در تعریف پایگاه داده با دو مفهوم بسیار مهم مواجه شدیم. اولین مفهوم، «مدیریت متمرکز» بود که نشان دادیم این وظیفه مدیریتی بر عهده نرم‌افزار مدیریت پایگاه داده است. دومین مفهوم «مدل داده‌ای» است که آن را در این بخش بررسی می‌کنیم.

فرض کنید از شما خواسته شده سیستم آموزشی مدرسه خود را در قالب یک پایگاه داده تعریف نمایید. برای تبدیل یک ساختار واقعی به سیستم نرم‌افزاری باید در چارچوب یک مدل کار کنید تا بتوانید عناصر، داده‌ها و مفاهیم موجود در ساختار آموزشی یک مدرسه را در محیط مجازی تعریف و در قالب یک ساختمان داده پیاده‌سازی نمایید. مثلاً عنصری مثل «دانش‌آموز»، داده‌ای مانند «نمره» یا مفهومی مثل «پیش‌نیاز بودن» را در قالب یک مدل نشان بدهید؛ به‌گونه‌ای که این تعریف‌ها در همه بخش‌های طراحی شما صادق باشد.

هم‌زمان با ایجاد مفهومی به نام پایگاه داده، روش‌هایی هم برای طراحی و مدل‌سازی داده‌ها ابداع شد که معروف‌ترین آن‌ها عبارتند از :

الف) مدل سلسله مراتبی (Hierarchical)

ب) مدل شبکه‌ای (Network)

ج) مدل شیءگرا (Object Oriented)

د) مدل رابطه‌ای (Relational)

هر یک از مدل‌های فوق دارای قواعدی برای طراحی هســـتند و در نوع خود مزایا و معایبی دارند اما در این کتاب ما صرفاً بر روی مدل رابطه‌ای تمرکز خواهیم کرد. مدل رابطه‌ای در سال ۱۹۷۰ توسط ریاضیدانی به نام کاد[1] پیشنهاد شد و در آن از منطق گزاره‌ها و نظریه مجموعه‌ها استفاده شده است. اغلب سیستم‌های مدیریت پایگاه داده که در ســـال‌های اخیر تولید شده‌اند، از جمله Access و SQL Server که مبنای کار ما در این کتاب هستند، از مدل رابطه‌ای تبعیت می‌کنند.

۲-۱ آشنایی با مفاهیم پایگاه داده رابطه‌ای

۱-۲-۱ فیلد[2]

منظور از فیلد یک قطعه داده است که «نام» و «مقدار» دارد. برای مثال وقتی می‌گوییم :

شماره تلفن = ۷۷۶۶۵۵۴۴

«شـــماره تلفن» نام فیلد را نشـــان می‌دهد و «۷۷۶۶۵۵۴۴» مقدار فیلد است. هر چند اغلب اوقات، دو مفهوم «داده» و «اطلاع» به جای هم مورد استفاده قرار می‌گیرند اما بین این دو تفاوت ظریفی وجود دارد. در مثال بالا، مقدار فیلد همان «داده» است و نام فیلد به همراه مقدار، حکم «اطلاع» را دارد؛ یعنی اگر عدد بالا را بدون نام فیلد به شما نشان دهند نمی‌توانید تشخیص بدهید چه چیزی است اما وقتی عبارت «شماره تلفن» در کنار آن قرار می‌گیرد شما اطلاع پیدا می‌کنید که عدد مذکور، یک شماره تلفن است.

یک فیلد می‌تواند نوع داده‌ای متفاوتی از فیلد دیگر داشـــته باشد؛ مثلاً «معدل» یک عدد اعشاری است اما «نشانی» باید از نوع رشته باشد.

۲-۲-۱ رکورد[3]

وقتی چند فیلد مرتبط، در کنار هم قرار می‌گیرند تشکیل یک رکورد می‌دهند.

مثلاً وقتی در مورد مشـــتری‌های یک ناشر صحبت می‌کنیم، رکوردِ هر مشتری می‌تواند شامل این فیلد باشد: کد مشتری، نام مشتری، تلفن، شهر، نشانی کامل.

بنابراین دو نمونه از رکورد مشتری‌های یک ناشر می‌تواند چیزی شبیه به این باشد :

۱۲۰۱، نشر سحر، ۷۷۶۶۵۵۴۴، تهران، خیابان انقلاب–کوچه محمدی-پلاک ۱۵

۱۲۰۶، کتابفروشی پیام، ۲۲۳۳۴۴۵۵، اصفهان، خیابان بزرگمهر–پلاک ۳۶۰

---

1 . Codd

2 . Field

3 . Record

نکته مهم این‌جاست که هر یک از فیلدهای یک رکورد می‌تواند نوع داده‌ای و طول متفاوتی داشته باشد.

۳-۲-۱ جدول[1]

وقتی چند رکورد با فیلدهای مشابه در کنار هم قرار می‌گیرند، یک جدول ایجاد می‌شود که کلیدی‌ترین مفهوم در مدل رابطه‌ای به شـــمار می‌رود. هر جدول تعدادی ستون دارد که همان فیلدها هستند و می‌تواند صفر یا میلیون‌ها ســـطر داشته باشـــد که در واقع رکوردهای جدول هستند. برای مثال اگر بخواهیم جدولی برای مشتریان یک ناشر مثال بزنیم، با ساختار شکل ۲-۱ روبرو خواهیم شد.

فیلدها

نام مشتری	تلفن	شهر	نشانی
نشر سیمرغ	۶۶۴۹۴۲۰۶	تهران	خیابان انقلاب–کوچه نوروزی-پلاک ۱۵
کتابفروشی پیام	۲۲۳۳۴۴۵	اصفهان	خیابان بزرگمهر–بازار پردیس-طبقه سوم
دانشگاه فردوسی	۶۶۵۵۶۶۴	مشهد	خیابان امام رضا(ع) – دانشگاه فردوسی
کتابفروشی پیام	۸۸۵۵۴۴۱۱	تهران	خیابان کریم‌خان- پلاک ۱۲۵

رکوردها

شکل ۲-۱

۴-۲-۱ کلید اصلی[2]

به فیلد یا مجموعه‌ای از فیلدها گفته می‌شود که باعث یکتایی رکوردهای جدول می‌شوند؛ یعنی امکان اشاره به فقط «یک رکورد» را فراهم می‌کنند. مثالی از جدول بالا می‌زنم تا مفهوم کلید اصلی را بهتر متوجه شـــوید. می‌خواهیم به سیســـتم مدیریت پایگاه داده بگوییم : «در جدول مشتری‌ها، نشانی مشتری خاصی را تغییر بده». اگر برای نشـــان دادن آن مشـــتری از فیلد «نام مشتری» استفاده کنیم، به دلیل امکان وجود نام‌های تکراری، رکورد انتخاب شـــده، یکتا[3] نخواهد بود. مثلاً «کتابفروشی پیام» در دو رکورد وجود دارد و بنابراین نمی‌تواند به تنهایی کلید اصلی باشد.

حالا اگر با اســـتفاده از فیلدهای «نام مشتری» و «شهر» یک کلید ترکیبی[4] بسازیم، وضع تا حدی بهتر می‌شود و می‌توانیم به پایگاه داده بفهمانیم که منظور ما «کتابفروشی پیام در شهر تهران» است. اما اگر چند

1 . Table
2 . Primary Key
3 . Unique
4 . Compound Key

کتابفروشی با این نام در تهران وجود داشته باشد تکلیف چیست؟

برای غلبه بر چنین وضعیتی، معمولاً یکی از فیلدهایی را که مقدار آن برای همه رکوردها منحصربه‌فرد است به عنوان کلید اصلی انتخاب می‌کنند. مثلاً در جدول اطلاعات پرسنلی یک اداره، فیلد کد ملی کارکنان کلید اصلی است چون برای هر فرد یکتاست. اگر هم در سیستمی چنین فیلدی وجود نداشته باشد، یک فیلد جدید تعریف نموده و با اختصاص شماره‌های منحصربفرد به رکوردها، آن را به کلید اصلی جدول تبدیل می‌کنند؛ کاری که ما بعداً در جدول مثال‌مان انجام خواهیم داد.

با توضیحات بالا این نکته قابل درک است که کلید اصلی نمی‌تواند خالی یا تکراری باشد.

۱-۲-۵ ویژگی‌های پایگاه داده رابطه‌ای

در یک جمع‌بندی کلی می‌توان ویژگی‌های زیر را برای پایگاه داده رابطه‌ای ذکر کرد:

الف) متداول‌ترین مدل است.

ب) بر اساس تئوری ریاضی است.

ج) داده‌ها و ارتباطات بین آن‌ها به صورت مجموعه‌ای از جداول پیاده‌سازی می‌شود.

د) هیچ جدولی سطرهای تکراری ندارد.

ه) ترتیب سطرها و ستون‌ها در یک جدول مهم نیست.

ز) ایجاد و توسعه آن آسان است.

۱-۳ ترسیم نمودارهای موجودیت-رابطه‌ای[1]

۱-۳-۱ شناخت ساختار

اولین قدم در ایجاد یک پایگاه داده استاندارد و کارآمد، بررسی ساختار واقعی موجود، شناسایی نیازهای اطلاعاتی و پردازشی آن و نیز تشخیص قوانین حاکم بر ساختار است. فرض کنید از ما خواسته شده سیستم انبارداری و فروش یک مؤسسه انتشاراتی را در قالب یک پایگاه داده اکسس پیاده‌سازی کنیم. در این ساختار، موجودیت هایی مثل کتاب، مشتری و ... وجود دارد که هر یک ویژگی‌های مختص به خود دارند. مثلاً هر کتاب دارای نام و قیمت مشخص است یا هر مشتری، نشانی، تلفن و مبلغ بدهکار یا بستانکار دارد. از سوی دیگر در این ساختار، پردازش‌هایی مثل «فروش» انجام می‌شود و این پردازش‌ها تابع قواعدی مشخص هستند. مثلاً «اگر موجودی یک کتاب کم‌تر از ۵۰ نسخه باشد، باید فروش آن متوقف شود».

---

1 . E-R (Entity-Relationship)

برای شناســـایی موجودیت ها، پردازش‌ها و قواعد حاکم بر یک ســـاختار، روش‌هایی وجود دارد که در کتاب‌های «تجزیه و تحلیل سیســـتم‌های اطلاعاتی» به صورت مفصل مورد بحث قرار گرفته است اما آن‌چه در این کتاب بررسی خواهد شد، نگاهی اجمالی به نحوه انجام این کار است.

برای کســـب اطلاع از عناصر و پردازش‌های موجود در یک ســـاختار، باید با افـــراد مطلع و به خصوص ســـفارش‌دهنده پایگاه داده و استفاده کنندگان از آن مصاحبه کنید. نکات مطرح شده توسط وی را به دقت ثبت نموده و با پرســـیدن ســـؤالاتی مناســـب، ابهام‌های موجود را برطرف نمایید. با پرسیدن این سؤال که «قصد دارید چه گزارش‌هایی از پایگاه داده دریافت کنید؟» بخش عمده‌ای از نیازهای اطلاعاتی شـــما پاسخ داده می‌شود. دقت داشته باشید که عدم اطلاع کافی از خواست سفارش‌دهنده پایگاه داده، باعث اشتباه در طراحی و نهایتاً دوباره‌کاری می‌شود.

پس از انجام این مرحله که «بررسی نیازمندی‌ها» نامیده می‌شود، باید به سراغ مرحله طراحی بروید. با استفاده از اطلاعات جمع‌آوری شده می‌توانید یک مدل‌سازی معنایی از ساختار ایجاد کنید.

۲-۳-۱ مفاهیم طراحی

روش‌های مختلفی برای مدل‌ســـازی معنایی وجود دارد اما در میان آن‌ها، مدل موجودیت-رابطه (E-R) رایج‌تر است. این مدل در سال ۱۹۷۶ توسط فردی به نام چِن[1] ابداع گردیده و تاکنون چندین بار توسط خود وی و دیگران اصلاح شده است.خروجی این مرحله، رسم نمودارهای E-R است اما برای رسم این نمودارها، ابتدا باید با چند مفهوم کلیدی آشنا شوید.

۱-۲-۳-۱ محیط عملیاتی[2]

به محیطی گفته می‌شـــود که می‌خواهیم در آن یک سیســـتم ذخیره و بازیابی اطلاعات طراحی کنیم. مؤسسه انتشاراتی، دانشگاه، فرودگاه و ... نمونه‌هایی از محیط عملیاتی هستند.

۲-۲-۳-۱ موجودیت[3]

موجودیت، مفهوم یا شـــیئی اســـت که در محیط عملیاتی به طور مجزا قابل شناســـایی باشد. در مثال مؤسســـه انتشاراتی، کتاب و مشتری دو موجودیت این ساختار هستند. در یک محیط عملیاتی ممکن است موجودیت‌هایی زیادی وجود داشـــته باشـــد امـــا آن‌هایی برای ما اهمیت دارند که می‌خواهیم در سیســـتم

---

1 . Chen

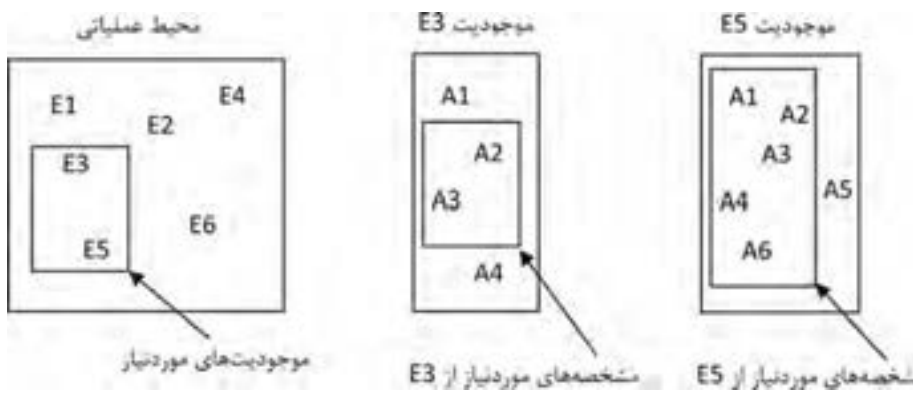
2 . Operation Environment

3 . Entity

پیاده‌ســـازی کنیم. برای نمونه در یک مؤسسه انتشاراتی موجودیتی به نام «کارکنان» هم قابل تعریف است اما چون قصد طراحی سیســـتم «انبارداری و فروش» را داریم، این موجودیت برای ما اهمیتی ندارد. طبیعی اســـت اگر در این پایگاه داده، زیرسیستم «حقوق و دســـتمزد» هم تعریف شده بود، موجودیت «کارکنان» برای ما حایز اهمیت بود.

۳-۲-۳-۱ مشخصه‌های موجودیت[1]

هر موجودیت، مشخصه‌هایی دارد که در طراحی نهایی تبدیل به «فیلد» می‌شوند. برای مثال موجودیت کتاب، دارای مشخصه‌های کد، نام، نام ناشر، قیمت و ... است. نکته‌ای که در مورد موجودیت‌های یک محیط عملیاتی گفتیم در مورد مشخصه‌های یک موجودیت هم صادق است؛ ممکن است موجودیتی ده‌ها مشخصه داشـــته باشد اما فقط مشخصه‌هایی در طراحی لحاظ می‌شوند که در پایگاه داده موردنیاز باشند. شکل ۳-۱ این مفهوم را به خوبی نمایش می‌دهد.



شکل ۳-۱

در محیط عملیاتی «مؤسسه انتشاراتی» این موجودیت‌ها و مشخصه‌ها وجود دارند :

موجودیت‌ها: کتاب، مشتری

مشخصه‌های موجودیت کتاب: کد کتاب، عنوان کتاب، ناشر، سال انتشار و قیمت.

مشخصه‌های موجودیت مشتری: کد مشتری، نام مشتری، تلفن، نشانی، مبلغ بدهکار یا بستانکار

1 . Entity Attributes

۴-۲-۳-۱ رابطه[1]

ارتباط میان موجودیت‌ها را «رابطه» می‌گوییم. برای مثال در سیستمی در حال مدل‌سازی آن هستیم، «مشـــتری»، «کتـــاب» را «خریداری» می‌کند یعنی بین دو موجودیت نوعـــی «عمل» اتفاق می‌افتد که در این‌جا «خرید» است. بنابراین :

- در هر رابطه، یک یا چند موجودیت حضور دارند.
- رابطه، یک مفهوم یا عمل‌کرد است و هنگام تعریف سیستم از آن به صورت «مصدر» یاد می‌شود : مثل خرید کردن، ثبت‌نام شدن، استخدام نمودن و ...
- رابطه می‌تواند مشخصه‌های مخصوص به خود داشته باشد مثل تاریخ خرید، مجموع خرید و ...
- هر رابطه ماهیت[2] مشـــخصی دارد. مثلاً «یک» مشـــتری می‌تواند «چند» کتاب خریداری کند بنابراین رابطه موجودیت مشتری و موجودیت کتاب، «یک به چند» است.
- تعریـــف رابطه منجر به تولید مفهومی به نام «کلید خارجی»[3] می‌شـــود. با این مفهوم در ادامه مطالب همین فصل آشنا خواهید شد.

۵-۲-۳-۱ انواع ماهیت در رابطه:

رابطه بین موجودیت‌ها در یک سیســـتم دارای ماهیتی اســـت که بســـته به نحوه تعامل موجودیت‌ها با یک‌دیگر می‌تواند یکی از سه حالت زیر باشد.

الـــف) یک به یک (1:1)[4] : در این نوع رابطه، یک نمونه از موجودیت اول فقط با یک نمونه از موجودیت دوم رابطه دارد. مثلاً هر مؤسسه انتشاراتی فقط یک مدیر دارد.

ب) یک به چند (N:1)[5] : اگر موجودیت اول بتواند با چند نمونه از موجودیت دوم رابطه داشـــته باشد، ماهیت رابطه، یک به چند می‌شود. مثلاً هر مشتری می‌تواند چند نمونه کتاب بخرد.

ج) چند به چند (M:N)[6] : در این نوع رابطه، یک نمونه از موجودیت اول با چند نمونه از موجودیت دوم رابطه دارد و عکس آن صادق است یعنی یک نمونه از موجودیت دوم هم می‌تواند با چند نمونه از موجودیت اول رابطه داشته باشد. مثلاً هر مشتری می‌تواند چند نمونه کتاب بخرد و هر نمونه کتاب هم می‌تواند توسط چند مشتری خریداری شود.

---

1 . Relationship
2 . Cardinality
3 . Foreign Key
4 . One To One
5 . One To Many
6 . Many To Many

مثال ۱ : در محیط عملیاتی دانشگاه، نمونه‌ای از ماهیت رابطه‌ها به صورت زیر است :

الف) رابطه یک به چند : هر اســـتاد عضو یک دانشـــکده است اما هر دانشکده می‌تواند چند استاد داشته باشد.

ب) رابطه چند به چند : دانشجو در هر ترم، چندین درس را ثبت‌نام می‌کند و یک درس به‌وسیله چندین دانشجو انتخاب می‌شود.

مثال ۲ : در محیط عملیاتی بیمارستان، رابطه‌ها می‌توانند ماهیتی به شکل زیر داشته باشند :

الف) رابطه یک به یک : هر بیمارِ بستری یک تخت دارد و هر تخت فقط متعلق به یک بیمار است.

ب) رابطه یک به چند : هر پرستار در یک بخش کار می‌کند و هر بخش می‌تواند چندین پرستار داشته باشد.

ج) رابطه چند به چند : هر پزشک چندین بیمار دارد و هر بیمار می‌تواند چندین پزشک داشته باشد.

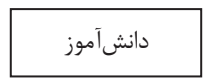
۳-۳-۱ مراحل طراحی

۱-۳-۳-۱ رسم نمودار E-R

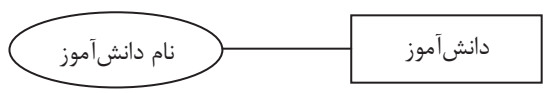
رســـم نمودارهای موجودیت-رابطه، روشی برای نشان دادن رابطه میان موجودیت‌ها و کارکرد و ماهیت این روابط است. به این ترتیب یک مدل‌سازی معنایی از داده‌ها ایجاد می‌شود و ما یک گام دیگر به طراحی نهایی پایگاه داده نزدیک می‌شویم.

نمادهایی که برای رسم نمودار E-R از آن‌ها استفاده می‌کنیم به صورت زیر هستند:

الف) موجودیت: یک مستطیل که نام موجودیت درون آن نوشته می‌شود.



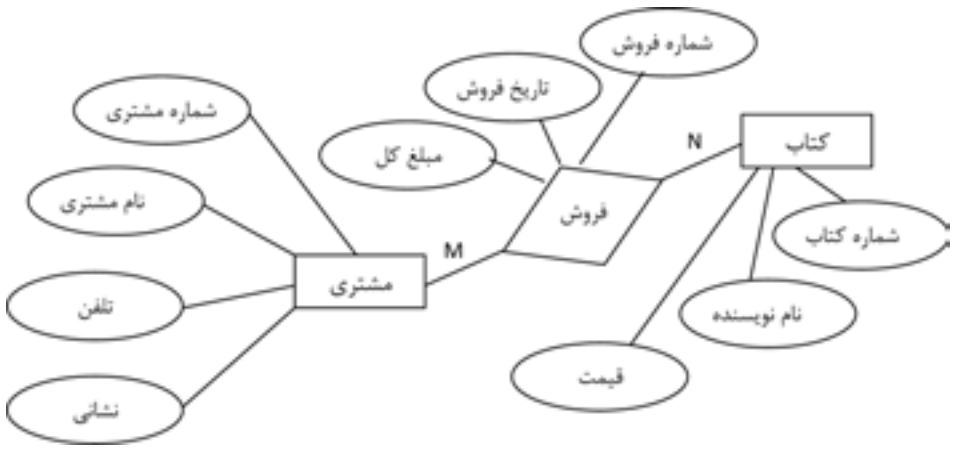
ب) مشـــخصه موجودیت و رابطه: یک بیضی که نام مشـــخصه در آن قرار می‌گیرید و با یک خط به موجودیت یا رابطه وصل می‌شود.



ج) رابطه: خطی صاف که موجودیت‌ها را به هم وصل می‌کند و در دو سر آن ماهیت رابطه قید می‌شود.

د) کارکرد رابطه: یک لوزی که روی خط رابطه قرار می‌گیرد و کارکرد رابطه درون آن ذکر می‌شود.

با این توضیحات، نمودار E-R سیستم فروش یک مؤسسه انتشاراتی به این صورت است :



شکل ۴-۱

۲-۳-۳-۱ تبدیل موجودیت‌ها به جدول

پس از رسم نمودارهای E-R نوبت به طراحی جداول می‌رسد. در پایگاه داده‌های رابطه‌ای، هر موجودیت و رابطه تبدیل به یک جدول می‌شـــود و مشخصه‌های موجودیت یا رابطه، فیلدهای آن را تشکیل می‌دهند. بنابراین در مثالی که دنبال می‌کنیم، در نگاه اول جدول‌های زیر قابل طراحی هستند؛ البته هنگام پیاده‌سازی نهایی باید نام جداول و فیلدها به زبان انگلیسی درج شوند اما چون فعلاً در مرحله طراحی مقدماتی هستیم، نام‌گذاری‌ها را به صورت فارسی انجام داده‌ایم.

جدول کتاب			
شماره کتاب	نام کتاب	نام نویسنده	قیمت

جدول مشتری				
شماره مشتری	نام مشتری	تلفن	نشانی	مبلغ اعتبار

در جدول مشـــتری، فیلدی به نام نشانی وجود دارد که می‌تواند شامل نام استان، نام شهر، نشانی دقیق و کدپستی باشد. بهتر است این نوع فیلدهای مرکب را به اجزاء سازنده آن تقسیم نموده و برای هر کدام به صورت زیر، فیلدی جداگانه در نظر بگیریم.

نمونه طراحی ضعیف

جدول مشتری				
شماره مشتری	نام مشتری	تلفن	نشانی	مبلغ اعتبار
۱۲۰۱	کتابفروشی امید		استان تهران – شهر تهران – خیابان انقلاب – کوچه فرهادی – پلاک ۱۰ – کدپستی ۱۲۳۴۵۶۷۸۹۱	
۱۲۰۵	نشر طلوع		شیراز – خیابان ملاصدرا- کوچه ۲۱- پلاک ۱۵	

طراحی اصلاح شده

جدول مشتری							
شماره مشتری	نام مشتری	تلفن	استان	شهر	نشانی	کدپستی	مبلغ اعتبار
۱۲۰۱	کتابفروشی امید		تهران	تهران	خیابان انقلاب– کوچه فرهادی – پلاک ۱۰	۱۲۳۴۵۶۷۸۹۱	
۱۲۰۵	نشر طلوع		فارس	شیراز	خیابان ملاصدرا-کوچه ۲۱- پلاک ۱۵		

احتمالاً این پرسش برای شما پیش می‌آید که این کار چه مزیتی دارد؟

الف) در جدول اول امکان مرتب کردن نشانی‌ها بر حسب استان یا شهری که مشتری در آن زندگی می‌کند وجود ندارد.

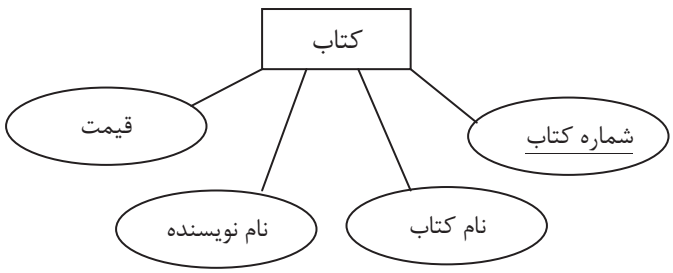
ب) اگر بخواهید پردازشی روی پایگاه داده انجام دهید تا مثلاً معلوم شود که در استان فارس چند مشتری وجود دارد، این کار فقط در جدول دوم امکان‌پذیر است.

در ادامه به شما نشان خواهم داد که حتی روی جدول اصلاح شده هم می‌توان تغییراتی را اعمال کرد تا عمل‌کرد پایگاه داده بهتر شود.

۳-۳-۳-۱ تعیین کلید اصلی

در بخش «آشنایی با مفاهیم پایگاه داده» توضیح داده شد که چرا هر جدول باید یک کلید اصلی داشته باشد. کلید اصلی به سیستم مدیریت پایگاه داده امکان می‌دهد تا به سراغ رکوردی یکتا برود. در دو جدول طراحی شده، جدول کتاب‌ها می‌تواند به خودی‌خود دارای یک فیلد یکتا باشد؛ چون وقتی کتاب چاپ می‌شود به آن یک شماره استاندارد بین‌المللی به نام شابک[1] اختصاص داده می‌شود که آن را از بقیه کتاب‌ها متمایز می‌کند.

در نمودارهای E-R با کشیدن یک خط زیر نام مشخصه، کلید اصلی را نشان می‌دهند.



در جدول طراحی شده، این کار را با رسم یک مستطیل نقطه‌چین دور نام فیلد انجام می‌دهند.

1 . International Standard Book Number (ISBN)

جدول مشتریان : Tbl_Customers				
CustomerID	CustomerName	Tel	Address	Credit
شماره مشتری	نام مشتری	تلفن	نشانی	مبلغ اعتبار

اگر در میان فیلدهای یک جدول، فیلد یکتایی وجود نداشـــته باشد که بتوان آن را به عنوان کلید اصلی معرفی کرد یا باید با ترکیب چند فیلد، یک کلید اصلی ترکیبی بسازید یا با اختصاص شماره‌ای منحصربفرد به هر رکورد، این مشـــکل را حل نمود. در طراحی پایگاه داده، روش دوم مرســـوم‌تر است بنابراین فیلدی به نام شماره مشتری به جدول اضافه می‌شود.

۴-۳-۳-۱ تبدیل رابطه‌ها به جدول

اتفاقی که در فرایند فروش یک مؤسسه انتشاراتی می‌افتد این است که یک یا چند عنوان کتاب و از هر عنوان تعداد خاصی در قالب یک فاکتور به مشـــتری فروخته می‌شـــود و در این فاکتور شماره فاکتور، تاریخ فاکتور، شماره مشتری و موارد خریداری شده درج می‌شود.

یک روش ابتدایی برای پیاده‌سازی فرایند فروش این است که جدول زیر را طراحی و اطلاعات فاکتورها را درون آن درج کنیم.

جدول فاکتورها					
شماره فاکتور	شماره مشتری	تاریخ فاکتور	شماره کتاب	تعداد	تخفیف
۱۰۰	۱۲۰۱	۱۳۸۹/۱۱/۰۱	۵۰۱۰	۳۵	٪۲۵
۱۰۰	۱۲۰۱	۱۳۸۹/۱۱/۰۱	۵۰۱۲	۱۰	٪۳۰
۱۰۰	۱۲۰۱	۱۳۸۹/۱۱/۰۱	۵۰۰۹	۳۶	٪۳۰
۱۰۱	۱۲۰۵	۱۳۸۹/۰۵/۱۱	۵۰۱۲	۲۰	٪۲۰
۱۰۱	۱۲۰۵	۱۳۸۹/۰۵/۱۱	۵۰۲۰	۸۰	٪۰

در این طراحی ضعیف، شماره فاکتور، شماره مشتری و تاریخ در هر رکورد تکرار می‌شود که باعث بروز

افزونگی[1] در سیستم خواهد شـــد. افزونگی، تکرار بی‌مورد اطلاعات در سطرهای جدول است که می‌تواند با یک طراحی خوب بر طرف شود.

جدول فاکتورها		
شماره فاکتور	شماره مشتری	تاریخ فاکتور
۱۰۰	۱۲۰۱	۱۳۸۹/۱۱/۰۱
۱۰۱	۱۲۰۵	۱۳۸۹/۰۵/۱۱

جدول جزییات فاکتورها			
شماره فاکتور	شماره کتاب	تعداد	تخفیف
۱۰۰	۵۰۱۰	۳۵	٪۲۵
۱۰۰	۵۰۱۲	۱۰	٪۳۰
۱۰۰	۵۰۰۹	۳۶	٪۳۰
۱۰۱	۵۰۱۲	۲۰	٪۲۰
۱۰۱	۵۰۲۰	۸۰	٪۰

به این ترتیب جدول قبلی را به دو جدول جدید تقسیم کردیم تا از بروز افزونگی در پایگاه داده جلوگیری کنیم. انجام این کار مزیت‌های زیر را دارد :

- در طراحی اولیه اگر بخواهیم رکورد جدیدی به جدول اضافه کنیم باید شـــمار مشتری و تاریخ را چندین بار وارد نماییم که وقت‌گیر است.
- در طراحی اولیه، اگر نیازی به تغییر در فاکتور فروش وجود داشـــته باشـــد، مثلاً بخواهیم تاریخ آن را تغییر دهیم باید این کار برای همه رکوردهای آن فاکتور انجام شـــود اما در طراحی دوم، با تغییر یک فیلد در «جدول فاکتورها» همه چیز درست می‌شود و اصلاً نیازی نیست به سراغ «جدول جزییات فاکتور» برویم.
- بـــا اصلاح طراحـــی، حجم جدول و به تبع آن پایگاه داده کاهش قابل ملاحظه‌ای پیدا می‌کند و بازیابی داده‌ها از آن با سرعت بیش‌تری انجام خواهد شد.

مبانی نظری تجزیه جداول که نرمال‌سازی نامیده می‌شود در انتهای این فصل بررسی خواهد شد.

۵-۳-۳-۱ کلید خارجی

به جدول‌های «فاکتورها» و «مشتریان» خوب دقت کنید.

1 . Redundancy

جدول فاکتورها		
شماره فاکتور	شماره مشتری	تاریخ فاکتور
۱۰۰	۱۲۰۱	۱۳۸۹/۱۱/۰۱
۱۰۱	۱۲۰۵	۱۳۸۹/۰۵/۱۱

جدول مشتریان					
شماره مشتری	نام مشتری	تلفن	استان	شهر	نشانی
۱۲۰۱	کتابفروشی امید		تهران	تهران	خیابان انقلاب – کوچه فرهادی – پلاک ۱۰
۱۲۰۵	نشر طلوع		فارس	شیراز	خیابان ملاصدرا-کوچه ۲۱- پلاک ۱۵

- در جدول «فاکتورها»، فیلد «شماره مشتری» وجود دارد.
- فیلد «شماره مشتری» در جدول «مشتری» کلید اصلی است.

در این حالت می‌گوییم «شـــماره مشـــتری» در «جدول فاکتورها» کلید خارجی است. برای نشان دادن کلید خارجی در یک جدول، از بیضی نقطه‌چین استفاده می‌کنیم.

۶-۳-۳-۱ طراحی نهایی

با طی مراحل زیر، طراحی شما به تدریج نهایی می‌شود :

- تبدیل موجودیت‌ها و رابطه‌ها به جدول
- ایجاد تغییرات در ساختار و تعداد جداول برای جلوگیری از افزونگی
- تعیین نام مناسب برای فیلدها
- مشخص کردن کلیدهای اصلی و خارجی

نتیجه یک طراحی که بر روی سیستم فروش مؤسسه انتشاراتی انجام گرفته به صورت شکل ۵-۱ است. ممکن است شما برحسب سطح نرمال‌سازی[1] انجام شده به طرح دیگری برسید که می‌تواند درست هم باشد. با مبحث نرمال‌سازی در بخش بعد آشنا خواهید شد.

1 . Normalization

جدول کتاب‌ها : Tbl_Books			
ID	Title	Author	Price
شماره کتاب	نام کتاب	نام نویسنده	قیمت

جدول مشتری : Tbl_Customers						
ID	Name	Tel	Province	City	Address	PostalCode
شماره مشتری	نام مشتری	تلفن	استان	شهر	نشانی	کد پستی

جدول فاکتورها : Tbl_Factors		
ID	CustomerID	Date
شماره فاکتور	شماره مشتری	تاریخ فاکتور

جدول جزییات فاکتور : Tbl_FactorDetails			
FactorID	BookID	Quantity	Discount
شماره فاکتور	شماره کتاب	تعداد	تخفیف

شکل ۵-۱

- در جدول جزییات فاکتور، ترکیب «شـــماره فاکتور و شماره کتاب» منحصربفرد است بنابراین از آن به عنوان کلید اصلی جدول استفاده کرده‌ایم. می‌توانستیم به جای این کار، یک شماره ترتیبی تعریف نموده و به هر رکورد یک شماره اختصاص دهیم.

- ممکن است ترجیح بدهید نام همه شهرهای ایران را درون یک جدول مجزا قرار داده و به هر شهر یک شـــماره اختصاص دهید. سپس در جدول مشـــتریان، فیلدهای «استان» و «شهر» را حذف کرده و این شماره را جایگزین کنید. این کار مزیت‌ها و معایبی دارد. خوبی کار در این است که حجم پایگاه داده کاهش

می‌یابد؛ مثلاً به جای اســـتان «چهارمحال و بختیاری» و شهر «شهرکرد» که در جداول به صورت «رشته» ذخیره می‌شود و چندین بایت اشغال می‌کند، یک عدد چهار رقمی ذخیره خواهد شد و وقتی صدها رکورد در جدول داشـــته باشـــید، تفاوت حجم قابل ملاحظه خواهد بود. مزیت دوم این است که اگر مثلاً نام شهر تغییر کرد، کافی اســـت این تغییر را در یک رکورد جدول شـــهرها اعمال کنید تا تغییر در کل پایگاه داده (پرس‌وجوها[1] و گزارش‌ها[2]) منعکس شود. عیب کار هم در این است که اگر بنا باشد همه موارد مشابه را در قالب یک شماره ذخیره کنید و برای توضیح شماره‌ها یک جدول جدید تعریف نمایید، ممکن است سیستم مدیریت پایگاه داده مجبور شـــود بـــرای تهیه یگ گزارش به ده‌ها جدول مراجعـــه کند و در نتیجه کارایی سیستم کم شود. در مبحث نرمال‌سازی با این موضوعات بیش‌تر آشنا می‌شوید.

- نام‌گذاری فیلدهای جدول به خصوص در زمانی که تعداد جدول‌های پایگاه داده زیاد می‌شود اهمیت فوق‌العاده‌ای دارد. سعی کنید نام‌های انتخاب شده دقیقاً متناسب با محتوای فیلدها باشد.

۱-۴ نرمال‌سازی جداول

وقتی به عنوان طراح پایگاه داده، یک ســـاختار واقعی را بـــه تعدادی جدول تبدیل می‌کنید همواره این سؤال برای شما مطرح می‌شود که آیا نتیجه کار، بهترین حالت ممکن است؟ در مدل رابطه‌ای روشی بسیار معروف برای پاســـخ دادن به این پرسش وجود دارد که آن‌را نرمال‌ســـازی می‌نامیم. با استفاده از این روش می‌توان جداول خوبی طراحی کرد یا طراحی صورت گرفته را بهبود بخشید. اما نرمال‌سازی دارای سطوحی است که پیش از توضیح آن ها باید با چند مفهوم تئوری آشنا شوید.

۱-۴-۱ رابطه

اگر فرض کنیم n مجموعه به صورت S_1، S_2، ... و S_n وجود داشـــته باشد که الزاماً متمایز نیستند، رابطه R روی این n مجموعه عبارت اســـت از : مجموعه‌ای از n تایی‌ها، به صورتی که برای هر n تایی، جزء اول از S_1، جزء دوم از S_2 و به همین ترتیب، جزء nام از S_n مقدار بگیرد. به بیان بهتر رابطه R عبارت است از ضرب دکارتی[3] $S_1 \times S_2 \times ... \times S_n$

مثال ۱: اگر S_1 مجموعه شـــماره‌های دانش‌آموزی، S_2 مجموعه اســـامی دانش‌آموزان و S_3 رشـــته

1 . Queries
2 . Reports
3 . Cartesian Product

تحصیلی باشـــد، STUDENT با تعدادی ســـه‌تایی به صـــورت زیر، یک رابطه اســـت و آن را به صورت زیر می‌نویسیم :

STUDENT (ID, NAME , COURSE)

یک نمونه از این رابطه می‌تواند مجموعه زیر باشد :

{ <ID : '۸۶۰۲۰' >, <NAME : 'مهدی ریاحی' > , <COURSE : 'تجربی' > }

و نمایش جدولی این رابطه به صورت زیر است :

STUDENT

ID	NAME	COURSE
۸۵۶۵۳	علی محمدی	ریاضی
۸۶۰۲۰	مهدی ریاحی	تجربی
۸۶۰۳۰	رضا عباسی	تجربی
۸۶۳۶۰	مجید منصوری	انسانی

به هر کدام از این n تایی‌ها یک چندگانه یا تاپل[1] می‌گوییم که در نمایش جدولی همان «رکورد» است. بین مفاهیم ریاضی رابطه و نمایش جدولی آن، تناظرهای زیر وجود دارد :

مفهوم رابطه‌ای	نمایش جدولی
رابطه	جدول
چندگانه	رکورد – سطر
مشخصه	ستون

۲-۴-۱ ویژگی‌های رابطه

رابطه یک مفهوم ریاضی است که در پایگاه داده‌های رابطه‌ای به صورت جدول پیاده‌سازی می‌شود. این مفهوم دارای چهار ویژگی زیر است:

الف) رابطه، تاپل تکراری ندارد چون بدنه رابطه، مجموعه است و مجموعه نمی‌تواند عنصر تکراری داشته باشد.

ب) تاپل‌ها، نظم ندارند و ترتیب آن‌ها مهم نیســـت چون مجموعه در حالت کلی فاقد نظم اســـت. برای

1 . Tuple

نمونه، این جمله که رکورد «علی محمدی» در ابتدای جدول قرار دارد بی‌معنی است.

ج) مشخصه‌های رابطه، نظم مکانی ندارند یعنی نمی‌توانیم بگوییم مثلاً نام دانش‌آموز مشخصه اول است و رشته او مشخصه دوم!

د)مشـــخصه‌ها، تک‌مقداری هستند یعنی در نمایش جدولی رابطه، در هر تاپل، برای هر مشخصه دقیقاً یک مقدار وجود دارد.

۳-۴-۱ وابستگی تابعی[1]

اگر A و B دو مجموعه مشـــخصه در شمای R باشـــند، آن‌گاه وابستگی تابعی A و B برقرار است هرگاه برای تمام رابطه‌ها در R، به ازای هر مقدار A فقط یک مقدار B وجود داشـــته باشـــد. وابستگی تابعی A و B را به صورت $A \rightarrow B$ نشان می‌دهیم.

مثال ۱: اگر A و B دو مجموعه به صورت $B=\{b_1,b_2,b_4\}$ و $A=\{a_1,a_2,a_3,a_5\}$ باشند، در رابطه

$$R1=\{(a_1,b_1),(a_2,b_2),(a_3,b_4),(a_5,b_1)\}$$

وابســـتگی $A \rightarrow B$ برقرار است چون به ازای مقادیر مساوی از A، مقادیر متفاوت از B وجود ندارد. ولی اگر دوتایی (a_1,b_4) را به این رابطه اضافه و رابطه $R2=\{(a_1,b_1),(a_1,b_4)(a_2,b_2),(a_3,b_4),(a_5,b_1)\}$ را بســـازیم، وابستگی تابعی از بین می‌رود چون برای a_1 دو مقدار b_1 و b_4 وجود خواهد داشت.

وابستگی تابعی، قواعد محیط عملیاتی را بیان می‌کند. مثلاً از عبارت زیر

معلم ← درس

مشخص می‌شود که هر درس فقط توسط یک معلم قابل ارایه است.

مثال ۲: همه مشخصه‌های یک موجودیت با مشخصه کلید آن وابستگی تابعی دارند.

مثلاً در مورد موجودیت دانش‌آموزان مدرســـه، نام دانش‌آموز با شـــماره دانش‌آموزی او وابستگی تابعی دارد، یعنی:

$$ID \rightarrow NAME$$

چون به ازای هر شماره دانش‌آموزی فقط یک نام وجود دارد.

---

1 . Functional Dependency (FD)

۱-۴-۴ وابستگی تابعی کامل[1]

مشخصه b با مشخصه a وابستگی تابعی کامل دارد یعنی :

$$a=>b$$

اگر b با a وابستگی تابعی داشته باشد (a⟶b) اما با هیچ زیرمجموعه‌ای از a وابستگی تابعی نداشته باشد.

مثال ۱ : اگر رابطه زیر را داشته باشیم :

A	B	C
a_1	b_1	c_1
a_2	b_1	c_2
a_1	b_2	c_2

وابستگی تابعی کامل بین C و (A , B) برقرار است چون :

اولاً : C⟶ (A , B) برقرار است چون برای هر دوتایی (a_i , b_j) فقط یک c_k وجود دارد.

ثانیاً : A⟶C به دلیل وجود$((a_1 , c_2) , (a_1 , c_1))$ برقرار نیست. همچنین B⟶C به خاطر دوتایی‌های $((b_1 , c_3) , (b_1 , c_1))$برقرار نمی‌باشد.

۱-۴-۵ نمودار وابستگی تابعی[2]

با استفاده از قواعد موجود در محیط عملیاتی، نمودار وابستگی تابعی رسم می‌شود تا به ما دید دقیق‌تری از طراحی صورت گرفته بدهد.

مثال ۱ : اگر قواعد زیر در یک محیط عملیاتی برقرار باشد، نمودار FD متناظر را رسم و یک جدول با مقادیر دلخواه ایجاد نمایید.

قاعده ۱ : هر ناشر، تعدادی کتاب چاپ می‌کند.
قاعده ۲ : ناشر از یک کتاب، شمارگان خاصی منتشر می‌کند.
قاعده ۳ : هر ناشر در یک شهر فروشگاه دارد.
قاعده ۴ : هر ناشر کتاب‌های خود را با تخفیف مشخصی می‌فروشد.
قاعده ۵ : فروشگاه‌های یک شهر کتاب‌ها را با تخفیف یکسانی می‌فروشند.

---

1 . Full Functional Dependency (FFD)
2 . FD Diagram

پاسخ : ابتدا برای نشـــان دادن وابستگی‌ها، به موجودیت‌ها و مشخصه‌ها حروف اختصاری زیر را نسبت می‌دهیم.

P# : شماره ناشر　　B#: شماره کتاب

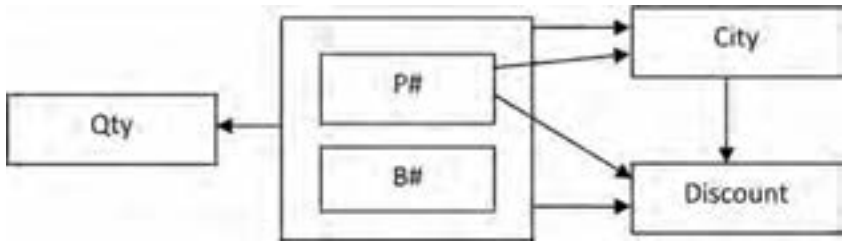
Qty : شمارگان　　City : شهر

Discount : درصد تخفیف

وجود یا عدم وجود وابستگی‌های تابعی در این محیط عملیاتی به صورت زیر است :

1. $P\# \nrightarrow B\#$
2. $(P\#, B\#) \rightarrow Qty$
3. $P\# \rightarrow City$
4. $P\# \rightarrow Discount$
5. $City \rightarrow Discount$

بنابراین نمودار وابستگی تابعی به صورت زیر قابل رسم است:



جدولی از مقادیر معتبر می‌تواند به صورت زیر باشد. این رابطه را FIRST نام‌گذاری می‌کنیم.

P#	B#	Qty	City	Discount
$P_۱$	$B_۱$	۱۱۰۰	$C_۱$	٪۲۰
$P_۲$	$B_۴$	۱۲۰۰	$C_۲$	٪۲۵
$P_۳$	$B_۳$	۵۰۰۰	$C_۱$	٪۲۰
$P_۱$	$B_۲$	۴۴۰۰	$C_۱$	٪۲۰
$P_۴$	$B_۳$	۳۳۰۰	$C_۳$	٪۳۰
$P_۴$	$B_۵$	۱۱۰۰	$C_۳$	٪۳۰

۶-۴-۱ آنومالی[1]

آنومالـــی یا بی‌نظمی یعنی بروز وضعیـــت نامطلوب در جریان انجام یک عملیات درون پایگاه داده که به دلیل طراحی نامناسب رخ می‌دهد. آنومالی در سه محور زیر قابل بررسی است :

الف) انجام‌ناپذیری یک عملیات در پایگاه داده

ب) بروز پیامد نامطلوب در پی انجام یک عملیات

ج) فزون‌کاری برای انجام یک عملیات

نرمال‌سازی، روشی مؤثر برای کاهش آنومالی در یک پایگاه داده است.

۷-۴-۱ شرح آنومالی‌ها

برای بررسی سه وجه آنومالی، رابطه تعریف شده در مثال اخیر را در نظر بگیرید.

الـــف) انجام‌ناپذیری (آنومالی در درج) : فرض کنید می‌خواهیم رکورد< C_2 , 25% , 1400 , B_2 , < را در رابطه FIRST درج کنیم. اما این درج امکان‌پذیر نیســـت چون معلوم نیست این کتاب توسط چه ناشری منتشـــر شده است. در آنومالی ناشـــی از درج، کلید اصلی یا بخشی از آن تعریف نشده و همان‌طور که قبلاً توضیح داده شد، مقدار کلید اصلی باید مشخص باشد.

ب) بـــروز پیامد نامطلوب (آنومالی در حذف) : قصد داریم رکـــورد < 1200 , B_4 , P_2 > را حذف کنیم. اگرچه این کار امکان‌پذیر است اما باعث می‌شود اطلاع < 25% , C_2 >هم از بین برود و دیگر ندانیم میزان تخفیف فروشگاه‌ها در شهر C_2 چه‌قدر است. بنابراین حذف یک رکورد منجر به بروز تبعات نامطلوب خواهد شد.

ج) فزون‌کاری (آنومالی در بروزرسانی) : می‌خواهیم میزان تخفیف فروشگاه‌ها در شهر C1 را از 20% به 15% تغییر دهیم. به این ترتیب باید در ســـه سطر از جدول، مقدار Discount را بروزرسانی کنیم که نوعی فزون‌کاری و نشان‌دهنده طراحی ضعیف است.

مطالب ذکر شده نشـــان می‌دهد رابطه FIRST ساختار خوبی ندارد و نوعی «اختلاط اطلاعاتی» در آن روی داده؛ یعنی اطلاعات مربوط به موجودیت‌ها، بی‌جهت با هم مخلوط شـــده است. به عنوان نمونه لزومی

---

1 . Anomaly

ندارد میزان تخفیف فروشگاه‌های یک شهر که اطلاعی پایه‌ای محسوب می‌شود با اطلاعات کتاب‌های چاپ شده و شمارگان آن‌ها ترکیب شود.

برای ایجاد یک طراحی مناسب باید رابطه را از نظر سطح نرمال بودن بررسی کنیم و در صورت نرمال نبودن، با پیروی از مجموعه‌ای از دستورالعمل‌ها جداول را تجزیه نموده و به سطح نرمال موردنظر برسانیم.

۸-۴-۱ سطوح نرمال[1]

سطوح مختلف نرمال را می‌توان به صورت زیر بیان نمود.

۱. سطح اول نرمال یا 1NF
۲. سطح دوم نرمال یا 2NF
۳. سطح سوم نرمال یا 3NF
۴. سطح نرمال بایس-کاد یا BCNF[2]
۵. سطح چهارم نرمال یا 4NF
۶. سطح پنجم نرمال یا 5NF

با توجه به پیچیدگی سطوح بالای نرمال‌سازی و استفاده از آن‌ها در موارد خاص، این کتاب فقط تا سطح نرمال سوم را بررسی خواهد کرد.

۱-۸-۴-۱ سطح نرمال اول

تعریف : رابطه R در سطح اول نرمال (1NF) است اگر همه مشخصه‌ها (فیلدها) تک‌مقداری باشند.

مثـــال ۱ : رابطه کتاب‌های موجود در یک کتاب‌خانه را در نظر بگیرید، آیا این رابطه در 1NF قرار دارد؟

شماره کتاب	نام کتاب	اطلاعات نشر
۱۲۰۳	برنامه‌نویسی C	نقش‌سیمرغ - ۱۳۸۸
۱۲۰۴	SQL Server	عابد – ۱۳۸۹
۱۲۰۶	Access	برگ زیتون - ۱۳۸۹

فیلد اطلاعات نشر قابل تجزیه به دو فیلد «ناشر» و «سال انتشار» است و اگر بخواهیم سال انتشار را از

1 . Normal Form (NF)
2 . Boyce-Codd Normal Form

رابطه به دســـت بیاوریم باید این فیلد را تجزیه کنیم. بنابراین رابطه غیرنرمال است و در 1NF قرار ندارد. با تبدیل رابطه به حالت زیر، در 1NF قرار می‌گیرد.

شماره کتاب	نام کتاب	ناشر	سال انتشار
۱۲۰۳	برنامه‌نویسی C	نقش‌سیمرغ	۱۳۸۸
۱۲۰۴	SQL Server	عابد	۱۳۸۹
۱۲۰۶	Access	طلوع	۱۳۸۹

۲-۸-۴-۱ سطح نرمال دوم

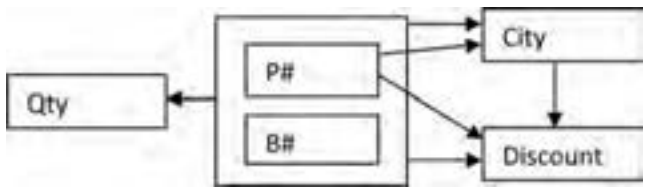
تعریف : رابطه R در سطح دوم نرمال است اگر و فقط اگر:

الف) 1NF باشد.

ب) تمام مشخصه‌های غیرکلید، با کلید اصلی وابستگی تابعی کامل داشته باشند.

مثال ۱ : رابطه FIRST را درنظر بگیرید، آیا این رابطه در 2NF است؟

کلیـــد اصلی در این رابطه، صفت مرکب (P#,B#) اســـت. اگر به نمـــودار دقت کنید می‌بینید که تمامی فیلدها با این کلید وابستگی تابعی دارند.



به عنوان نمونه : $(P\#,B\#) \longrightarrow City$

طبق تعریف در صورتی وابستگی تابعی کامل برقرار است که :

اولاً (P#,B#) با City وابستگی تابعی داشته باشد.

ثانیاً City با هیچ زیرمجموعه‌ای از (P#,B#) وابستگی تابعی نداشته باشد اما طبق نمودار :

$P\# \longrightarrow City$ و $B\# \not\longrightarrow City$

لذا وابســـتگی تابعی کامل (FDD) بین City و (P#,B#) وجود ندارد. در مورد مشخصه‌های Discount و Qty هم وضع به همین منوال است بنابراین رابطه FIRST در 2NF قرار ندارد.

در بخش ۷-۴-۱ آنومالی‌های موجود در رابطه FIRST را شرح دادیم، حال می‌توانیم چنین نتیجه‌گیری کنیم که دلیل این آنومالی‌ها نقض شـــدن وابســـتگی تابعی کامل است. برای رفع آنومالی و بالا بردن سطح نرمال باید رابطه FIRST را تجزیه کنیم.

تجزیه رابطه R به روابط R_1 و R_2 باید به گونه‌ای انجام گیرد که:

اولاً با پیوند دو رابطه R_1 و R_2 رابطه R تولید شود و در این میان هیچ چندتایی[1] از دست نرود.

ثانیاً تمامی وابستگی‌های تابعی رابطه R حفظ شود.

مثال ۲ : رابطه FIRST را به‌گونه‌ای تجزیه کنید که رابطه‌های جدید در 2NF قرار گیرند.

اطلاعات پایه‌ای نشر را از جدول کتب چاپ شده جدا می‌کنیم تا دو رابطه به صورت زیر به دست بیاید:

PB (P# , B# , Qty)

SECOND (P# , City , Discount)

PB

P#	B#	Qty
P_1	B_1	1100
P_2	B_4	1200
P_3	B_3	5000
P_1	B_2	4400
P_4	B_3	3300
P_4	B_5	1100

SECOND

P#	City	Discount
P_1	C_1	20%
P_2	C_2	25%
P_3	C_1	20%
P_4	C_3	30%

رابطه SECOND در سطح دوم نرمال است چون:

اولاً 1NF است.

ثانیاً وابستگی تابعی کامل برقرار است.

مثال ۳ : آنومالی‌های موجود در رابطه SECOND را شرح دهید.

الف) در درج : درج رکورد جدیدی مثل <C_4 , 15%> , > که به صورت مســـتقل بامعناســـت و نشـــان می‌دهد در فروشـــگاه‌های شـــهر C_4 کتاب‌ها با ۱۵ درصد تخفیف عرضه می‌شود «انجام ناپذیر» است چون

---

۱ . رکورد یا سطری از جدول

ناشر نامشخص است.

ب) در حذف : حذف اطلاع $< P_4 , C_3 , 30\% >$ منجر به از دست رفتن رکوردی می‌شود که نشان می‌دهد ناشر P_4 در شهر C_3 فروشگاه دارد.

ج) در بروزرسانی : برای تغییر درصد تخفیف در شهر C_1 باید بیش از یک رکورد تغییر یابد.

۳-۸-۴-۱ سطح نرمال سوم

تعریف : رابطه R در سطح سوم نرمال است اگر و فقط اگر :

الف) 2NF باشد.

ب) هر مشخصه غیرکلید با کلید اصلی فقط وابستگی تابعی بی‌واسطه داشته باشد.

مثال ۱ : ثابت کنید اگر $B \rightarrow A$ و $C \rightarrow B$ برقرار باشد آن‌گاه $A \rightarrow C$ هم برقرار است.

با استفاده از برهان خلف، فرض می‌کنیم $A \not\rightarrow C$ و لذا حداقل در دو تاپل، به ازای یک مقدار از A، دو مقدار متمایز از C داریم :

R (A , B , C)
a1 ….. c1
a1 ….. c2

اما به ازاء دو مقدار متمایز از C ، مقدار B می‌تواند متمایز یا یکسان باشد بنابراین دو حالت زیر قابل تصور است:

الف

a1	b1	c1
a1	b1	c2

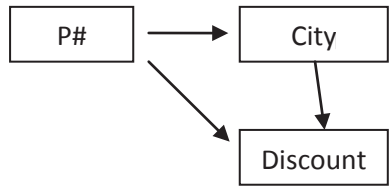
ب

a1	b1	c1
a1	b2	c2

در حالت «الف»، نادرستی $C \rightarrow B$ و در قسمت «ب» نادرستی $B \rightarrow A$ ثابت می‌شود و لذا فرض نادرست است و $A \rightarrow C$ ثابت می‌شود. این قاعده در جبر رابطه‌ای، تعدی نامیده می‌شود.

مثال ۲ : آیا رابطه SECOND در 3NF قرار دارد؟

این رابطه 2NF است بنابراین شرط اول در تعریف رابطه 3NF برقرار است اما اگر نمودار FD این رابطه را در نظر بگیرید:



می‌بینید که City ← P# و Discount ← City و بنابراین طبق قاعده تعدی، Discount ← P# ثابت می‌شـــود. به بیان دیگر Discount با P# به واســـطه City وابستگی تابعی دارد و لذا شرط دوم تعریف 3NF نقض شده است.

برای رفع آنومالی در SECOND و بردن آن به سطح 3NF، این رابطه باید به دو رابطه زیر تجزیه شود:

PC (P# , City)
CD (City , Discount)

PC

P#	City
P_1	C_1
P_2	C_2
P_3	C_1
P_4	C_3

CD

City	Discount
C_1	20%
C_2	25%
C_3	30%

به این ترتیب آنومالی‌های شرح داده شده در بخش‌های قبلی از بین می‌رود. نرمال‌سازی غالباً تا سطحی انجام می‌گیرد که ســـرعت اجرای پرس‌وجوها و برگرداندن نتایج را بیش از حد مجاز کاهش ندهد چون در پایگاه داده‌هایی با ســـطح نرمال‌ســـازی بالا باید چندین جدول با هم پیوند بخورند تا نتیجه موردنظر حاصل شـــود. به‌خصوص برای پایگاه داده‌هایی که تحت وب اجرا می‌شوند و سرعت بازیابی اطلاعات اهمیت بالایی دارد رعایت این مسأله مهم است.

The design process

The design process consists of the following steps:

- Determine the purpose of your database

This helps prepare you for the remaining steps.

- Find and organize the information required

Gather all of the types of information you might want to record in the database, such as product name and order number.

- Divide the information into tables

Divide your information items into major entities or subjects, such as Products or Orders. Each subject then becomes a table.

- Turn information items into columns

Decide what information you want to store in each table. Each item becomes a field, and is displayed as a column in the table. For example, an Employees table might include fields such as Last Name and Hire Date.

- Specify primary keys

Choose each table's primary key. The primary key is a column that is used to uniquely identify each row. An example might be Product ID or Order ID.

- Set up the table relationships

Look at each table and decide how the data in one table is related to the data in other tables. Add fields to tables or create new tables to clarify the relationships, as necessary.

Refine your design

Analyze your design for errors. Create the tables and add a few records of sample data. See if you can get the results you want from your tables. Make adjustments to the design, as needed.

Apply the normalization rules

Apply the data normalization rules to see if your tables are structured correctly. Make adjustments to the tables, as needed.

۱. متن بالا را مطالعه کرده و در مورد آن توضیح دهید.

۲. مراحل طراحی یک پایگاه داده را نام ببرید.

۳. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. مزایا و معایب سیستم مدیریت پایگاه داده را توضیح دهید.

۲. وجود کلید اصلی در یک جدول چه ضرورتی دارد؟

۳. در محیط عملیاتی دانشگاه، روابط زیر برقرار است :

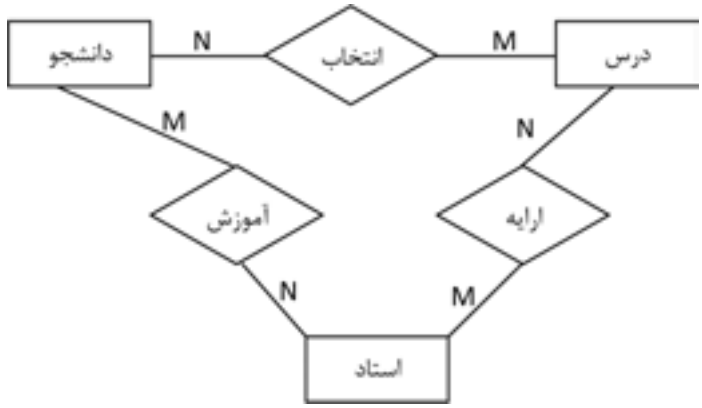
الف) دانشجو، درس را انتخاب می‌کند.
ب) درس توسط دانشجو حذف می‌شود.
ج) استاد، درس را ارایه می‌کند.

با تخصیص تعدادی مشخصه به موجودیت‌ها و روابط، نمودار E-R این محیط عملیاتی را رسم نمایید.

۴. ویژگی‌های یک رابطه را توضیح دهید.

۵. وابستگی تابعی و وابستگی تابعی کامل را با ذکر مثال توضیح دهید.

۶. نمودار E-R زیر را در نظر بگیرید:



فرض کنید اطلاع‌های زیر را از نمودار فوق استنتاج کرده‌ایم:

I_1 : دانشجوی S_1 درس C_1 را انتخاب کرده است.

I_2 : درس C_1 توسط استاد P_1 ارایه می‌شود.

I_3 : استاد P_1 به دانشجوی S_1 درس می دهد.

آیا اطلاع زیر قابل نتیجه‌گیری است؟

I_4 : استاد P_1 درس C_1 را به دانشجوی S_1 تدریس می‌کند.

۷. رابطه $R(A,B,C)$ را با بســـط زیر در نظر بگیرید. این رابطه در چه ســـطحی از نرمال‌سازی قرار دارد؟ این رابطه را به گونه ای تجزیه کنید که 3NF باشد.

A	B	C
a_1	b_1	c_1
a_1	b_2	c_1
a_1	b_1	c_2
a_1	b_2	c_2
a_2	b_3	c_3
a_3	b_4	c_1
a_3	b_4	c_4

فصل دوم

فصل دوم : تغییر جداول پایگاه داده

حال که با روش طراحی یک پایگاه داده آشنا شدید به سراغ مطالب پیشرفته‌تری در کار با سیستم پایگاه داده Access می‌رویم. آشـــنایی با مهارت‌های انتقال اطلاعات بین دو سیســـتم پایگاه داده متفاوت و تبدیل یک جدول به فرمت‌های قابل استفاده در سایر نرم‌افزارها، موضوعی است که در این فصل مورد بررسی قرار می‌گیرد.

۱-۲ اصول واردکردن جدول

یکـــی از مهارت‌های موردنیاز برای طراحـــان یا مدیران پایگاه داده، توانایـــی انتقال دادن اطلاعات بین سیستم‌های مختلف مدیریت پایگاه داده است. در برخی موارد هم لازم می‌شود شما یک فایل Excel یا فایل متنی ساده را که درون آن رکوردهای اطلاعاتی وجود دارد به درون جداول یک پایگاه داده انتقال دهید.

تعریف : عملیاتی را که طی آن، داده‌ها از یک منبع خارجی وارد پایگاه می‌شـــود Importing یا Import کردن می‌گوییم.

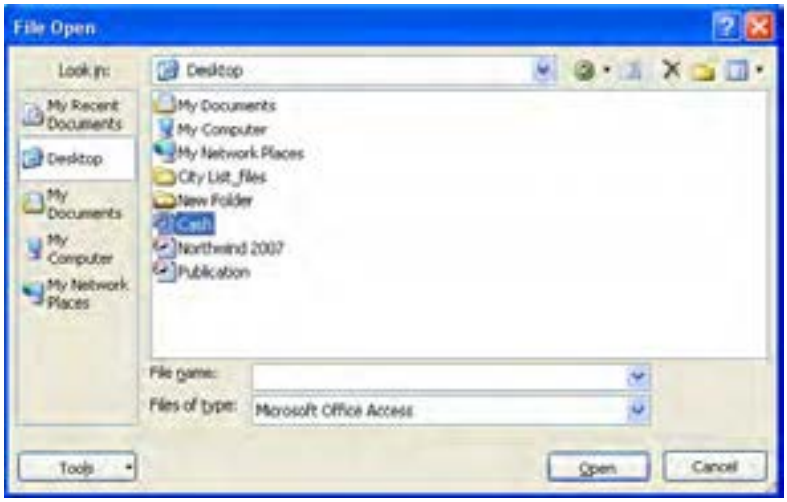
وارد کردن یک جدول از یک بانک اطلاعاتی دیگر

گاهی اوقات لازم می‌شـــود شـــما یکی از جداول موجود در یک پایگاه داده Access یا SQL Server را وارد پایگاه داده‌ای کنید که در حال ســـاخت و تکمیل آن هستید. برای مثال فرض کنید در یک پایگاه داده Access جدول تمامی شـــهرهای ایران وجود دارد و شما در پایگاه داده جدید به این جدول نیاز دارید. یک راه‌حل ابتدایی این است که جدول خامی ایجاد نموده و لیست شهرها و شماره آن‌ها را تایپ کنید اما قابلیت Importing سیستم‌های مدیریت پایگاه داده به شما امکان می‌دهد این کار را با سهولت و سرعت بیش‌تری انجام دهید.

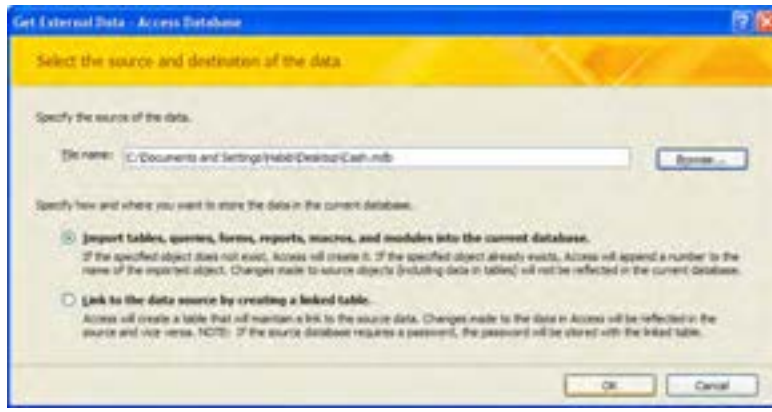
۱. پایگاه داده‌ای را که قصد دارید جدول جدیدی به آن اضافه کنید در محیط Access باز کنید.

۲. به زبانه External Date رفته و روی دکمه Access که در شکل زیر نشان داده شده کلیک کنید.

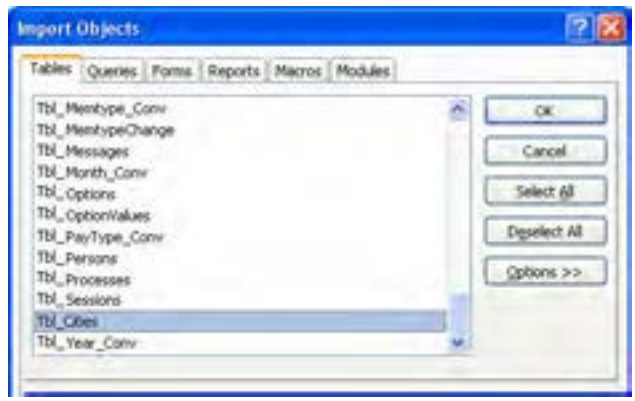
در پنجره‌ای که ظاهر می‌شـــود روی دکمه Browse کلیک نمـــوده و در پنجره File Open، پایگاه داده حاوی جدول موردنظر را انتخاب کنید. روی دکمه Open کلیک کنید.



۳. در پنجره تعیین پایگاه داده مبدأ، دکمه OK را بزنید.

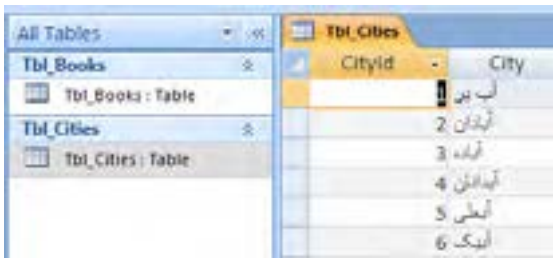


۴. لیستی از جداول، پرس‌وجوها، گزارش‌ها و سایر عناصر موجود در پایگاه داده انتخاب شده ظاهر می‌گردد. در زبانه Tables روی جدول موردنظر کلیک کرده و دکمه OK را بزنید.



۵. با کلیک کردن روی دکمه Close که در آخرین پنجره ویزارد قرار دارد، عملیات انتقال صورت می‌گیرد.

۶. جدول انتخاب شده به لیست جداول پایگاه داده اضافه خواهد شد.



وارد کردن یک جدول از یک کارپوشه Excel

در بســـیاری از محیط‌های عملیاتی، افراد ترجیح می‌دهند اطلاعات خود را در فایل‌های Excel نگه‌داری کنند، چون ذخیره‌سازی داده‌ها در این محیط بسیار ساده و سریع است. اما کم‌کم با افزایش حجم داده‌ها و نیاز به انجام پردازش روی آن‌ها، دیگر نرم‌افزار Excel کارایی خود را از دست می‌دهند و انتقال داده‌ها درون یک پایگاه داده نظیر Access اجتناب‌ناپذیر می‌شود. در این بخش با روش انجام این کار آشنا می‌شوید.

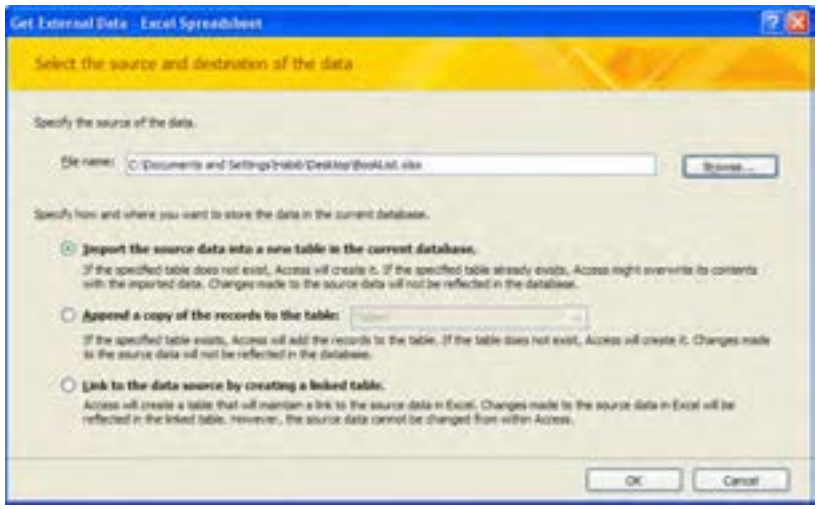
۱. یک کارپوشـــه Excel حاوی جدول مشخصات کتاب‌های یک مؤسسه انتشاراتی ایجاد کنید. این فایل می‌تواند حاوی ستون‌های شماره کتاب، نام کتاب، نام و نام خانوادگی نویسنده، قیمت کتاب و ... باشد. چند رکورد در آن ایجاد نموده و با نام BookList ذخیره کنید.

BookCode	BookTitle	AuthorName	AuthorLastname	Price
1202	برنامه نویسی C	محمد	کبیری	55,000
1203	دلفی	علی	رضایی	62,000
1206	مدیریت پایگاه داده	عاطفه	مجیدی	75,000
1207	Access 2007	رضا	مؤمنی	61,000

۲. حال برنامه Access 2007 را باز نموده و یک فایل پایگاه داده خالی با نام Books ایجاد نمایید.

۳. سپس به زبانه External Data رفته و روی دکمه سبزرنگ Excel کلیک کنید.

۴. پنجره انتخاب فایل Excel ظاهر می‌شـــود. با کلیـــک کـــردن روی دکمـــه Browse، فایل BookList را به پنجره معرفی کنید.

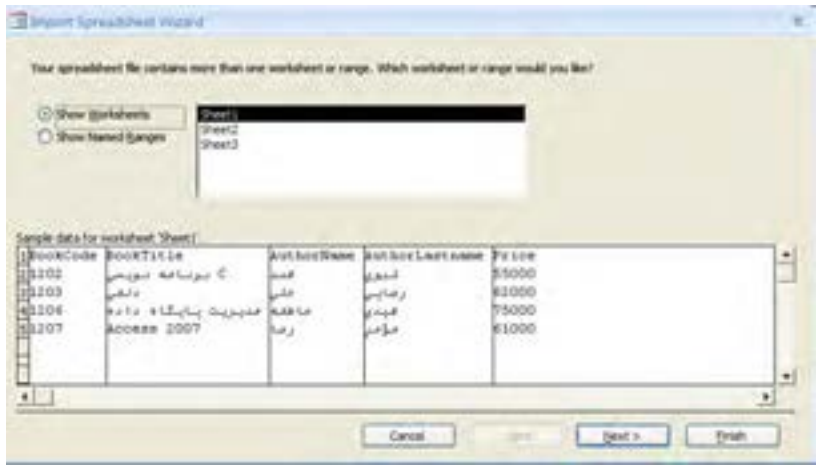


در این پنجره سه گزینه وجود دارد که دوتای اول به صورت زیر عمل می‌کنند :

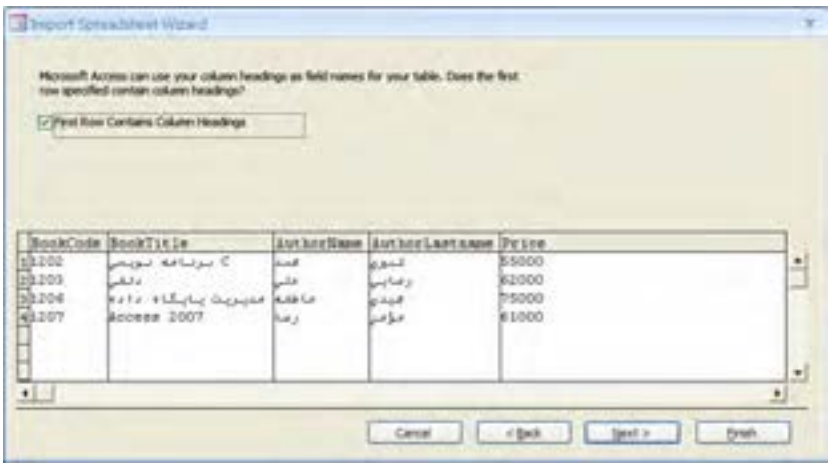
گزینه اول، کارپوشه Excel را وارد یک جدول خام Access نموده و فیلدهای این جدول را با سرستون‌های فایل Excel نام‌گذاری می‌کند. **گزینه دوم** فرض می‌کند شما قبلاً در Access جدولی حاوی فیلدهای متناظر با فایل Excel ایجاد نموده‌اید و قصد دارید رکوردهای موجود در کارپوشه را به جدول Access اضافه کنید.

۵. گزینه اول را انتخاب و روی دکمه OK کلیک نمایید.

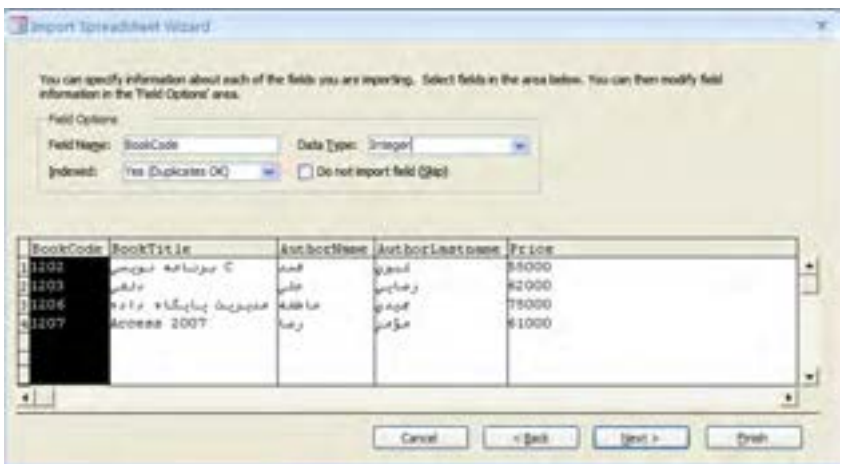
۶. در پنجره ظاهر شده، گزینه Show worksheets را انتخاب نموده و در لیست روبه‌روی آن روی کارپوشه موردنظر کلیک کنید. داده‌های موجود در این کارپوشه نمایش داده می‌شود. روی دکمه Next کلیک کنید.



۷. بـــا علامت زدن عبـــارت بالای پنجره جدیـــد به برنامه اعـــلام می‌کنید که ردیـــف اول حاوی عنوان سرستون‌هاست. روی دکمه Next کلیک نمایید.



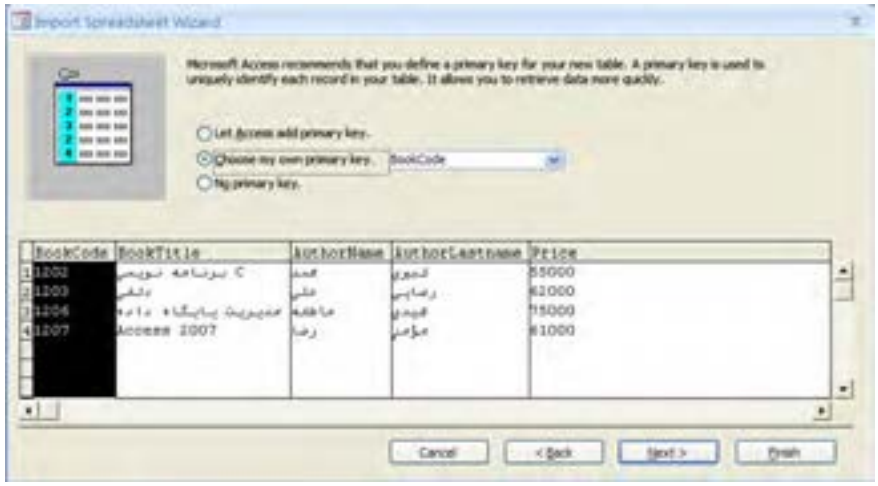
۸. با کلیک کردن روی هر ســـتون، پس‌زمینه آن به رنگ سیاه در می‌آید و امکان تغییر تنظیمات ستون در بخش Field Options فراهم می‌آید. برای مثال می‌توانید نام ســـتون[1] و نوع داده‌ای[2] آن را تغییر دهید. در این مثال BookCode را از Double به Integer تغییر دهید چون شـــماره کتاب یک عدد صحیح اســـت. تغییرات موردنظر را روی سایر ستون‌ها اعمال و روی دکمه Next کلیک نمایید.



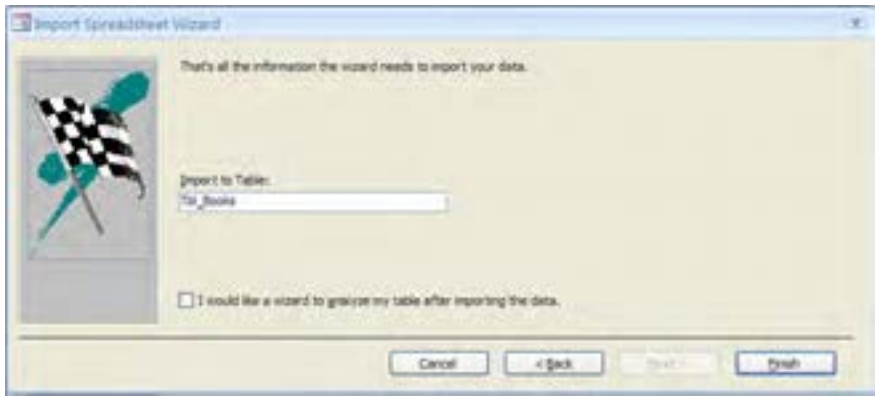
1 . Field Name

2 . Data Type

۹. پنجـــره بعدی مخصوص تعیین کلید اصلی جدول اســـت. گزینه اول، بـــه Access اجازه می‌دهد یک شناســـه[1] یکتا به جدول اضافه نماید. این کار در مورد جداولی که فیلد یکتا ندارند ضروری اســـت اما در این مثال، شماره کتاب برای هر رکورد منحصربه‌فرد است؛ بنابراین گزینه دوم را انتخاب و از لیست روبه‌رو، فیلد BookCode را انتخاب نمایید. نهایتاً روی دکمه Next کلیک کنید.



۱۰. در ایـــن پنجره، یک نام مناســـب برای جـــدول وارد نموده و روی دکمـــه Finish کلیک کنید تا عمل Importing انجام شود.



1 . ID

۱۱. جدولی با نام تعیین شده و حاوی رکوردهای کارپوشه Excel ایجاد می‌شود.

وارد کردن یک جدول از یک فایل متنی

فایل‌های متنی با فرمت txt. می‌توانند به عنوان منبعی برای ذخیره‌سازی داده‌ها مورد استفاده قرار بگیرند. در این نوع فایل‌ها، هر رکورد اطلاعاتی درون یک سطر ذخیره می‌شود و فیلدهای آن رکورد با یک علامت جداکننده[1] مانند کاما، نقطه-کاما، علامت نقل قول، Tab، Space و ... از هم تفکیک می‌شوند.

فرض کنید یک فایل متنی حاوی نام و نام خانوادگی نویسندگان یک مؤسسه انتشاراتی و نشانی پست الکترونیک آن‌ها داریم و می‌خواهیم این فایل را درون پایگاه داده Access وارد کنیم. در این فایل، فیلدها با Tab از یک‌دیگر جدا شده‌اند.

۱. فایل پایگاه داده را باز کرده و در زبانه External Data روی دکمه Text File کلیک کنید.

1 . Delimiter

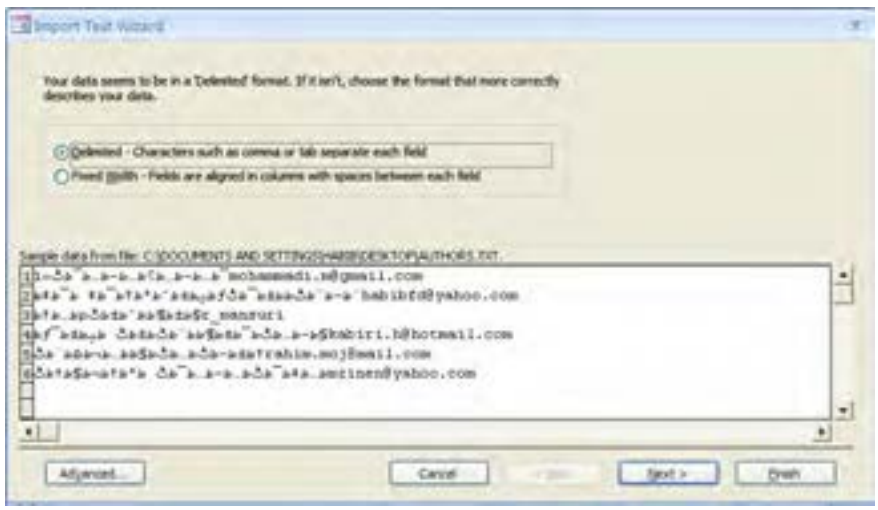
۲. با کلیک کردن روی دکمه Browse، فایل متنی را باز نموده و روی دکمه OK کلیک کنید.

۳. در پنجره بعدی رکوردهای موجود در فایل نشـــان داده می‌شود. ضمناً در بالای پنجره، دو گزینه وجود دارد که به صورت زیر عمل می‌کنند:

Delimited : با انتخاب این گزینه به برنامه اعلام می‌کنید که فیلدهای هر رکورد با نوعی جداکننده از هم تفکیک شده‌اند.

Fixed Width : گاهی اوقات برای ســـاخت یک فایـــل متنیِ حاوی رکوردهای اطلاعاتی، برای هر فیلد عرض مشخصی در نظر گرفته می‌شود تا به این ترتیب، فیلدهای یک رکورد از هم جدا شوند.

ما در ساخت فایل متنی از روش اول استفاده کرده‌ایم بنابراین گزینه Delimited را انتخاب می‌کنیم



همان‌طور که در تصویر بالا می‌بینید، عباراتی که در فایل متنی به زبان فارســـی نوشـــته شده بودند، در

این پنجره ناخوانا هســـتند. به دلیل این که هنگام ذخیره‌ســـازی فایل متنی، باید روش کدگذاری[1] مناسبی انتخاب شود. اگر در حین Importing روش کدگذاری مناسبی برای فایل اعلام نشود ممکن است Access در تشخیص نویسه‌ها[2]ی غیرانگلیسی دچار مشکل شود. برای حل این مشکل :

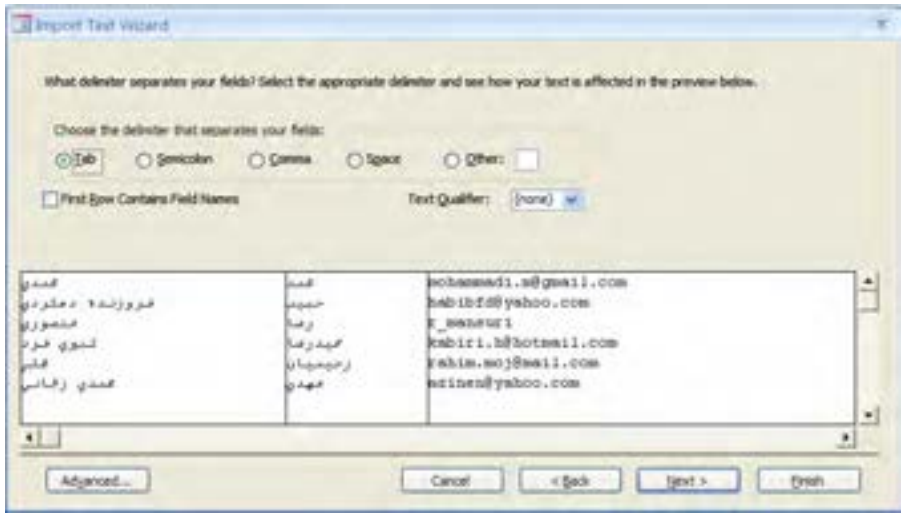
۴. روی دکمه Advanced کلیک کنید تا پنجره Import Specification ظاهر شـــود. در بالای این پنجره می‌توانید تعیین کنید که فیلدها با چه نوع جداکننده‌ای از هم تفکیک شـــده‌اند. همچنین با انتخاب گزینه مناســـب در لیست Code Page مشکل ناخوانا بودن نوشـــته‌های فارسی حل می‌شود. فایل متنی مثال ما با کدگذاری UTF-8 ذخیره شده که نسبت به بقیه روش‌های کدگذاری کارآمدتر است.

۵. پـــس از انجام تنظیمات، روی دکمه OK کلیک کنید تا بـــه پنجره اصلی برگردید. در این پنجره روی دکمه Next کلیک نمایید.

۶. در پنجره بعد هم امکان تعیین نویسه جداکننده وجود دارد. ضمناً اگر نام سرستون‌ها در ردیف اولِ فایل

1 .Encoding
2 . Characters

متنی وجود داشـــته باشد، با تیک زدن عبارت First Row Contains Fields Names می‌توانید نام فیلدها را هم از فایل متنی به جدول Access منتقل کنید. روی دکمه Next کلیک کنید تا پنجره بعد ظاهر شود.



۷. نوبت به تعیین نام فیلدها و نوع داده‌ای آن‌ها می‌رسد. روی ستون موردنظر کلیک و در بالای پنجره نام دلخواه را وارد و نوع داده‌ای را مشخص کنید. نهایتاً روی دکمه Next کلیک نمایید.

۸. از آن‌جا که رکوردهای وارد شـــده فیلد منحصربه‌فردی ندارند تا تبدیل به کلید اصلی جدول شـــود، با انتخاب گزینه اول به Access اجازه دهید یک فیلد شناسه یکتا به جدول اضافه کند.

۹. در پنجره بعد هم نام جدول را تعیین نموده و روی دکمه Finish کلیک کنید.

۱۰. عملیات Import داده‌ها انجام و جدول جدیدی ایجاد می‌شود.

۲-۲ اصول صدور جدول

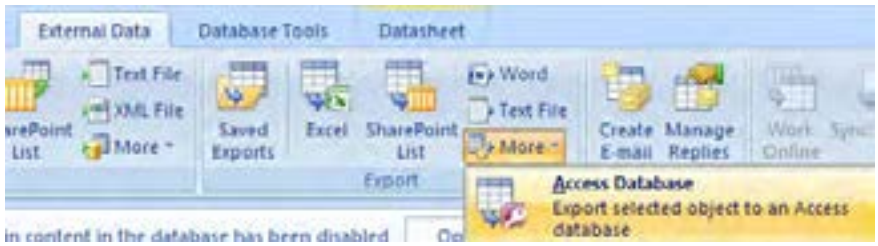
تبادل اطلاعات میان پایگاه‌های داده مختلف یکی از مهارت‌های اساسی در کار با DBMS ها محسوب می‌شود. چنان‌چه نرم‌افزار پایگاه داده مبدأ و مقصد با هم سازگار و ترجیحاً محصول یک شرکت باشند معمولاً این انتقال به آسانی انجام می‌شود؛ به عنوان نمونه، بین سیستم‌های مدیریت پایگاه داده Access و SQL Server قابلیت‌های قدرتمندی برای انتقال داده‌ها وجود دارد. اما همیشه کار به این سادگی نیست. گاهی اوقات لازم می‌شود داده‌های درون یک جدول به قالب‌های ساده‌ای مثل فایل متنی یا HTML تبدیل شوند تا سیستم پایگاه داده مقصد قادر به شناسایی آن‌ها باشد. به همین دلیل مبحث Exporting داده‌ها جداول اهمیت زیادی دارد.

الف) صدور یک جدول به یک بانک اطلاعاتی دیگر

می‌خواهیم یکی از جداول پایگاه داده Access را به فایل Access دیگر Export کنیم.

۱. با دوبار کلیک روی نام جدول موردنظر، آن را باز کنید.

۲. در زبانه External Data روی دکمه More کلیک و از منوی ظاهر شـــده، گزینه Access Database را انتخاب نمایید.



۳. پایگاه داده مقصد را انتخاب کنید.

۴. در پنجره‌ای که ظاهر می‌شـــود مشـــخص کنید که آیا می‌خواهید فقط ساختار جدول منتقل شود یا داده‌هـــای درون آن هم صادر شـــوند. گزینه اول یک کپی از جدول و محتویـــات آن در پایگاه داده انتخابی ایجاد می‌کند.

۵. در صـــورت تمایل، نام جدید برای جدول انتخاب و روی دکمـــه OK کلیک کنید تا عملیات Export انجام شود.

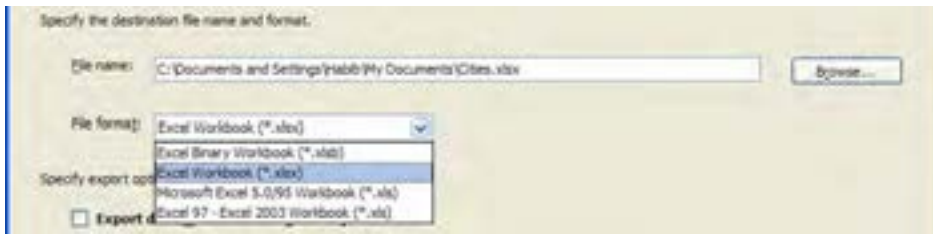
ب) صدور یک جدول به یک کارپوشه Excel

۱. پایگاه داده موردنظر را در محیط Access باز کنید.

۲. در لیست جداول، روی جدول موردنظر دوبار کلیک کنید تا باز شود.

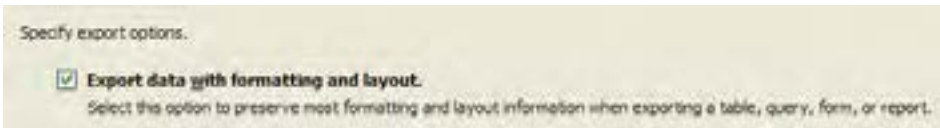
۳. در قاب Export روی دکمه Excel کلیک کنید.

۴. محل ذخیره‌سازی فایل کارپوشه و نام آن را تعیین نمایید.



۵. اگر می‌خواهید فایل تولیدی با نسخه‌های قدیمی‌تر Excel مثل ۲۰۰۳ و ۹۷ سازگار باشد، از لیست File format گزینه آخر را انتخاب کنید.

۶. ترجیحاً گزینه Export data with formatting and layout را تیک بزنید تا برای صدور نویسه‌های فارسی و حروفی مانند «ی»، «گ» و ... مشکلی ایجاد نشود.



۷. با کلیک کردن روی دکمه OK، کارپوشه ساخته می‌شود.

ج) صدور یک جدول به یک فایل متنی

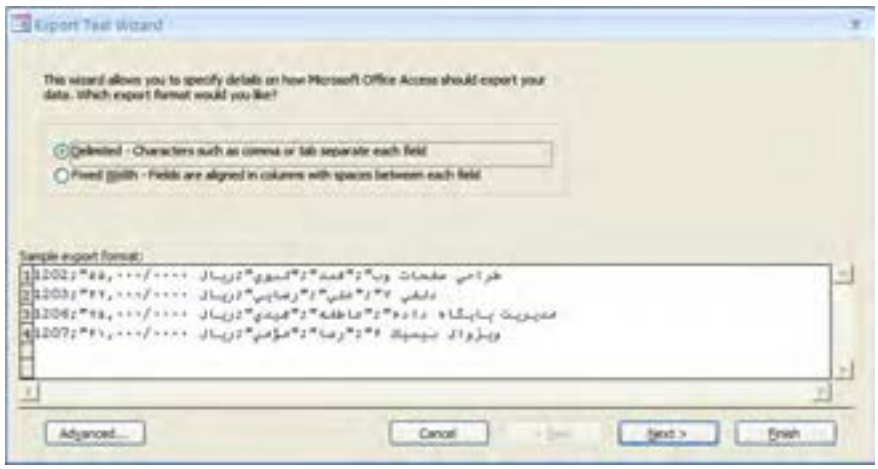
در بسیاری از موارد، انتقال اطلاعات میان دو پایگاه داده متفاوت و غالباً ناسازگار با واسطه یک فایل متنی انجام می‌شود.

۱. جدول موردنظر را در محیط Access باز کنید. مثال این بخش، صدور جدول «کتاب‌ها» به یک فایل متنی است.

۲. در زبانه External Data و قاب Export روی دکمه Text File کلیک نمایید.

۳. محل ذخیره‌سازی فایل متنی را مشخص کنید.

۴. پنجره زیر نشان داده می‌شود.

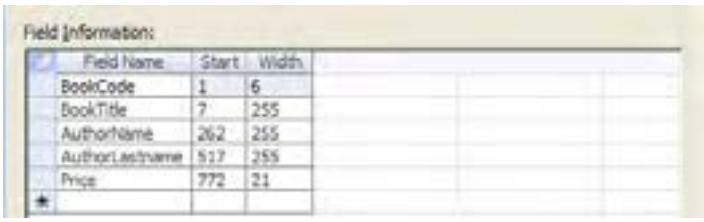


این‌بار می‌خواهیم به جای استفاده از جداکننده برای تفکیک فیلدهای هر رکورد، روش طول ثابت را به کار ببریم.

۵. گزینه Fixed Width را انتخاب نمایید.

بخشی از فیلدها از محدوده دید خارج می‌شوند چون طول درنظر گرفته شده برای آن‌ها زیاد است؛ مثلاً به صورت پیش‌فرض برای فیلدهای متنی (Text) طول ۲۵۵ تعیین می‌شود.

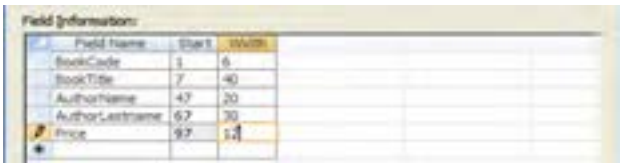
۶. روی دکمه Advanced کلیک کنید تا پنجره زیر ظاهر شود.



در جدول Field Information سه ستون وجود دارد :

- **ستون اول** : نام فیلد
- **ستون دوم** : محل شروع درج فیلد
- **ستون سوم** : طول فیلد

۷. طول فیلدها را روی مقدار مناسب تنظیم کنید.



۸. از لیست Code Page گزینه UTF-8 را انتخاب کنید تا مشکلی برای نمایش نویسه‌های فارسی در فایل متنی ایجاد نشود. نهایتاً روی دکمه OK کلیک نموده و به پنجره اصلی برگردید.

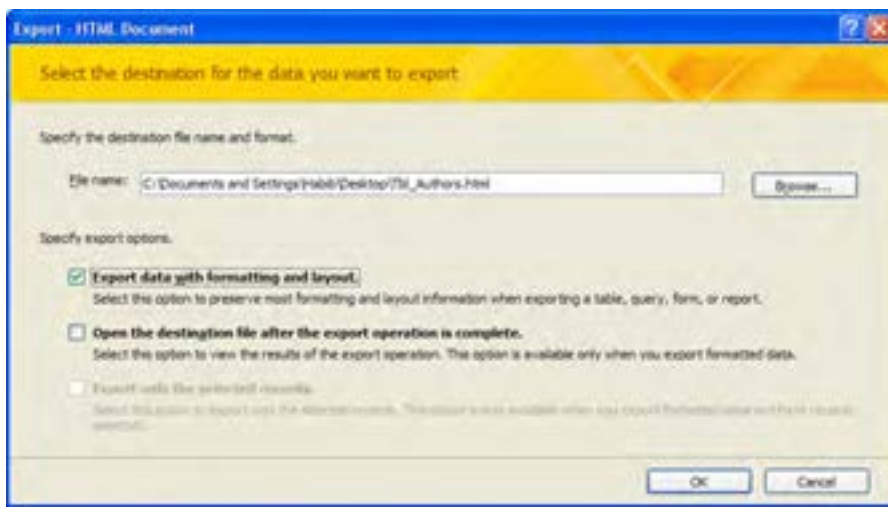
۹. با کلیک کردن روی دکمه Finish، عملیات Export انجام می‌شود.

د) صدور یک جدول به یک فایل HTML

اگر می‌خواهید اطلاعات یکی از جداول پایگاه داده را مستقیماً روی اینترنت قرار دهید یا آن را به فرمت HTML که ســـازگار با اغلب سیستم‌های مدیریت پایگاه داده است درآورید، باید آن را به یک فایل HTML صادر کنید.

۱. جدول موردنظر را در Access باز و در قاب Export، لیســـت More را باز کنید. سپس گزینه HTML Document را باز نمایید.

۲. با کلیک روی دکمه Browse، محل ذخیره‌سازی فایل html را تعیین کنید.



۳. گزینه Export data with formatting and layout را تیک بزنید تا قالب‌بندی[1] اعمال شـــده در جدول Access روی فایل HTML منعکس شود.

۴. روی دکمه OK کلیک کنید.

۵. با ظاهر شدن پنجره تعیین کدگذاری، حالت UTF-8 را انتخاب نمایید.

۶. روی دکمه OK کلیک کنید تا فایل html ساخته شود.

۳-۲ فیلدهای Hyperlink

نوع داده‌ای Hyperlink[2] یکی از انواع داده‌ای موجود در Access به شمار می‌رود و می‌تواند به یک نشانی

1 . Formatting

۲ . اَبَر پیوند

اینترنتی، به یکی از اجزاء پایگاه داده یا فایلی از برنامه‌ای دیگر متصل شـــود. داده‌های Hyperlink در اصل چهار بخشـــی و به صورت زیر هســـتند اما غیر از Address، ورود سه بخش دیگر اختیاری است و داده‌های شما را گویاتر می‌کند.

```
Text to display#Address#SubAddress#ScreenTip
```

بخش Text to Display، عبارت قابل نمایش را نشان می‌دهد و چنان‌چه اشاره‌گر ماوس روی فیلد برود، جمله‌ای که در بخش ScreenTip قرار گرفته نشان داده می‌شود. با کلیک روی پیوند هم به نشانی درج شده منتقل خواهید شد. ضمناً از SubAddress برای مشخص کردن نقطه‌ای درون صفحه وب استفاده می‌شود.

۲-۴ تعیین و اضافه کردن Hyperlink در فیلدها

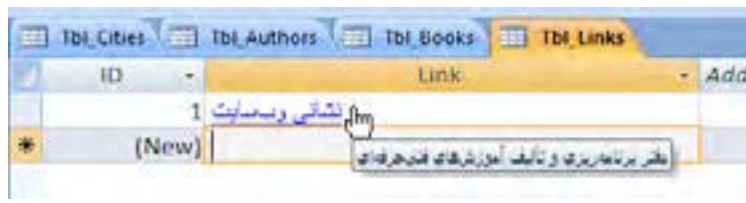
در یک فیلد از نوع داده‌ای Hyperlink می‌توان پیوندهایی را به صورت زیر ایجاد کرد :

الف) پیوند به یک نشانی اینترنتی

۱. جدول را در حالت نمایش Data Sheet قرار داده و عبارت زیر را در فیلد Hyperlink وارد نمایید:

```
دفتر برنامه‌ریزي و تألیف آموزش‌هاي فني‌حرفه‌اي##http://www.tvoccd.sch.ir#نشاني وب‌سایت
```

۲. کلید Enter را بزنید. همان‌طور که در تصویر زیر می‌بینید پیوند برقرار می‌شود.



یک روش سریع برای ویرایش محتوای فیلدهای Hyperlink این است که :

۱. درون فیلد راست‌کلیک کرده و از منوی Hyperlink دستور Edit Hyperlink را انتخاب نمایید.

۲. در پنجره‌ای که ظاهر می‌شود می‌توانید اجزاء مختلف یک داده Hyperlink را تنظیم کنید.

۳. برای ظاهر کردن پنجره ویرایش ScreenTip کافی است روی دکمه‌ای که همین عبارت روی آن نقش بسته کلیک نمایید.

ب) پیوند به یکی از اجزاء پایگاه داده دیگر

برای ایجاد پیوند به فرم Customer List در پایگاه داده Northwind 2007که (در این مثال) درون درایو C ذخیره شده، باید در فیلدی از نوع Hyperlink، عبارت زیر را وارد نمایید :

برای نمایش لیست مشتریان کلیک کنید.

#فرم مشتریان#c:\Northwind 2007.accdb#Customer List#

ج) پیوند به یک فایل از برنامه دیگر

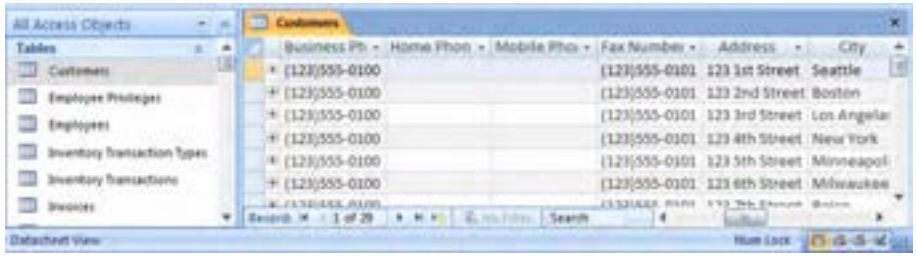
با اســـتفاده از قابلیت نوع‌داده‌ای Hyperlink می‌توان پیونـــدی را به یک فایل خارج از برنامه (مثلاً یک فایل متنی) و یا حتی به بخشی خاص از یک فایل (به عنوان نمونه یکی از اسلایدهای یک فایل پاورپوینت) ایجاد کرد.

بـــرای مثال وارد کردن عبـــارت زیر در یک فیلد از نوع Hyperlink باعث ایجاد پیوند به Sheet2 از فایل اکسل Book.xslx می‌شود که در درایو D قرار دارد و سلول E5 را در حالت ویرایش قرار می‌دهد.

موجودی کتاب در انبار#D:\Books.xlsx#Sheet2!E5#Click to view

۲-۵ Freeze کردن فیلدهای یک جدول

در جداولی که تعدادی فیلدها (ستون‌ها) بیش از حد معمول است، غالباً تعدادی از فیلدهای یک رکورد خارج از محدوده پنجره قرار می‌گیرند و لذا برای مشاهده آن‌ها باید نوارهای پیمایش[1] را جابه‌جا کنید. برای مثال در پایگاه داده Northwind و جدول مشخصات مشتریان (Customers)، برای مشاهده شماره تلفن‌های مشـــتریان باید نوار پیمایش افقی را به سمت راست حرکت دهید. این کار یک عیب بزرگ دارد؛ ستون‌های ابتدایی جدول که حاوی مشـــخصات اصلی هستند از دید خارج می‌شوند و بنابراین کار ردگیری اطلاعات و ویرایش آن‌ها دشوار می‌شود.

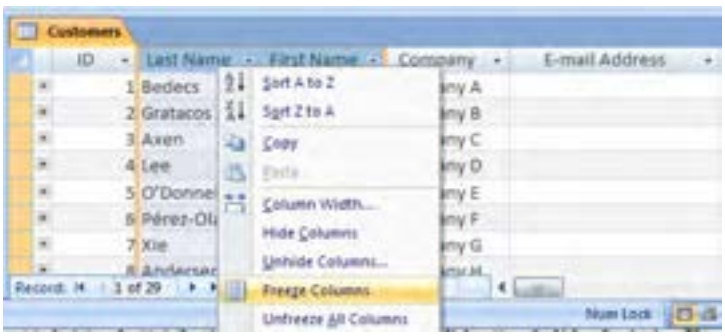


1 . Scroll Bars

برای حل این مشکل، روشی به نام Freeze کردن ستون‌ها پیش‌بینی شده است.

۱. ستون یا ستون‌هایی را که می‌خواهید هنگام حرکت دادن نوارهای پیمایش جابه‌جا نشوند انتخاب کنید. این کار با کلیک کردن روی سرستون اول و حرکت دادن اشاره‌گر به سمت ستون‌های بعدی انجام می‌شود.

۲. سپس روی یکی از این ستون‌ها راست‌کلیک کرده و عبارت Freeze Columns را انتخاب کنید.



۳. ستون‌های Freeze شده به سمت چپ جدول منتقل می‌شوند و از این پس با حرکت دادن نوارهای پیمایش، جابه‌جا نمی‌شوند. به این ترتیب فیلدهای مهم هر رکورد همیشه در ابتدای سطر آن رکورد قرار دارند.

۴. برای خارج کردن ستون‌ها از حالت Freeze، روی یکی از آن‌ها راست‌کلیک نموده و گزینه Unfreeze All Columns را انتخاب کنید.

Understand importing and linking to data from another Access database

When you import from another database, Access creates a copy of the data or objects in the destination database without altering the source. During the import operation, you can choose the objects you want to copy, control how tables and queries are imported, specify whether relationships between tables should be imported, and so on.

Common scenarios for importing data or objects from an Access database

You typically import data for the following reasons:

- You want to merge two databases by copying all the objects in one database to another. When you import, you can copy all the tables, queries, forms, reports, macros, and modules, along with table relationships, to another database in a single operation.

You need to create some tables that are similar to tables that exist in another database. You might want to copy the entire table or just the table definitions to avoid manually designing each of these tables. When you choose to import only the table definition, you get an empty table. In other words, the fields and field properties are copied to the destination database, but not the data in the table. Another advantage of importing (compared to a copy-paste operation) is that you can choose to import the relationships between the tables along with the tables themselves.

You need to copy a set of related objects to another database. For example, you want to copy the Employees table and the Employees form to a second database. Importing enables you to copy an object and all of its related objects to another database in a single operation.

۱. متن بالا را مطالعه کرده و در مورد آن توضیح دهید.

۲. مزایای Importing داده‌ها را نسبت به عملیات کپی کردن توضیح دهید.

۳. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. توانایی Import و Export داده‌ها در محیط اکسس چه مزیت‌هایی برای کاربر پایگاه داده دارد؟

۲. در اکسس جدولی حاوی مشخصات دانش‌آموزان یک کلاس ایجاد و تعدادی رکورد در آن درج کنید. سپس این جدول را به یک فایل متنی، یک فایل HTML و کارپوشه Excel صادر کنید.

۳. در هنگام Import داده‌ها به اکسس، با دو گزینه Delimited و Fixed Width روبرو می‌شوید. تفاوت این دو گزینه در چیست؟

۴. Freeze کردن ستون‌های جدول در چه مواردی کاربرد دارد؟

فصل سوم

فصل سوم : ایجاد پرس‌وجوهای[1] محاسباتی و عملیاتی

برای ذخیره‌سازی داده‌ها درون جداول یک پایگاه داده رابطه‌ای و استخراج نتایج موردنظر از آن‌ها، روش‌های متعددی وجود دارد. برای مثال شما می‌توانید رکوردهای موجود در یک جدول Access را به صورت مستقیم یا به واسطه یک فرم اکسس تغییر دهید. همچنین می‌توانید با استفاده از ابزارهای گزارش‌سازی، نتایج دلخواه را از مجموعه داده‌ها به دست آورید.

اما روش استاندارد و معمول برای انجام این کار به‌خصوص در نرم‌افزارهای کاربردی که با پایگاه داده در ارتباط هستند، استفاده از پرس‌وجوهایی است که به زبان SQL [2] نوشته می‌شوند. SQL که «زبان پرس‌وجوی ساخت‌یافته» هم نامیده می‌شود در پایگاه داده‌هایی مثل Access، SQL Server و Oracle کاربرد دارد و می‌تواند برای انجام یک عملیات (مثل درج یک رکورد یا ایجاد یک جدول) و یا انجام یک محاسبه (مانند محاسبه میانگین تعدادی عدد) مورد استفاده قرار گیرد.

پیش از شروع این فصل لازم است مثالی را که در فصل اول برای طراحی پایگاه داده بررسی کردیم، تکمیل نموده و جداول، فیلدها و ارتباطات میان آن‌ها را دقیقاً مشخص کنیم تا بتوانید عمل‌کرد پرس‌وجوهای ذکر شده در این فصل و همچنین مطالب فصول بعدی را بهتر متوجه شوید. علاوه بر این می‌توانید با پیاده‌سازی پایگاه داده در محیط نرم‌افزار Access و وارد کردن داده‌های آزمایشی، پرس‌وجوهای توضیح داده شده در این کتاب را روی رایانه خودتان اجرا نمایید.

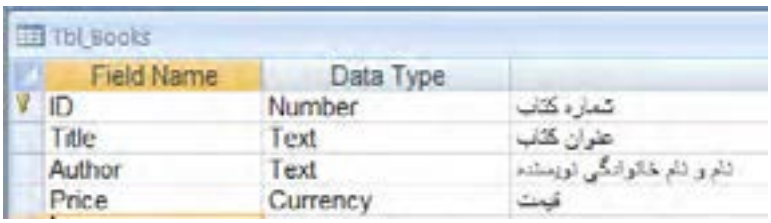
---

1 . Query

2 . Structured Query Language

پایگاه داده Publisher مخصوص نگه‌داری اطلاعات فروش کتاب‌های چاپ شـــده توســـط یک مؤسسه انتشاراتی است و چهار جدول به صورت زیر دارد :

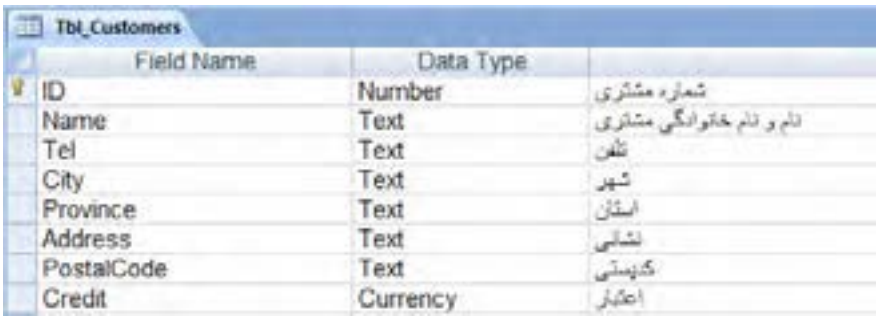
جدول کتاب‌ها



Tbl_Books

	Field Name	Data Type	
🔑	ID	Number	شماره کتاب
	Title	Text	عنوان کتاب
	Author	Text	نام و نام خانوادگی نویسنده
	Price	Currency	قیمت

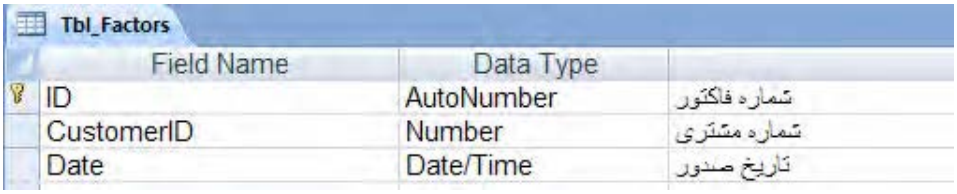
جدول مشتریان



Tbl_Customers

	Field Name	Data Type	
🔑	ID	Number	شماره مشتری
	Name	Text	نام و نام خانوادگی مشتری
	Tel	Text	تلفن
	City	Text	شهر
	Province	Text	استان
	Address	Text	نشانی
	PostalCode	Text	کدپستی
	Credit	Currency	اعتبار

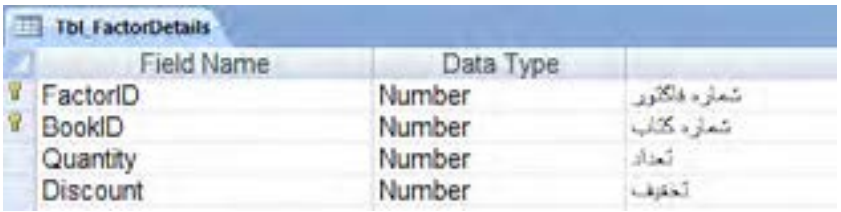
جدول فاکتورها



Tbl_Factors

	Field Name	Data Type	
🔑	ID	AutoNumber	شماره فاکتور
	CustomerID	Number	شماره مشتری
	Date	Date/Time	تاریخ صدور

جدول جزییات فاکتور



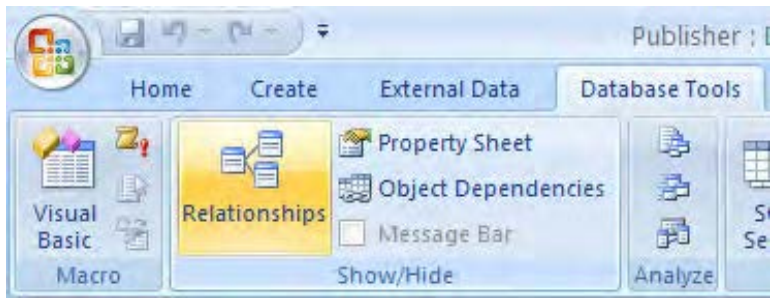
Tbl_FactorDetails

	Field Name	Data Type	
🔑	FactorID	Number	شماره فاکتور
🔑	BookID	Number	شماره کتاب
	Quantity	Number	تعداد
	Discount	Number	تخفیف

پیش از ادامه کار باید ارتباطات[1] میان جداول را مشـــخص کنیم؛ یعنی تعیین نماییم که چه فیلدی از یک جدول با فیلدی از جدول دیگر در ارتباط است. دو جدول «مشتریان» و «فاکتورها» را درنظر بگیرید. در جدول «مشـــتریان»، هر مشتری یک شماره دارد که در فیلد ID ذخیره می‌شود. در جدول «فاکتورها» هم برای مشخص کردن این که فاکتور برای کدام مشتری صادر شده، شماره مشتری در فیلد CustomerID نگـــه‌داری می‌گردد. در هنگام ایجاد پایـــگاه داده باید بین این نوع فیلدها ارتبـــاط ایجاد کنیم تا اولاً ایجاد پرس‌وجوهایی که نتایج را از دو یا چند جدول برمی‌گردانند، با ســـرعت و سهولت انجام شود و ثانیاً از درج داده‌های بی‌معنی در جداول جلوگیری شود.

برای نمونه فرض کنید در جدول «مشـــتریان»، شماره‌های ۱ تا ۱۰۰ به مشتری‌ها اختصاص داده شده است. حال اگر در جدول فاکتورها برای یک مشتری به اشتباه شماره ۱۰۱ را وارد کنیم، در واقع برای کسی فاکتور صادر کرده‌ایم که در جدول مشـــتریان ما وجود ندارد و این کار باعث ایجاد رکورد بی‌معنی در پایگاه داده خواهد شد.

برای ایجاد ارتباط میان جداول، در زبانه Database Tools بر روی دکمه Relationships کلیک کنید.



سپس در پنجره‌ای که ظاهر می‌شود، بر روی دکمه Show Table کلیک نمایید.

1 .Relationships

در ادامه، جداول موردنظر را انتخاب و روی دکمه Add کلیک کنید تا وارد صفحه شوند.

بر روی فیلد دلخواه کلیک و آن را بر روی فیلد متناظر درون جدول دیگر بکشـــید و رها کنید. ســـپس در پنجره‌ای که ظاهر می‌شـــود، سه گزینه موجود را تیک بزنید و روی دکمه OK کلیک نمایید تا ارتباط ایجاد شود.

همین کار را برای ایجاد ارتباط میان ســـایر جداول تکرار کنید تـــا ارتباطاتی مانند آن چه در تصویر صفحه بعد مشـــاهده می کنید ایجاد گردد. در پایان هم با کلیک بر روی دکمه  ، تغییرات ایجاد شده را ذخیره نمایید. شـــاید این پرسش برای شما پیش بیاید که هر یک از گزینه‌های موجود در این پنجره چه کاربردی دارند. این پرسش در بخش «پرس‌وجوی بروزرسانی داده‌ها» در ادامه همین فصل بررسی شده است.

ارتباطات جداول در این پایگاه داده به صورت زیر است:

۳-۱ انجام محاسبات در یک پرس‌وجو

برای انجام محاســـبات روی داده‌های درون پایگاه داده مانند محاسبه مجموع یا میانگین تعدادی عدد و همچنین تطبیق دادن داده‌ها با یک معیار[1] خاص باید از عمل‌گرهای محاسباتی استفاده کنید.

۳-۱-۱ عمل‌گرهای مقایسه‌ای

در پرس‌وجوهای SQL، برای بررســـی مقادیر موجود در رکوردهای یک جدول، عمل‌گرهای ریاضی زیر پیش‌بینی شده است.

عمل‌گر	کارکرد	مثال
=	مساوی با	= “Tehran”
<	کوچک‌تر از	< 25
>	بزرگ‌تر از	>25
<=	کوچک‌تر یا مساوی	<= 25
>=	بزرگ‌تر یا مساوی	>= 25
<>	مخالف (نامساوی)	<>20

می‌خواهیم پرس‌وجویی طراحی کنیم که نام و تلفن مشتریان ساکن در شهر «شیراز» را به دست آورد. برای انجام این کار :

۱. پایگاه داده Publisher را باز کنید.

1 . Criteria

۲. در زبانه Create و قاب Other روی دکمه Query Design کلیک کنید.

۳. بلافاصله پنجره‌ای باز می‌شود که حاوی جداول موجود در پایگاه داده است. جدول مشتریان (Tbl_Customers) را انتخاب و روی دکمه Add کلیک کنید.

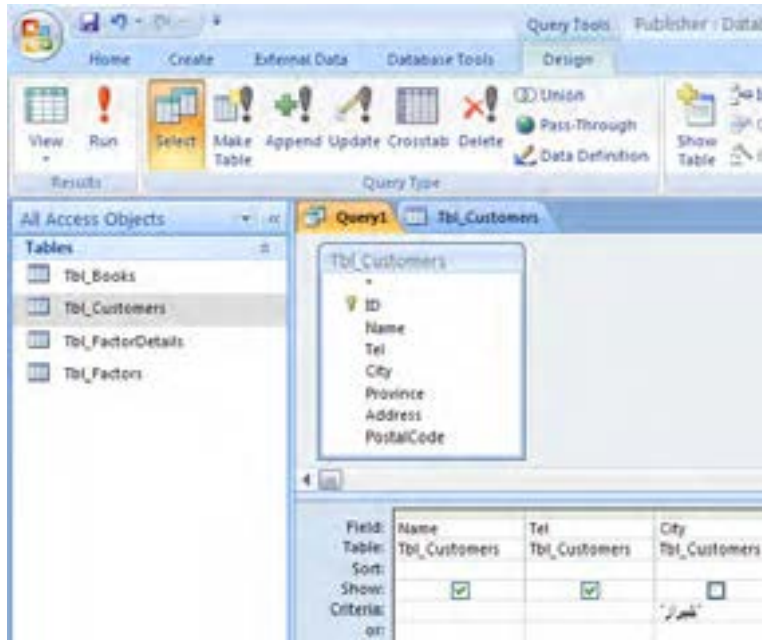
۴. برای ساخت این پرس‌وجو فقط به جدول مشتریان نیاز داریم، بنابراین با کلیک روی دکمه Close پنجره Show Table را ببندید.

۵. در شبکه‌ای که زیر جدول انتخاب شده قرار دارد، در ستون اول و ردیف Filed کلیک کنید تا گزینه‌های قابل انتخاب نشان داده شوند. گزینه Name را انتخاب کنید. همین کار را برای دو ستون دیگر هم انجام داده و این بار گزینه‌های Tel و City را انتخاب کنید.

در ردیف Table، نام جدولی که فیلدها به آن تعلق دارند به صورت خودکار درج می‌شود چون صرفاً یک جدول را وارد پرس‌وجو کرده‌ایم.

۶. در ردیف Criteria و ستون City کلمه شیراز را وارد کنید؛ به این معنی که معیار جداسازی رکوردها، برابر بودن فیلد City آن‌ها با مقدار «شیراز» است.

۷. از آن‌جا که در نتیجه نهایی فقط باید نام و شـــماره تلفن مشـــتری نشان داده شود، در ردیف Show، علامت تیک را برای ستون City بردارید.



۸. در زبانه Design و قاب Results روی دکمه Run کلیک کنید تا پرس‌وجو اجرا شود.

۹. نتایج حاصل از اجرای پرس‌وجو در یک شـــبکه نشان داده می‌شود. برای برگشت به نمای طراحی باید در زبانه Home و قاب Views روی عبارت View کلیک کنید تا منوی انتخاب نما ظاهر شود. سپس گزینه Design View را انتخاب کنید.

۱۰. از آن‌جا که قصد تغییر پرس‌وجوی طراحی شـــده را نداریم، با کلیک روی گزینه SQL View به نمای کدنویسی می‌رویم تا ببینیم در پشت صحنه ساخت یک پرس‌وجو چه کدهایی تولید شده است. برای ساخت این پرس‌وجو، کدهای زیر توسط اکسس ایجاد شده است:

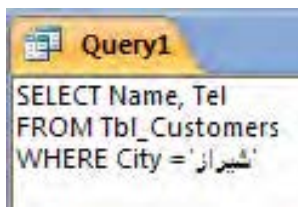
```
SELECT Tbl_Customers.Name, Tbl_Customers.Tel
FROM Tbl_Customers
WHERE (((Tbl_Customers.City)='شیراز'));
```

در واقـــع نمای طراحی به شـــما کمک می‌کند تا بدون درگیر شـــدن با جزییات نـــگارش کدها بتوانید پرس‌وجوی موردنظر را ایجاد کنید. اما دقت داشـــته باشـــید که استفاده از نمای طراحی مخصوص کاربران مبتدی است و به تدریج در طول این کتاب می‌آموزید که چگونه بدون نیاز به نمای طراحی، پرس‌وجوهای موردنظر را ایجاد نمایید.

۱۱. کد زیر را که ســـاده شده پرس‌وجوی فوق است در نمای SQL وارد و روی دکمه Run کلیک کنید تا اجرا شود.

```
SELECT Name, Tel
FROM Tbl_Customers
WHERE City = 'شیراز'
```

برای مقایســـه یا تغییر فیلدهایی که داده‌های درون آن‌ها از نوع رشـــته‌ای هســـتند باید هنگام نوشتن پرس‌وجو، رشـــته را درون علامت‌های نقل‌قول (' ') قرار دهید که در محیط ویرایش پرس‌وجوی Access یا SQL Server به صورت ' ' در اطراف عبارت ظاهر می‌شوند.



مثال ۱: پرس‌وجویی بسازید تا عنوان کتاب‌هایی را استخراج کند که قیمت آن‌ها ۵۰۰۰۰ ریال یا بیش‌تر است.

برای انجام این کار باید مطابق توضیحات قبل، با کلیک روی دکمه Query Design یک پرس‌وجوی جدید ایجاد و جدول Tbl_Books را وارد بخش جداول کنید. سپس فیلدهای Title و Price را انتخاب نموده و در ردیف Criteria از ستون Price، شرط >= 50000 را وارد کنید تا تنها کتاب‌هایی که قیمت آن‌ها با این شرط تطابق دارد انتخاب شوند.

پرس‌وجویی که با این کار تولید می‌شود به صورت زیر است :

```
SELECT Tbl_Books.Title
FROM Tbl_Books
WHERE Tbl_Books.Price >= 50000
```

۲-۱-۳ عمل‌گرهای منطقی[1]

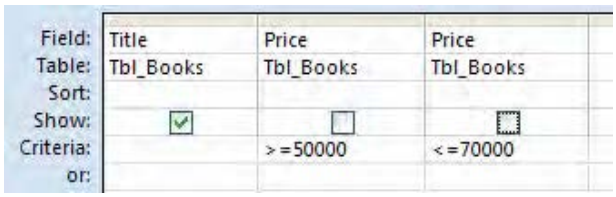
در بخش قبل با روش ایجاد یک شرط در پرس‌وجو آشنا شدید. حالا این سؤال مطرح می‌شود که اگر بخواهیم مقادیر فیلدها را با بیش از یک شرط ارزیابی کنیم راه‌حل چیست؟ برای انجام این کار مجموعه‌ای از عمل‌گرهای منطقی طراحی شده که به صورت زیر عمل می‌کنند.

AND : مانند «و» منطقی عمل می‌کند یعنی باید همه شرط‌ها درست باشند تا کل عبارت درست باشد.

1 . Logical

مثـــال ۱: پرس‌وجویی طراحی کنید تا عنوان کتاب‌هایی را برگرداند که قیمت آن‌ها بین ۵۰۰۰۰ ریال و ۷۰۰۰۰ ریال است.

در ایـــن حالـــت باید فیلد موردنظر را در دو ســـتون انتخاب نموده و در ردیف معیـــار هر کدام، یکی از شرط‌های موجود در AND منطقی را قرار دهیم.



Field:	Title	Price	Price
Table:	Tbl_Books	Tbl_Books	Tbl_Books
Sort:			
Show:	☑	☐	☐
Criteria:		>=50000	<=70000
or:			

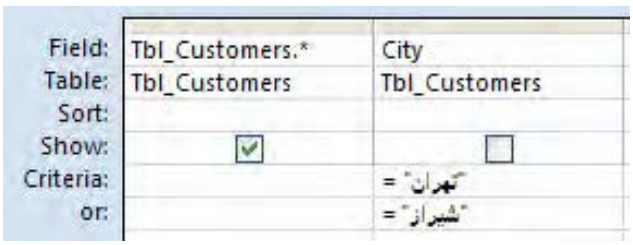
کد تولید شده برای این پرس‌وجو به صورت زیر است:

```
SELECT Title FROM Tbl_Books
WHERE Price>=50000 AND Price<=70000
```

OR : همان «یا»ی منطقی اســـت و درست بودن فقط یک شـــرط برای صحیح بودن کل عبارت کافی است.

مثال ۲: پرس‌وجویی بنویســـید که مشخصات مشتریان ساکن در شهر «شیراز» یا «تهران» را به دست آورد.

از آن‌جا که قصد داریم همه مشـــخصات یک مشتری برگردانده شـــود یا باید تک‌تک فیلدها را انتخاب کنیم و یا با انتخاب گزینه (*. نام جدول) به اکسس اعلام کنیم که همه فیلدها را انتخاب کند. در ادامه باید فیلدی را که در انتخاب رکوردها تأثیر دارد به صورت جداگانه در یک ســـتون جدید انتخاب نموده و شـــرط اول را در ردیف Criteria و شرط بعدی را در ردیف or درج کنیم.



Field:	Tbl_Customers.*	City
Table:	Tbl_Customers	Tbl_Customers
Sort:		
Show:	☑	☐
Criteria:		"تهران" =
or:		"شیراز" =

کد تولید شده برای این پرس‌وجو به صورت زیر است:

```
SELECT * FROM Tbl_Customers
WHERE City='شیراز' OR City='تهران'
```

این عمل‌گرهای منطقی را می‌توان با هم ترکیب کرد و برای جداسازی شرط‌ها از هم، از علامت پرانتز استفاده نمود.

مثال ۳: می‌خواهیم مشخصات کتاب‌های «فتوشاپ» یا «اینترنت» منتشر شده با قیمت بین ۵۰۰۰۰ تا ۷۰۰۰۰ ریال را استخراج کنیم.

```
SELECT * FROM Tbl_Books
WHERE Price>=50000 AND Price<=70000 AND (Title='فتوشاپ' OR Title='اینترنت')
```

مثال ۴: اگر در پرس‌وجوی مثال قبل، پرانتزها را از اطراف دو شرط OR حذف کنیم چه اتفاقی می‌افتد؟

وقتی پرس‌وجو به صورت زیر نوشته می‌شود:

```
SELECT * FROM Tbl_Books
WHERE Price>=50000 AND Price<=70000 AND Title='فتوشاپ' OR Title='اینترنت'
```

شرط 'Title='اینترنت به صورت مستقل مورد ارزیابی قرار می‌گیرد؛ بنابراین کتاب‌های فتوشاپی که قیمت آن‌ها در محدوده موردنظر است لیست می‌شوند و در کنار آن‌ها مشخصات همه کتاب‌های اینترنت (صرف‌نظر از قیمت آن‌ها) نشان داده می‌شود. در واقع پرس‌وجو توسط اکسس به صورت زیر تفسیر می‌شود:

```
SELECT * FROM Tbl_Books
WHERE (Price>=50000 AND Price<=70000 AND Title='فتوشاپ') OR Title='اینترنت'
```

این مسأله از تقدم عمل‌گر AND بر OR ناشی می‌شود و نشان می‌دهد با استفاده از پرانتز می‌توان این تقدم را تغییر داد.

۳-۱-۳ عمل‌گر IN

با روش بررسی مقادیر موجود در پایگاه داده با استفاده از دستور OR آشنا شدید. مثلاً برای بازیابی مشخصات مشتریانی که در شهرهای شیراز، اصفهان یا شهرکرد سکونت دارند می‌توان این پرس‌وجو را نوشت:

```
Select * From Tbl_Customers
WHERE City='شهرکرد' OR City='اصفهان' OR City='شیراز'
```

عمل‌گر IN به شما کمک می‌کند تا معیارهایی از این دست را با سهولت بیش‌تری به پرس‌وجو اضافه کنید.

در پرس‌وجوی زیر بررسی می‌شود که آیا مقدار فیلد City با یکی از اعضای مجموعه‌ای درون پرانتز مساوی هست یا خیر.

```
Select * From Tbl_Customers
WHERE City IN ('شهرکرد' , 'اصفهان' , 'شیراز')
```

با توجه به مهارت نسبی که در نوشتن پرس‌وجوها در محیط اکسس پیدا کرده‌اید، از این پس، برای توضیح اغلب پرس‌وجوها به ذکر کد SQL آن‌ها اکتفا خواهیم کرد.

برای نوشتن و اجرای کد SQL در محیط اکسس باید به روش زیر عمل کنید :

۱. در زبانه Create و قاب Other روی دکمه Query Design کلیک کنید.
۲. با ظاهر شدن پنجره انتخاب جداول، روی دکمه Close کلیک کنید تا بسته شود.
۳. در زبانه Design و قاب Results روی عبارت SQL کلیک کنید تا محیط کدنویسی ظاهر شود.

۴. پس از نوشتن کد، روی دکمه Run کلیک کنید تا پرس‌وجو اجرا شود.

۳-۱-۴ عمل‌گر BETWEEN

در مطالب قبلی با روش بازیابی رکوردهایی که یکی از فیلدهای آن در محدوده‌ای مشخص قرار دارد آشنا شده‌اید. برای مثال پرس‌وجوی زیر مشخصات کتاب‌هایی را برمی‌گرداند که قیمتی بین ۵۰۰۰۰ و ۷۰۰۰۰ ریال دارند.

```
SELECT * FROM Tbl_Books
WHERE Price>=50000 AND Price<=70000
```

عمل‌گر BETWEEN این امکان را فراهم می‌آورد تا این پرس‌وجو را به صورت زیر بنویسید.

```
SELECT * FROM Tbl_Books
WHERE Price BETWEEN 50000 AND 70000
```

یکی از مهم‌ترین کاربردهای این عمل‌گر، برگرداندن رکوردهایی است که فیلد تاریخ آن‌ها در بازه‌ای خاص قرار گرفته است. برای نمونه، پرس‌وجوی زیر مشخصات فاکتورهایی را برمی‌گرداند که از ابتدای مرداد ۸۸ تا انتهای آذر ۸۹ صادر شده‌اند.

```
SELECT * FROM Tbl_Factors
WHERE Date BETWEEN#1388/05/01#AND#1388/09/30#
```

درج علامت # در ابتدا و انتهای تاریخ‌ها ضروری است وگرنه اجرای پرس‌وجو با مشکل مواجه می‌شود.

۳-۱-۵ عمل‌گر LIKE

این عمل‌گر ابزاری کارآمد برای جستجو در میان فیلدها و پیدا کردن موارد مشابه با عبارت موردنظر است. در این عمل‌گر با قرار دادن علامت * می‌توانید فیلد موردنظر را با مجموعه‌ای از نویسه‌ها (با طول نامشخص) مقایسه کنید. به عنوان نمونه عبارت LIKE 's*' عبارت‌هایی را بررسی می‌کند که با حرف s شروع می‌شوند اما طول آن‌ها اهمیت ندارد. در عوض عبارت LIKE 's???' به دنبال عبارت‌هایی می‌گردد که با حرف s شروع می‌شوند اما فقط چهار حرف دارند.

مثال ۱: مشخصات کتاب‌هایی را به دست بیاورید که نام نویسنده آن‌ها «رضا» باشد.

```
SELECT * FROM Tbl_Books
WHERE Author LIKE 'رضا*'
```

مثال ۲: مشخصات کتاب‌هایی را بازیابی کنید که در عنوان آن‌ها واژه «برنامه نویسی» باشد.

```
SELECT * FROM Tbl_Books
WHERE Title LIKE '*برنامه نویسی*'
```

Tbl_Customers | Tbl_FactorDetails | Query1 | Tbl_Factors | Tbl_Books

ID	Title	Author	Price
1206	برنامه نویسی به زبان C#	محمد محمدی	85,000
1212	مهارت های پایه در برنامه نویسی	رضا اکرمی	68,000
1203	برنامه نویسی به زبان دلفی	منصور کبیری	62,000
1205	برنامه نویسی به زبان ویژوال بیسیک 6	رضا رحمانی	61,000

مثال ۳: لیست کتاب‌هایی را به دست آورید که شماره آن‌ها چهار رقمی و رقم دهگان ۲، ۳ یا ۴ باشد.

```
SELECT * FROM Tbl_Books
WHERE ID LIKE '??[2-4]?'
```

۳-۱-۶ ایجاد فیلدهای محاسباتی

یکی از قابلیت‌های بسیار پرکاربرد پرس‌وجوها، توانایی در ایجاد فیلدهای محاسباتی است. شما با استفاده از عمل‌گرهای ساده ریاضی می‌توانید روی مجموعه‌ای از فیلدهای عددی محاسبه موردنظر را انجام

داده و نتیجه را در فیلد جدیدی ذخیره کنید.

جدول جزییات فروش را درنظر بگیرد. می‌خواهیم پرس‌وجویی ایجاد کنیم که یک فیلد محاسباتی به این جدول اضافه نموده و مجموع قیمت هر رکورد را با اعمال تخفیف محاسبه کند.

به جدول جزییات فروش دقت کنید.

Tbl_Customers | Tbl_FactorDetails | Query1 | Tbl_Factors

FactorID	BookID	Qyantity	Discount	Add New Field
1	1202	10	15	
2	1203	15	20	
1	1204	12	14	
1	1205	20	0	
2	1206	30	10	
3	1207	12	20	

قیمت کتاب‌ها در این جدول نگه‌داری نمی‌شود، بنابراین باید با استفاده از شماره کتاب، قیمت آن را از جدول کتاب‌ها به‌دست بیاوریم. برای انجام این کار باید از دو جدول Tbl_FactorDetails و Tbl_Books استفاده نماییم.

۱. در زبانه Create و قاب Others روی دکمه Query Design کلیک کنید تا پنجره طراحی پرس‌وجو ظاهر شود.

۲. جداول Tbl_FactorDetails و Tbl_Books را وارد پنجره کنید.

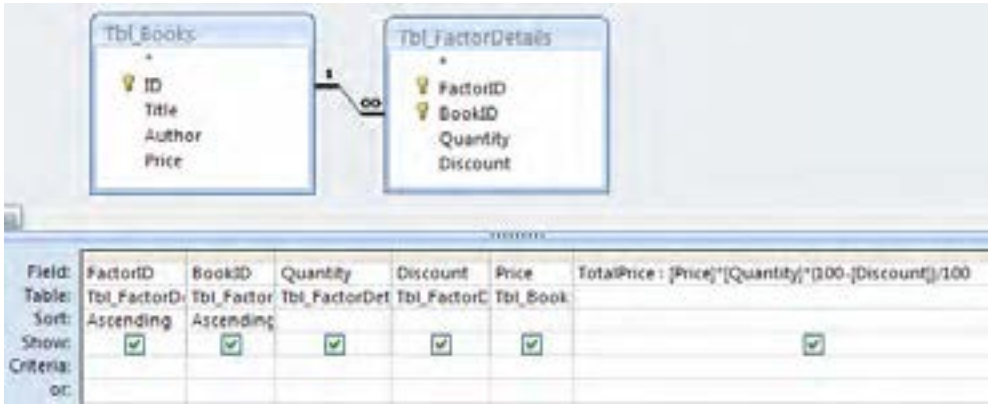
۳. از جدول جزییات فروش، همه فیلدها و از جدول کتاب‌ها فقط فیلد قیمت (Price) را وارد شبکه نمایید.

۴. در فیلد آخر کلیک نموده و عبارت زیر را وارد کنید :

TotalPrice : [Price]*[Quantity]*(100-[Discount])/100

TotalPrice نام فیلد جدید است که با علامت دو نقطه از فرمول روبروی آن جدا شده است.

۵. پرس‌وجو را بر اساس شماره فاکتور و سپس شماره کتاب‌ها به صورت صعودی مرتب کنید.



۶. در زبانه Design روی دکمه Run کلیک کنید تا پرس‌وجو اجرا شود.

۷. نتیجه اجرای پرس‌وجو، جدول زیر است. با استفاده از دستور Save این پرس‌وجو را با نام Qry_TotalPrice ذخیره کنید.

FactorID	BookID	Quantity	Discount	Price	TotalPrice
	1202	10	15	55,000	467500
1	1204	12	14	75,000	774000
1	1205	20	0	60,000	1200000
2	1203	15	20	62,000	744000
2	1206	30	10	85,000	2295000
3	1201	19	15	50,000	807500

استفاده از Expression Builder

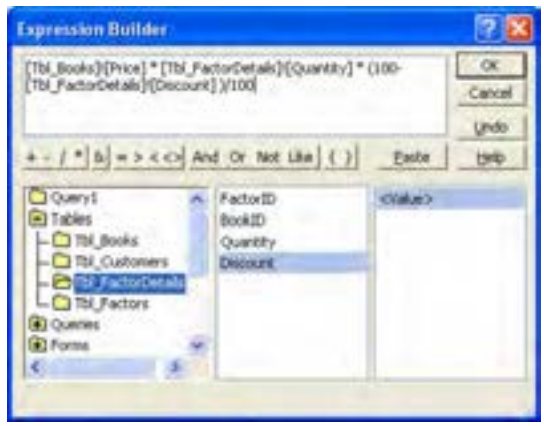
در مرحله ۴ از مثال قبل، فرمول را به صورت دستی در فیلد وارد کردیم اما برای انجام این کار در اکسس یک پنجره مجزا پیش‌بینی شده که بهتر است برای تولید فرمول‌های پیچیده‌تر از آن‌ها استفاده کنید. برای پیاده‌سازی فرمول محاسبه مبلغ کل هر ردیف فاکتور از طریق پنجره Expression Builder به روش زیر عمل نمایید:

۱. درون خانه Field کلیک کنید.

۲. در زبانه Design و قاب Query Setup روی دکمه Builder کلیک نمایید.

۳. پنجـــره زیر ظاهر می‌شـــود. با کلیک کردن روی عبارت Tables لیســـت جـــداول را باز و روی جدول موردنظر کلیک نمایید.

۴. با کلیک کردن روی فیلد موردنظر و سپس فشار دادن دکمه Paste، این فیلد وارد کادر فرمول‌نویسی می‌شود.

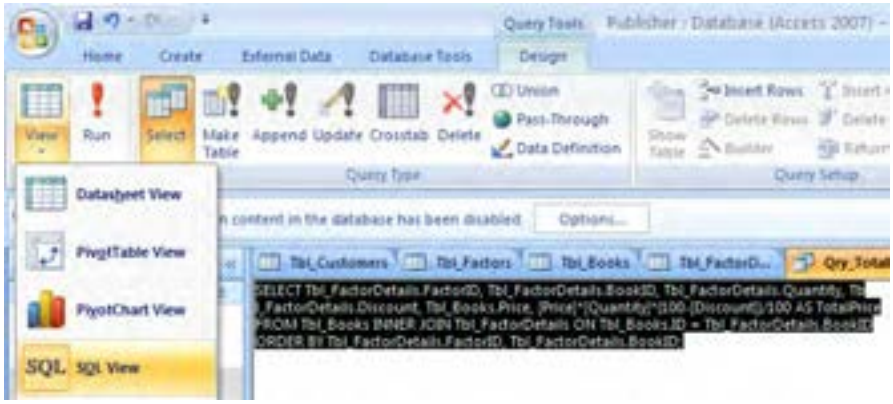


۵. با اســـتفاده از دکمه‌های پایین کادر فرمول‌نویســـی، عمل‌گرهای لازم را وارد نموده و نهایتاً روی دکمه OK کلیک کنید تا فرمول ساخته شده وارد خانه Filed شود.

۶. در ابتدای این فرمول به صورت پیش‌فرض عبارت Exp1 درج می‌شـــود که نام ســـتون جدید محسوب می‌شود و البته می‌توانید آن را تغییر دهید.

بررسی پرس‌وجو

با کلیک کردن روی منوی دکمه View، کد SQL پرس‌وجوی ساخته شده را ظاهر نمایید.



به این پرس‌وجو دقت کنید. کدی که باعث ایجاد ستون محاسباتی جدید شده به صورت زیر است :

نام فیلد جدید AS عبارت محاسباتی

به عنوان نمونه برای ساخت ستون جدیدی که قیمت کتاب‌ها را با اعمال بیست درصد تخفیف در کنار قیمت فعلی آن‌ها نشان دهد می‌توانید از کد زیر استفاده کنید.

```
SELECT Price , Price*0.8 AS NewPrice FROM Tbl_Books
```

۷-۱-۳ استفاده از توابع تجمعی[1]

گاهی اوقات لازم می‌شود برحسب مقادیر موجود در یک فیلد، محاسبه‌ای را روی فیلد دیگری انجام دهید؛ مثلاً مجموع آن فیلد یا تعداد تکرار رکورد مرتبط را در جدول محاسبه نمایید. به جدول حاصل از پرس‌وجوی ساخته شده در بخش قبل نگاهی دوباره بیندازید.

Tbl_Customers | Tbl_Factors | Tbl_Books | Tbl_FactorDetails | Qry_TotalPrice

FactorID	BookID	Quantity	Discount	Price	TotalPrice
	1202	10	15	55,000	467500
1	1204	12	14	75,000	774000
1	1205	20	0	60,000	1200000
2	1203	15	20	62,000	744000
2	1206	30	10	85,000	2295000
3	1201	19	15	50,000	807500

هر فاکتور چند ردیف دارد که قیمت کل هر ردیف در فیلد TotalPrice محاسبه شده است. حالا

1 . Aggregate

می‌خواهیم مبلغ کل یک فاکتور را به دست بیاوریم یعنی TotalPrice همه رکوردهایی را که فیلد FatorID مشابه دارند (مربوط به یک فاکتور هستند) جمع بزنیم. آیا این کار با استفاده از فیلدهای محاسباتی امکان‌پذیر است؟

پاسخ منفی است؛ در این‌گونه مواقع باید از توابع تجمعی که Aggregation هم نامیده می‌شوند استفاده کنیم. برخی توابع پرکاربرد تجمعی در جدول زیر فهرست شده‌اند :

تابع	عمل‌کرد
Min	محاسبه کوچک‌ترین مقدار
Max	محاسبه بزرگ‌ترین مقدار
Sum	محاسبه مجموع
Avg	محاسبه میانگین
Count	شمارش تعداد

مثال ۱: در جدول کتاب‌ها، قیمت گران‌ترین کتاب را به دست آورید.

```
SELECT MAX(Price) AS [بالاترین قیمت]
FROM Tbl_Books
```

خروجی مطابق تصویر، یک خانه از جدول با سرستون ذکر شده پس از کلمه AS است:

نکته

وقتی بین کلمات یک عبارت فاصله وجود داشته باشد باید آن عبارت را درون کروشه قرار دهید تا SQL آن را به عنوان یک عبارت واحد شناسایی کند.

مثال ۲: پرس‌وجویی بنویسید که نشان دهد چند جلد از کتاب شماره ۱۲۰۶ فروخته شده است.

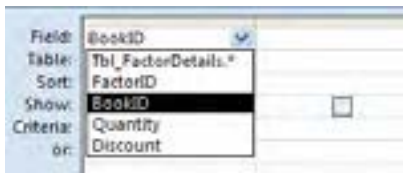
```
SELECT SUM(Quantity) AS [آمار فروش]
FROM Tbl_FactorDetails
WHERE BookID=1206
```

مثال ۳ : پرس‌وجویی بنویسد که آمار فروش همه کتاب‌ها را با ذکر شماره آن‌ها استخراج نماید.

۱. در زبانه Create و قاب Other روی دکمه Query Design کلیک کنید.

۲. جدول Tbl_FactorDetails را وارد پنجره طراحی نمایید.

۳. در ستون اول شبکه و از منوی Filed گزینه BookID را انتخاب کنید.

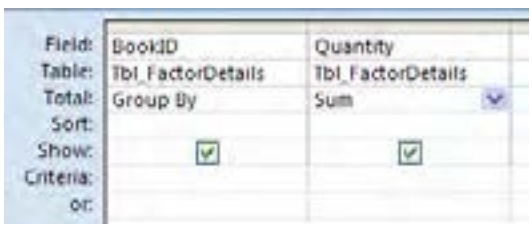


۴. در زبانه Design و قاب Show/Hide روی دکمه Totals کلیک کنید تا فعال شود. به این ترتیب ردیف جدید Total به شبکه بالا اضافه می‌شود.

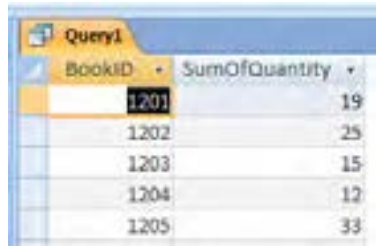
۵. در ستون فیلد BookID گزینه Group By انتخاب می‌شود. درج Group By از این جهت لازم است چون در نتیجه نهایی می‌خواهیم همه نتایج مربوط به هر BookID در یک سطر جمع (Aggriagte) شود.

۶. در ستون دوم و ردیف Filed، فیلد Quantity را انتخاب کنید چون قرار است تابع تجمعی روی این فیلد عمل کند.

۷. در ردیف Total ستون دوم با انتخاب تابع SUM نشان دهید که می‌خواهید عمل جمع انجام شود.



۸. روی دکمه Run کلیک کنید تا نتایجی شبیه به این تصویر به دست آید.


Query1

BookID	SumOfQuantity
1201	19
1202	25
1203	15
1204	12
1205	33

۹. اکسس به صورت خودکار نام SumOfQuantity را برای این ستون انتخاب کرده است.

۱۰. منوی View را باز کرده و کد SQL را ببنید.

```
SELECT BookID, Sum(Quantity) AS SumOfQuantity
FROM Tbl_FactorDetails
GROUP BY BookID;
```

۱۱. می‌خواهیم این کد را طوری تغییر دهیم تا تعداد کتب فروخته شـــده فقط برای کتاب‌هایی که شماره آن‌ها بین ۱۲۰۳ تا ۱۲۰۶ است به دست آید.

```
SELECT BookID, Sum(Quantity) AS SumOfQuantity
FROM Tbl_FactorDetails
GROUP BY BookID
HAVING BookID BETWEEN 1203 AND 1206
```


نکته

در پرس‌وجوهایی که حاوی عبارت GROUP BY هســـتند برای تعیین معیار جداسازی داده‌ها باید از عبارت HAVING استفاده نمایید.

مثال ۴ : مبلغ کل هر فاکتور را به دست آورید.

پیش از این یک پرس‌وجو با نام Qry_TotalPrice ساختیم که در آن مبلغ هر کدام از ردیف‌های فاکتور قید شده بود حالا با استفاده از این پرس‌وجو و تابع SUM می‌خواهیم به جواب نهایی برسیم.

```
SELECT FactorID AS [شماره فاکتور], SUM(TotalPrice) AS [مبلغ فاکتور]
FROM Qry_TotalPrice
GROUP BY FactorID
```

۸-۱-۳ استفاده از توابع تاریخی

برای انجام پردازش روی مقدار فیلدهایی که از نوع داده‌ای Date/Time هســـتند، توابعی در زبان SQL وجود دارد که در جدول زیر توضیح داده شده‌اند. دقت داشته باشید که برخی از این توابع صرفاً برای کار با تاریخ‌های میلادی مناسب هستند.

عبارت	عمل‌کرد	مثال
Date ( )	تاریخ جاری سیستم را به میلادی تولید می‌کند.	Factors.Date = Date() تاریخ صدور فاکتور، امروز باشد
Day (Date)	عدد مربوط به روز را در تاریخ برمی‌گرداند	Day (Factors.Date) <= 7 فاکتور در هفته اول ماه صادر شده باشد
Month (Date)	عدد متناظر با شماره ماه را برمی‌گرداند	Month (Factors.Date) = 2 فاکتور در ماه دوم سال صادر شده باشد.
Year (Date)	بخشِ سال را از تاریخ جدا می‌کند	Year (Factors.Date) = 1386 فاکتور مربوط به سال ۱۳۸۶ باشد
Weekday (Date)	روز هفته را مشخص می‌کند	Weekday (2010/02/24) عدد ۴ را به عنوان روز چهارم هفته برمی‌گرداند
Between Date1 AND Date2	در پرس‌وجوهای شرطی برای تعیین محدوده زمانی بین Date1 و Date2 به کار می‌رود.	Factors.Date BETWEEN #1387/01/01#AND#1387/12/29# شرطی برای جداسازی فاکتورهای سال ۸۷ است.

مثال ۱: پرس‌وجویی بنویسید که تاریخ جاری را به میلادی برگرداند و نشان دهد امروز، چندمین

روز هفته است.

SELECT Date() AS [تاریخ میلادی جاری],Weekday(Date()) AS [شماره روز]

پرس‌وجوهای ذخیره شـــده از نظر بازیابی اطلاعات مانند جداول هستند و می‌توان در سایر پرس‌وجوها از آن‌ها استفاده نمود.

۹-۱-۳ استفاده از توابع رشته‌ای

بخش عمده‌ای از اطلاعات درون پایگاه داده با انواع داده‌ای رشـــته‌ای ذخیره‌ســـازی می‌شوند؛ نام و نام خانوادگی افراد، نشـــانی‌ها، نام کالا و ... از جمله این داده‌ها محسوب می‌شوند. شما می‌توانید اعدادی را هم که قرار نیست پردازش ریاضی روی آن‌ها انجام شود به صورت رشته‌ای ذخیره کنید؛ برای مثال هیچ‌گاه لازم نیســـت روی «شـــماره ملی» افراد یک پردازش ریاضی مثل محاسبه مجموع یا میانگین انجام گیرد بنابراین ذخیره کردن آن به صورت رشته‌ای قابل قبول است.

برای کار روی داده‌های رشـــته‌ای توابع متعددی در SQL طراحی شـــده که موارد زیر کاربرد بیش‌تری دارند.

تابع	عمل‌کرد	مثال
Len(String)	طول رشته را محاسبه می‌کند.	Len(‘abcdef’) عدد ۶ را تولید می‌کند
Left(String,a)	از سمت چپ رشته String، تعداد a حرف را جدا می‌کند.	Left(‘abcdef’,2) رشته ab را برمی‌گرداند
Mid(String,s,a)	از حرف sام رشته String، تعداد a حرف را جدا می‌کند.	Mid(‘abcdef’,3,2) رشته cd را جدا می‌کند
Right(String,a)	از سمت راست رشته String، تعداد a حرف را برمی‌گرداند.	Right(‘abcdef’,2) رشته ef را جدا می‌کند

مثال ۱: نام مشتریانی را به دست آورید که طول نشانی آن‌ها بیش‌تر از ۳۰ حرف است.

```
SELECT Name FROM Tbl_Customers
WHERE LEN(Address)>30
```

مثال ۲: در یک گزارش به نام مشـــتریان و نشانی حدودی آن‌ها نیاز داریم. پرس‌وجویی بنویسید که ۱۵حرف اول نشـــانی را جدا نموده و با ذکر نام شـــهر در ابتدا و علامت ســـه نقطه در انتهای این عبارت نشان دهد.

```
SELECT Name AS [نام مشتری] , City + '-' + LEFT(Address,15) + '...' AS [نشانی حدودی]
FROM Tbl_Customers
```

Query1 | Tbl_FactorDetails | Tbl_Factors | Tb

نام مشتري	نشاني حدودي
فروشگاه کتاب آدینه	تهران-خیابان انقلاب-ن...
کتابسراي محمدي	نجف آباد-خیابان ولیعصر-ن...
انتشارات امیرکبیر	شیراز-خیابان کریمخان-...
کتابفروشي اندیشه	تهران-خیابان شهید مطه...
کتابفروشي امیني	آباده-خیابان امیرکبیر...
فروشگاه بزرگ کتاب	اصفهان-روبروي هتل عباس...

نکته

با استفاده از عمل‌گر + می‌توانید یک رشته را به رشته دیگر متصل کنید.

۲-۳ طراحی پرس‌وجوهای عملیاتی[1]

تا این‌جا با پرس‌وجوهایی آشـــنا شـــدید که وظیفه اصلی آن‌ها بازیابی اطلاعات است. این پرس‌وجوها با دســـتور SELECT شروع می‌شـــوند و با پیوند دادن جداول و استفاده از توابع، نتایج موردنظر کاربر را تولید می‌کنند. دســـته‌ای دیگر از پرس‌وجوها وجود دارند که برای دســـت‌کاری اطلاعات کاربرد دارند؛ می‌توانند جدول جدید بسازند یا داده‌ها را حذف، اضافه یا ویرایش نمایند.

1 . Action Queries

۳-۲-۱ پرس‌وجوی جدول ساز

گاهی اوقات برای به دســت آوردن مجموعه‌ای از رکوردها که اغلب حاوی فیلدهای محاسباتی هستند، نوشتن یک پرس‌وجو کافی نیست بلکه باید نتیجه پرس‌وجوهای ابتدایی را درون یک یا چند جدول ذخیره نموده و سپس با اجرای پرس‌وجوی نهایی روی این جداول به جواب مطلوب برسیم.

جداول جدیدی که با اســتفاده از پرس‌وجوهای جدول‌ســاز ایجاد می‌کنیم، دارای فیلدهایی هم‌نوع با جدول اولیه هستند و می‌توانند همه یا بخشی از رکوردهای جدول اصلی را در خود جای دهند.

برای انجام این کار به روش زیر عمل کنید:

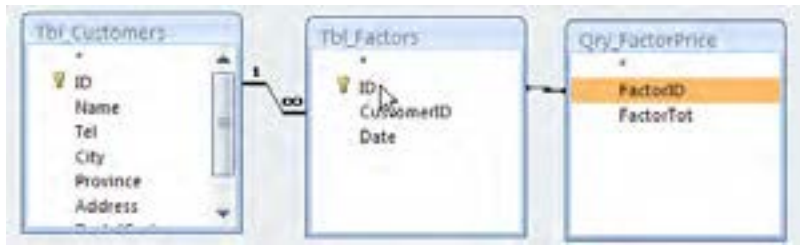
در این مثال قصد داریم یک پرس‌وجوی جدول‌ســاز حاوی شــماره فاکتور، نام مشتری، تاریخ فاکتور و مبلغ کل فاکتور ایجاد کنیم.

۱. با کلیک روی دکمه Query Design به نمای طراحی بروید.

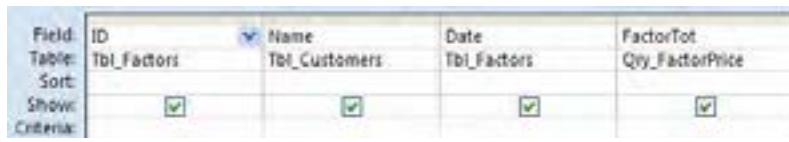
۲. از زبانه Tablesجداول Tbl_Customers، Tbl_Factors را وارد پنجره کنید.

۳. پیــش از این یک پرس‌وجو حاوی مبلغ فاکتورها ایجــاد و آن را با نام Qry_FactorPrice ایجاد کرده بودیم. در زبانه Queries این پرس‌وجو را هم به پنجره طراحی اضافه نمایید.

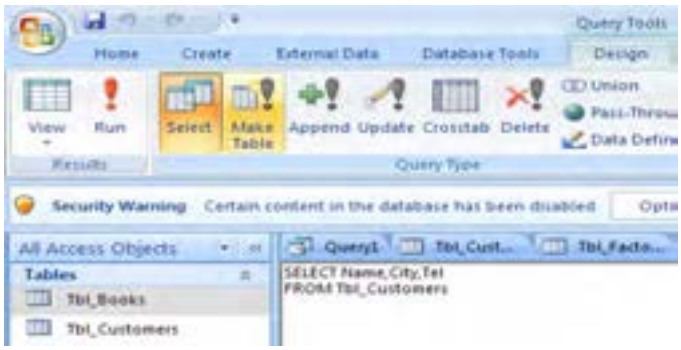
۴. با کلیک روی فیلد FactorID از پرس‌وجوی بهای کل فاکتور، آن را روی فیلد ID از جدول فاکتورهای بکشید تا یک پیوند بین این دو فیلد ایجاد شود.



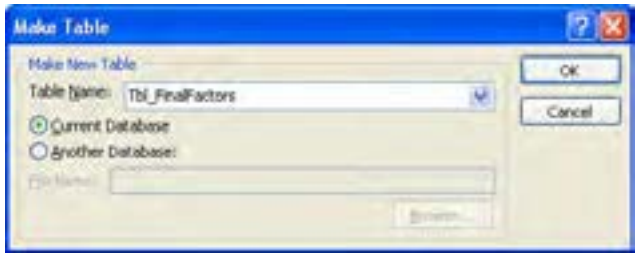
۵. فیلدهای شماره فاکتور، نام مشتری، تاریخ فاکتور و مبلغ کل فاکتور را انتخاب کنید تا به شبکه اضافه شوند.



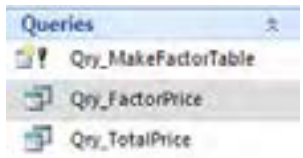
۶. در زبانه Design و قاب Query Type روی دکمه Make Table کلیک کنید.



۷. در پنجره‌ای که ظاهر می‌شود نامی برای جدول جدید وارد و روی دکمه OK کلیک کنید.



۸. روی دکمـــه Save کلیک نموده و نامی مثـــل Qry_MakeFactorTable را برای آن وارد کنید. به این ترتیب پرس‌وجو ذخیره می‌شود.



۹. اگر به لیســـت پرس‌وجوها نگاه کنید می‌بینید که یک علامت تعجب در کنار نام پرس‌وجو قرار گرفته و نشان می‌دهد که از نوع جدول‌ساز است.

۱۰. روی این پرس‌وجو دوبار کلیک کنید تا اجرا شود.

نکته

در Access 2007 بـــرای حفـــظ امنیت پایگاه داده، این نوع پرس‌وجوها بـــه صورت پیش‌فرض غیرفعال هســـتند. بـــرای فعال نمودن آن‌ها، در زیـــر زبانه‌های برنامه روی دکمـــه Options کلیک کنید و در پنجره ظاهر شـــده عبارت Enable this content را انتخاب نمایید. با کلیک کردن روی دکمه OK اجرای این نوع پرس‌وجوها امکان‌پذیر می‌شود.

۱۱. در پنجره تأیید اجرای پرس‌وجو، روی دکمه Yes کلیک کنید.

۱۲. جدول جدید ســـاخته شده و به لیســـت جداول اضافه می‌شود. توجه داشته باشید که اطلاعات جدول اولیه در این جدول کپی می‌شود و دیگر این دو جدول هیچ ارتباطی با هم ندارند؛ بنابراین تغییر داده‌ها در جدول اصلی روی داده‌های جدول ساخته شده تأثیری نخواهد گذاشت.

۱۳. پرس‌وجوی جدول‌ساز را در نمای SQL باز و کد تولیدی را بررسی کنید.

همان‌طور که می‌بینید برای تولید پرس‌وجوهای جدول‌ساز از این الگو استفاده می‌شود:

SELECT field_1, filed_2, …,field_n INTO table_name FORM table1

۲-۲-۳ پرس‌وجوی بروزرسانی[1] داده‌ها

برای تغییر داده‌های وارد شده درون یک جدول از الگوی زیر استفاده می‌شود.

```
UPDATE table_name
SET fileld_1=value_1, field_2=value_2 ,… , field_n=value_n
WHERE Criteria
```

مثال ۱: می‌خواهیم نام نویسنده کتاب شماره ۱۲۰۲ را به «حامد قنبری» تغییر دهیم.

```
UPDATE Tbl_Books
SET Author='حامد قنبری'
WHERE ID=1205
```

اگر عبارت WHERE ID=1205 را از پرس‌وجوی مثال قبل حذف کنیم چه اتفاقی می‌افتد؟

چون هیچ معیار خاصی برای انتخاب رکوردها ذکر نشده، بنابراین نام همه نویسنده‌ها تغییر خواهد کرد. دقت داشته باشید که پس از اجرای یک پرس‌وجو، امکان برگرداندن تغییرات (مثلاً با دستوری مانند Undo) وجود ندارد. بنابراین هنگام اجرای پرس‌وجوی تغییر داده‌ها باید دقت کافی به خرج دهید.

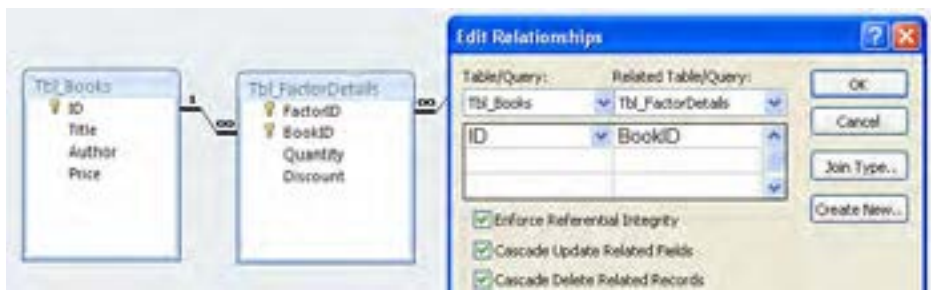
1 . Update

مثال ۲: پرس‌وجویی بنویسید که با اضافه کردن یک صفر به سمت راست شماره کتاب‌ها، همه شماره‌ها را ۵ رقمی کند. برای مثال شماره ۱۲۰۶ به ۱۲۰۶۰ تبدیل شود.

```
UPDATE Tbl_Books
SET ID=ID*10
```

وقتی شماره کتاب را در جدول Tbl_Books تغییر می‌دهیم، برای رکوردهای مرتبط با آن در جدول Tbl_FactorDetails چه اتفاقی می‌افتد؟

در زبانه Database Tools روی دکمه Relationships کلیک کنید تا ارتباطات میان جداول نشان داده شوند. روی ارتباط میان این دو جدول راست‌کلیک نموده و گزینه Edit Relationship را انتخاب کنید تا پنجره زیر ظاهر شود.



تیک خوردن گزینه Cascade Update Related Records به این معنی است که با تغییر شماره‌های کتاب در جدول Tbl_Books، شماره‌ی همه کتاب‌هایی که درون جدول Tbl_FatctorDetails ذخیره شده‌اند به صورت خودکار تغییر می‌کنند. به این نوع بروزرسانی، آبشاری[1] گفته می‌شود و در مورد دستور حذف هم صادق است.

مزیتی که اعمال این روش دارد این است که در جدول جزییات فاکتور، شماره کتابی ندارید که در جدول کتاب‌ها موجود نباشد. به این ترتیب، جامعیت پایگاه داده حفظ می‌شود و شما اصطلاحاً «داده سرگردان» نخواهید داشت.

1 . Cascade

مثال ۳ : می‌خواهیم شـــماره‌ها را به وضعیت قبل از مثال ۲ برگردانیم. پرس‌وجوی موردنیاز برای این کار را از طریق پنجره طراحی ایجاد کنید.

۱. در زبانه Create روی دکمه Query Wizard کلیک کنید تا پنجره طراحی ظاهر شود.

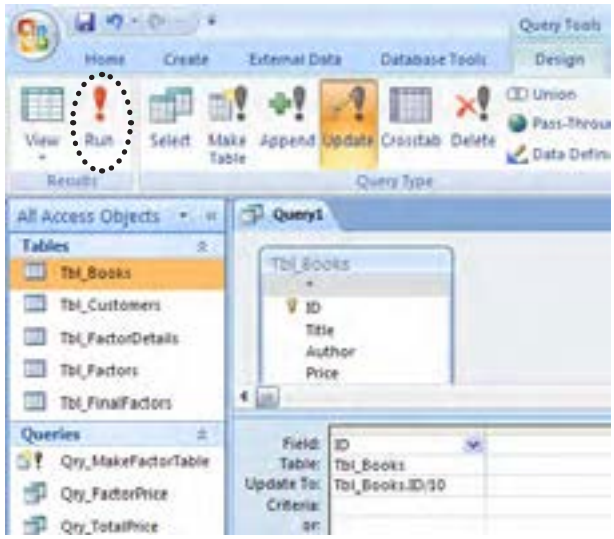
۲. جدول Tbl_Books را به پنجره اضافه کنید.

۳. در زبانه Design و قاب Query Type روی دکمه Update کلیک نمایید.

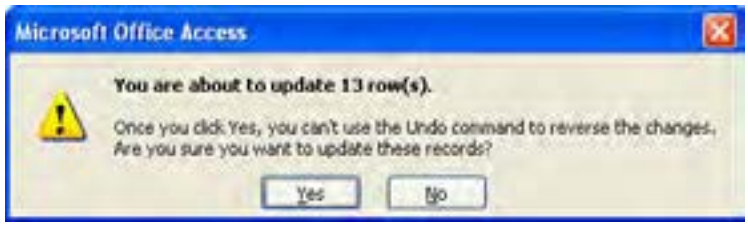
۴. در ستون اول و لیست Field، گزینه ID را انتخاب کنید.

۵. در ردیـــف Update to عبـــارت Tbl_Books.ID/10 را تایپ نمایید. این عبارت، شـــماره‌های کتاب را تقسیم بر ۱۰ کرده و صفر انتهای عدد را حذف می‌کند.

۶. روی دکمه Run کلیک کنید تا پرس‌وجو اجرا شود.



۷. پیغامی ظاهر شده و تعداد رکوردهایی را که تحت تأثیر این پرس‌وجو تغییر خواهند کرد نشان می‌دهد. روی دکمه Yes کلیک کنید تا پرس‌وجو اجرا شود.



۳-۲-۳ پرس‌وجوی حذف[1] داده‌ها

حذف رکوردهای وارد شده درون یک جدول به کمک الگوی زیر انجام می‌شود.

```
DELETE FROM table_name
WHERE Criteria
```

مثال ۱: فرض کنید از پایگاه داده خود یک نسخه پشتیبان تهیه کرده‌ایم و می‌خواهیم برای کاهش حجم پایگاه داده و افزایش سرعت بازیابی اطلاعات، فاکتورهای صادر شده قبل از سال ۱۳۸۹ را حذف کنیم. این پرس‌وجو را بنویسید.

```
DELETE FROM Tbl_Factors
WHERE Year(Date)<1389
```

۳-۲-۴ پرس‌وجوی درج[2] داده‌ها

با استفاده از الگوی زیر می‌توانید رکوردهای جدیدی را درون جداول پایگاه داده درج کنید.

```
INSERT INTO Tbl_name (field_1, field_2 , … , filed_n)
VALUES (value_1,value_2, … , value_n)
```

اگر فیلدهای رکورد به تعداد و متناظر با فیلدهای جدول باشند نیازی به ذکر نام فیلدها نیست.

مثال ۱: کتاب جدیدی را با مشخصات زیر وارد جدول کتاب‌ها کنید.

شماره : ۱۲۲۰، نام کتاب : Excel 2007، نویسنده : علی اکبری، قیمت : ۵۶۰۰۰

```
INSERT INTO Tbl_Books VALUES(۱۲۲۰,'Excel 2007' , 'علی اکبری' , ۵۶۰۰۰)
```

---

1 . Delete

2 . Insert

مثال ۲ : رکورد مشتری با مشخصات زیر را وارد جدول مشتری‌ها کنید.

شماره مشتری : ۱۰، نام : کتابسرای افق، شهر : فارسان

در این‌جا فقط مقدار ۳ فیلد رکورد مشـــخص است بنابراین باید نام فیلدها را در پرس‌وجو ذکر کنیم تا هر مقدار در فیلد متناظر درج شود.

```
INSERT INTO Tbl_Customers(ID,Name,City)
VALUES(۱۰,'کتابسرای افق','فارسان')
```

در دستور درج، فقط مقادیری می‌توانند فاقد مقدار باشند که هنگام تعریف جدول، مشخصه Required آن‌ها روی No تنظیم شـــده باشـــند و به بیان دیگر مقدار NULL را بپذیرند. NULL به معنی عدم وجود مقدار در فیلد است.

آیا اجرای پرس‌وجوی زیر موفقیت‌آمیز است؟

```
INSERT INTO Tbl_Customers(Name,City)
VALUES('دانشگاه فردوسی','مشهد')
```

خیر – در این رکورد، شـــماره مشـــتری قید نشده و از آن جا که شماره مشـــتری در جدول مشتریان کلید اصلی محســـوب می شـــود و از طرفی، کلید اصلی جدول، مقادیر NULL را نمی پذیرد، لذا درج این رکورد امکان‌پذیر نیست.

۳-۲-۵ پرس‌وجوهای CrossTab

ایجاد پرس‌وجوهای CrossTab روشـــی بسیار سریع و کارآمد برای خلاصه‌کردن داده‌های چند جدول و ایجاد یک گزارش نهایی در قالب چند ســـطر و ســـتون است؛ گزارشی که تولید آن با پرس‌وجوهای معمولی اگر غیرممکن نباشد حتماً دشوار خواهد بود.

در این پرس‌وجو، داده‌های یک فیلد را به عنوان ســـتون‌های خروجی و داده‌های فیلد دیگر را به عنوان ردیف‌های آن منظور می‌کنیم. در محل تقاطع هر ســـطر و ســـتون هم داده محاســـبه شـــده توسط توابع محاسباتی (مثل مجموع، میانگین و ...) را قرار می‌دهیم.

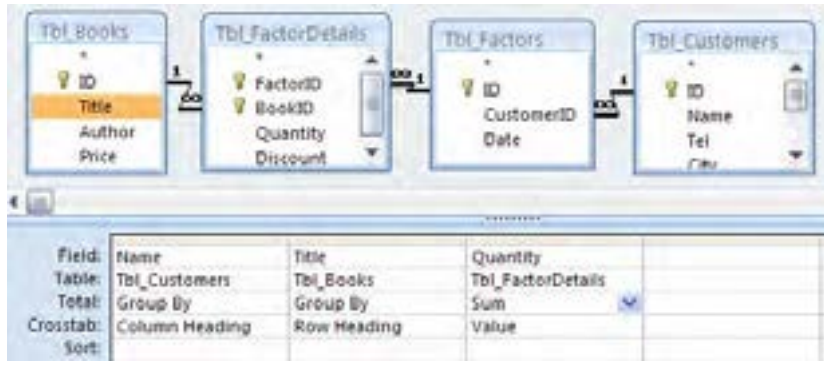
مثال ۱: می‌خواهیم بدانیم از هر کتاب توسط چه مشتری‌هایی و مجموعاً چند نسخه خرید صورت گرفته است.

۱. وارد پنجره طراحی پرس‌وجو شوید.

۲. چهار جدول تعریف شده برای پایگاه داده Publisher را به پنجره اضافه کنید.

۳. در زبانه Design و قاب Query Type روی دکمه CrossTab کلیک نمایید.

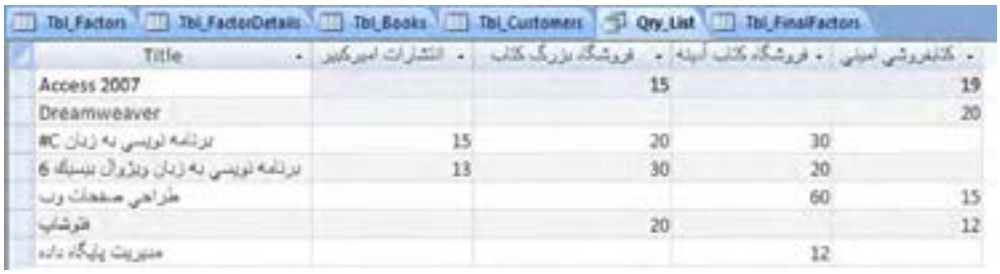
۴. در ستون اول، فیلد نام مشتری و در ردیف Crosstab حالت Column Heading را انتخاب کنید تا نام مشتری‌ها روی سرستون‌های خروجی قرار گیرند.



۵. در ســـتون دوم، فیلد نام کتاب و در ردیف Crosstab حالت Row Heading را انتخاب نمایید. به این ترتیب نام کتاب‌ها در ردیف‌های خروجی قرار می‌گیرند.

۶. می‌خواهیم مجموع خرید هر کتاب توســـط هر مشـــتری را تعیین کنیم. لذا بایـــد فیلد Quantity را انتخاب، ردیف Total را روی Sum و ردیف Crosstab را روی Value تنظیم نمایید.

۷. پرس‌وجوی ساخته شده را اجرا کنید تا نتیجه‌ای شبیه به تصویر زیر ظاهر شود.

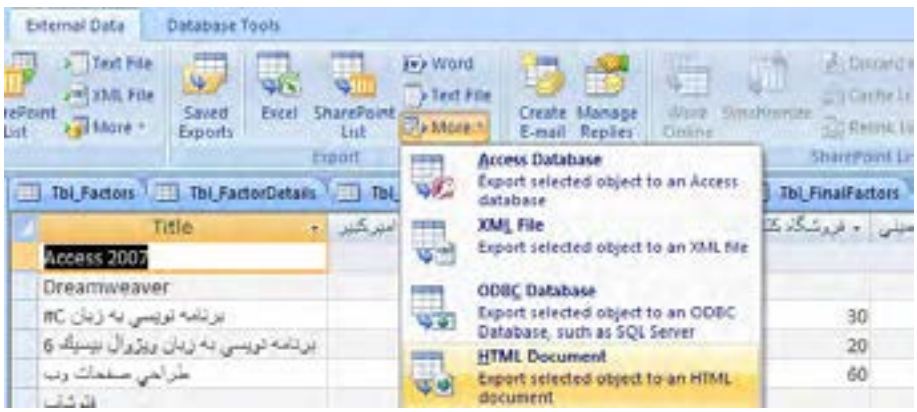


۳-۳ Export پرس‌وجوها به فایل HTML

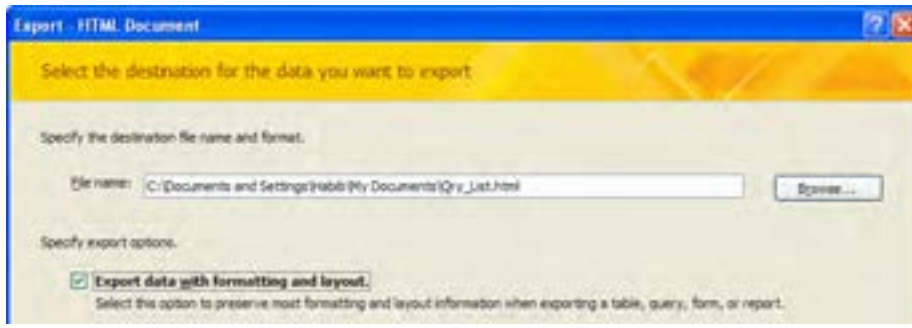
در فصل دوم با روش‌های متنوع صدور جداول آشنا شدید. در این بخش یاد می‌گیرید که چگونه می‌توان نتیجه یک پرس‌وجو را تبدیل به فایل HTML نمود.

۱. یک پرس‌وجوی CrossTab تولید و آمار فروش هر کتاب را به تفکیک مشتریان استخراج کنید.

۲. در زبانـــه External Data و قاب Export، منوی More را باز و روی گزینه HTML Document کلیک کنید.



۳. در پنجره‌ای که ظاهر می‌شـــود، نام و محل ذخیره‌ســـازی فایل HTML را تعیین نموده و ضمناً تیک گزینه اول را بزنید تا قالب‌بندی داده‌ها روی فایل نهایی منعکس شود. سپس روی دکمه OK کلیک کنید.



۴. با ظاهر شدن پنجره کدگذاری، حالت UTF-8 را انتخاب و روی دکمه OK کلیک کنید.

۵. می‌توانید نتیجه کار را در مرورگر اینترنت ببینید.

Title	انتشارات امیرکبیر	فروشگاه بزرگ کتاب	فروشگاه کتاب آدینه	کتابفروشی امینی
Access 2007		15		19
Dreamweaver				20
برنامه نویسی به زبان C#	15	20	30	
برنامه نویسی به زبان ویژوال دیسوگ ۶	13	30	20	
طراحی صفحات وب			60	15
فتوشاپ		20		12
مدیریت پایگاه داده			12	

Understand ways to sum data

You can sum a column of numbers in a query by using a type of function called an aggregate function. Aggregate functions perform a calculation on a column of data and return a single value. Access provides a variety of aggregate functions, including Sum, Count, Avg (for computing averages), Min and Max. You sum data by adding the Sum function to your query, you count data by using the Count function, and so on.

In addition, Office Access 2007 provides several ways to add Sum and other aggregate functions to a query. You can:

- Open your query in Datasheet view and add a Total row. The Total Row, a new feature in Office Access 2007, allows you to use an aggregate function in one or more columns of a query result set without having to change the design of your query.

- Create a totals query. A totals query calculates subtotals across groups of records; a Total row calculates grand totals for one or more columns (fields) of data. For example, if you

want to subtotal all sales by city or by quarter, you use a totals query to group your records by the desired category and you then sum the sales figures.

Create a crosstab query. A crosstab query is a special type of query that displays its results in a grid that resembles a Microsoft Office Excel 2007 worksheet. Crosstab queries summarize your values and then group them by two sets of facts — one set down the side (row headings), and the other across the top (column headings). For example, you can use a crosstab query to display sales totals for each city for the past three years, as the following table shows:

City	2003	2004	2005
Paris	254,556	372,455	467,892
Sydney	478,021	372,987	276,399
Jakarta	572,997	684,374	792,571
...	...	...	...

۱. متن بالا را مطالعه کرده و در مورد آن توضیح دهید.
۲. روش‌های موجود برای اضافه کردن توابع تجمعی را به پرس‌وجو توضیح دهید.
۳. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. پرس‌وجوهایی در نمای طراحی ایجاد کنید که نتایج زیر را برگردانند :

الف) مشخصات ناشرانی که ساکن «استان فارس» هستند اما در شهر «شیراز» سکونت ندارند.

ب) مشخصات کتاب‌هایی که در مورد «فتوشاپ» هستند و قیمت آن‌ها کم‌تر از ۵۰۰۰۰ ریال است.

ج) عنوان کتاب‌هایی که قیمت آن‌ها پس از بیســـت درصد تخفیـــف، هم‌چنان بیش‌تر از ۶۰۰۰۰ ریال است.

د) مجموع فروش کتاب شماره ۱۲۰۲

۲. پرس‌وجوهایی بنویسید که پس از اجرا، نتایج زیر را نشان دهند :

الف) مجموع فروش کتاب‌هایی که شماره آن‌ها کم‌تر از ۱۲۱۰ است.

ب) تعداد فاکتورهایی که در سال ۱۳۸۸ صادر شده‌اند.

ج) میانگین قیمت کتاب‌هایی که که توسط یک نویسنده خاص (مثلاً محمد محمدی) نوشته شده‌اند.

د) نام گران‌ترین و ارزان‌ترین کتاب موجود در پایگاه داده.

فصل چهارم

فصل چهارم : طراحی یک فرم پیشرفته

فرم، واسطه‌ای میان کاربر و پایگاه داده است و با استفاده از آن می‌توان داده‌ها را به سادگی وارد جداول پایـــگاه داده کرد. به‌خصوص وقتی بخواهید داده‌ها را درون چند جـــدول مرتبط با هم وارد کنید، کارایی و سادگی استفاده از فرم‌ها را بیش‌تر درک خواهید کرد.

در این فصل ضمن بیان نکاتی در خصوص ایجاد فرم و ویرایش اجزاء آن، با نحوه تولید فرم‌های پیشرفته که اطلاعات چند جدول مرتبط با هم را نشان دهند آشنا خواهید شد.

در Access 2007 روش‌هـــای متعـــددی برای ایجاد یک فرم پیش‌بینی شـــده که بســـیاری از آن‌ها به صورت خودکار در قالب یک ویزارد[1] عمل می‌کنند. این ویزاردها با توجه به جداول و فیلدهای تعیین شده، کنترل‌های موردنیاز را به پنجره طراحی اضافه می‌کنند. در ابتدای این فصل قصد داریم روش ایجاد یک فرم و به‌ویژه فرم‌های پیشـــرفته را بدون استفاده از ویزارد آموزش دهیم تا مهارت شما در ایجاد فرم‌های دلخواه و تغییر ظاهر و کارکرد آن‌ها به نحو قابل ملاحظه‌ای افزایش یابد. اولین گام در این مســـیر، آشنایی با جعبه ابزار[2] است.

---

1 . Wizard

2 . ToolBox

۱-۴ کار با جعبه ابزار

در زبانه Create و قاب Forms روی دکمه Form Design کلیک کنید تا یک فرم خام در اختیار شـــما قرار گیرد. فرم خام صفحه‌ای است که می‌توانید کنترل‌های موردنظر را بر روی آن قرار دهید تا برای نمایش اطلاعات یک جدول یا پرس‌وجو و نیز ایجاد قابلیت ویرایش آن‌ها به شکل موردنظر شما دربیاید.

به این ترتیب دو زبانه جدید به نام‌های Design و Arrange به نوار اصلی برنامه اضافه می‌شود.

در زبانه Design قابی به نام Controls وجود دارد که همه کنترل‌های موردنیاز را برای ســـاخت فرم در خود جای داده اســـت. به این قاب، «جعبه ابزار» گفته می‌شود و در نسخه‌های قبلی Access به صورت یک پنجره مجزا و قابل جابه‌جایی در دسترس بود.

کنترل‌ها، اشیایی گرافیکی هستند که برای نمایش داده‌ها و توضیحات یا اجرای دستورات، بر روی فرم قرار می‌گیرند. برای مثال یک کادر متنی (TextBox) برای نمایش داده‌های متنی و عددی کاربرد دارد و از کنترل دکمه (Button) هم برای اجرای دستور موردنظر استفاده می‌شود.

در جدول زیر نام و آیکن همه کنترل‌ها و اجزاء جعبه ابزار و کارکرد آن‌ها توضیح داده شده است.

نام کنترل	نام فارسی	آیکن	کارکرد
Text Box	کادر متنی		برای نمایش یک رشته یا ورود و ویرایش آن توسط کاربر به کار می‌رود.
Label	برچسب		برای نمایش توضیحات و متن‌های غیرقابل تغییر کاربرد دارد.
Button	دکمه		با کلیک کردن روی دکمه، دستورات موردنیاز اجرا می‌شود.
Combo Box	لیست بازشونده		یک لیست بازشونده است که کاربر می‌تواند یکی از گزینه‌های موجود در آن را انتخاب کند.
List Box	لیست		لیستی از عناصر قابل انتخاب را نشان می‌دهد.
Subform	زیرفرم		برای ایجاد فرم‌هایی که حاوی یک یا چند فرم دیگر هستند کاربرد دارد.
Line	خط		برای رسم خط صاف روی فرم استفاده می‌شود.
Rectangle	مستطیل		برای رسم مربع و مستطیل روی فرم کاربرد دارد.
Bound Object Frame	قاب اشیاء متصل		قابی است که یک شیء OLE را در خود جای می‌دهد و این شیء در یکی از فیلدهای جدول ذخیره می‌شود.
Option Group	گروه انتخاب		تعدادی کنترل مرتبط با هم مثل Option Buttonرا در خود جای می‌دهد.
Check Box	کادر تأیید		کادری دو حالته است که با کلیک روی آن انتخاب می‌شود و با کلیک بعدی از انتخاب خارج می‌گردد.
Option Button	دکمه انتخاب		این کنترل به دکمه رادیویی هم معروف است و تعدادی گزینه را نشان می‌دهد که فقط یکی از آن‌ها قابل انتخاب است.

نام کنترل	نام فارسی	آیکن	کارکرد
Toggle Button	دکمه دو حالته		دکمه‌ای است که می‌تواند در دو وضعیت بالا یا پایین باشد.
Tab Control	کنترل چند برگه‌ای		برای طراحی فرم‌هایی که دارای چند سربرگ هستند استفاده می‌شود.
Insert Chart	درج نمودار		یک نمودار را وارد فرم می‌کند.
Unbound Object Frame	قاب اشیاء غیرمتصل		قابی است که یک شیء OLE را در خود جای می‌دهد و این شیء خارج از پایگاه داده ذخیره شده است.
Image	تصویر		برای افزودن تصویر به فرم کاربرد دارد.
Insert Page Break	درج صفحه جدید		برای ایجاد یک صفحه جدید استفاده می‌شود.
Insert Hyperlink	درج پیوند		روی فرم یک پیوند به عناصر داخل یا خارج پایگاه داده ایجاد می‌کند.
Insert Attach-ment	درج ضمیمه		برای افزودن فایل ضمیمه به فرم کاربرد دارد.
Select	انتخاب		وقتی این ابزار فعال باشد می‌توان کنترل‌های روی فرم را انتخاب کرد.
Use Control Wizards	ویزارد کنترل‌ها		وقتی این ابزار فعال باشد، هنگام ایجاد برخی از کنترل‌ها، ویزاردی برای تنظیم خصوصیت‌های آن کنترل راه‌اندازی می‌شود.
Title	عنوان		عنوانی را به بالای فرم اضافه می‌کند.
Insert Page Number	شماره صفحه		صفحات یک فرم را شماره‌گذاری می‌کند.
Date & Time	زمان و تاریخ		زمان و تاریخ جاری را در بالای فرم درج می‌نماید.
Logo	لوگو		یک تصویر کوچک را به عنوان لوگو به فرم اضافه می‌کند.

۴-۲ درج و ویرایش اجزاء فرم و کنترل‌ها

کنترل‌ها را بر اساس نوع کارکردی که دارند می‌توان به سه دسته زیر تقسیم‌بندی کرد:

الف) کنترل‌های متصل[1]: این دسته از کنترل‌ها به یک فیلد از یکی از جداول پایگاه داده متصل هستند و با تغییر مقدار درون فرم، تغییرات بلافاصله روی جدول منعکس می‌شود. (مانند TextBox)

ب)کنترل‌های غیرمتصل[2]: این نوع کنترل‌ها با داده‌های درون جداول ارتباطی ندارند. (مانند Line)

ج) کنترل‌های محاسباتی[3]: این دسته از کنترل‌ها به واسطه یک عبارت محاسباتی با تعدادی از فیلدهای جداول در ارتباط هستند. (مانند یک Label که جمع کتب خریداری شده را نشان می‌دهد)

۴-۲-۱ درج کنترل

فرم‌های اکسس حاوی مجموعه‌ای از کنترل‌ها هستند که در میان کنترل‌های مثل TextBox، Label، Button و ComboBox کاربرد بیش‌تری دارند. برای اضافه کردن یک دکمه به فرم باید به روش زیر عمل کنید:

۱. با کلیک روی دکمه Form Design، یک فرم خام ایجاد کنید.

۲. در جعبه ابزار، دکمه Use Control Wizards را غیرفعال کنید تا بعد از رسم دکمه، ویزارد تنظیم خصوصیات آن ظاهر نشود.

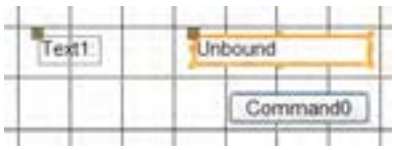
۳. روی کنترل دکمه کلیک کنید.

۴. اشاره‌گر را روی صفحه طراحی فرم ببرید. سپس کلیک کرده و با حرکت دادن اشاره‌گر، اندازه تقریبی دکمه را تعیین نمایید.

1 . Bound Control
2 . Unbound Controls
3 . Calculated Controls

۵. دکمه ماوس را رها کنید تا کنترل روی صفحه قرار گیرد.

۶. این‌بار روی کنترل TextBox کلیک کرده و با اشـــاره‌گر یک مستطیل روی فرم رسم کنید تا یک کادر متنی به فرم اضافه شود. همان‌طور که می‌بینید یک برچسب هم به صورت خودکار در کنار کادر متنی قرار می‌گیرد.

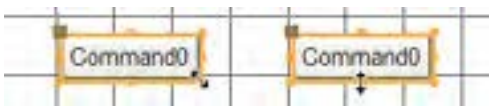


۲-۲-۴ تغییر اندازه و مکان کنترل

۱. روی دکمه کلیک کنید تا انتخاب شود.

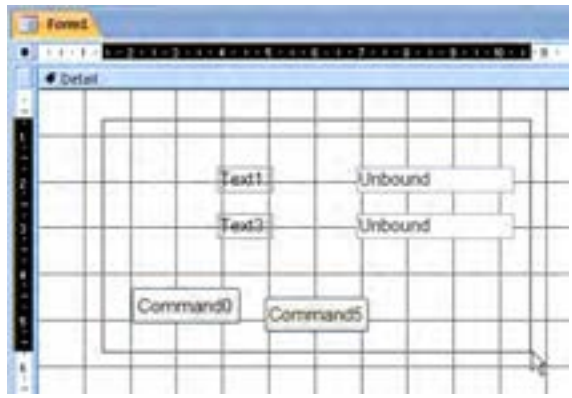
۲. هشت دستگیره مربع‌شکل در اطراف آن ظاهر می‌شود. روی دستگیره بزرگ‌تر کلیک کنید و دکمه را حرکت دهید.

۳. برای تغییر ابعاد دکمه، روی یکی از ۷ دستگیره‌ی نارنجی‌رنگ کلیک نموده و اشاره‌گر را حرکت دهید. در این حالت اشاره‌گر به شکل یک پیکان دوسر درمی‌آید.



یـــک دکمـــه و یک کادر متنی دیگر به فرم اضافه کنید. می‌خواهیـــم همه کنترل‌های روی فرم را با هم جابه‌جا کنیم.

۱. در حالی‌که دکمه انتخاب فعال است، در گوشه بالا و سمت راست فرم کلیک کرده و ضمن نگه‌داشتن کلید ماوس، اشاره‌گر را به صورت قطری حرکت دهید تا همه کادرهای متنی به همراه برچسب‌ها درون کادر مستطیلی قرار گیرند و انتخاب شوند.



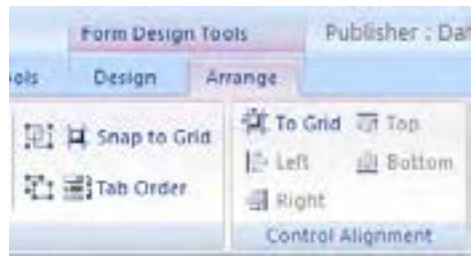
۲. اشاره‌گر را روی یکی از کنترل‌ها ببرید تا به شکل  در بیاید. در این حالت با کلیک کردن و حرکت دادن ماوس، همه کنترل‌ها با هم جابه‌جا می‌شوند.

۳-۲-۴ چینش کنترل‌ها

چیدن منظم کنترل‌ها درون یک فرم و رعایت ترازبندی نقش زیادی در زیبایی فرم و سادگی کار با آن دارد. وقتی یک فرم خام را ایجاد می‌کنید، در حالت پیش‌فرض بر روی آن خطوطی شبکه‌ای وجود دارد که Grid نامیده می‌شود.

۱. روی یکی از کنترل‌ها (مثلاً دکمه) کلیک کنید تا انتخاب شود.

۲. در زبانه Arrange دکمه Snap to Grid را غیرفعال کنید.

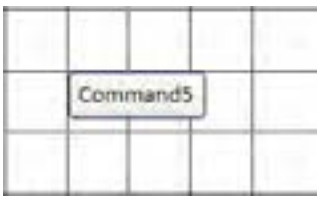


۳. با استفاده از دکمه‌های پیکانی صفحه کلید[1]، کنترل را حرکت دهید. همان‌طور که می‌بینید جابه‌جایی کنترل بسیار کم و دقیق است.

1 . Arrow Keys

۴. این بار دکمه Snap to Grid را فعال نمایید.

۵. به روش قبل، کنترل را حرکت دهید. کنترل، با پرش‌های کوتاه جابه‌جا می‌شود تا لبه‌های آن با خطوط شبکه‌ای مماس شود و به آن‌ها بچسبد.



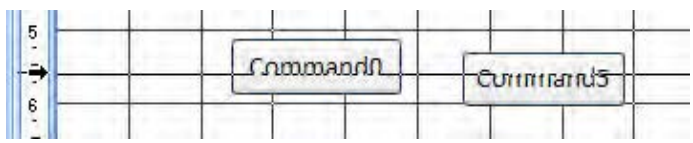
فعال کردن دکمه Snap to Grid روشـــی ســـریع برای قرار دادن کنترل ها روی فرم است تا اجزاء فرم نســـبت به خطوط شبکه ای و سایر کنترل ها ترازبندی مناسبی داشته باشند. توجه داشته باشید که هنگام حرکت دادن کنترل با استفاده از ماوس هم خاصیت Snap to Grid عمل می‌کند.

۴-۲-۴ ترازبندی کنترل‌ها

وقتی کنترلی را روی صفحه قرار می‌دهید باید لبه‌های آن را طوری تنظیم کنید تا با لبه سایر کنترل‌های مجاور در یک راســـتای عمودی یا افقی باشد. به این کار ترازبندی[1] گفته می‌شود. می‌خواهیم دو دکمه‌ای را که روی فرم قرار داده‌ایم از لبه بالا تراز کنیم.

۱. اشاره‌گر را روی خط‌کش سمت چپ فرم ببرید تا به یک پیکان سیاه‌رنگ تبدیل شود.

۲. در نقطه‌ای از خط‌کش که هم‌راســـتا با دو دکمه باشـــد کلیک کنید. این کار روشی سریع برای انتخاب کنترل‌هایی است که در یک راستای عمودی یا افقی قرار دارند.

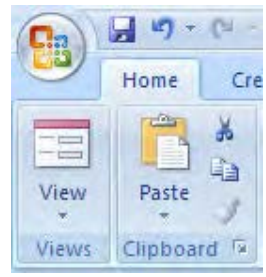


۳. در زبانـــه Arrange و قـــاب Control Alignment روی دکمه Top کلیـــک کنید تا لبه بالایی دکمه‌ها نسبت به هم تراز شود.

1 . Alignment

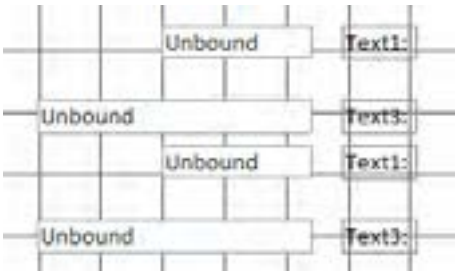
در فرم‌هایی که بیش‌ترِ داده‌ها به صورت فارسی وارد می‌شوند، باید برچسب‌ها را به سمت راست کادرهای متنی انتقال دهید.

می‌خواهیم فرمی برای ویرایش اطلاعات جدول کتاب‌ها بسازیم؛ بنابراین دو کادر متنی دیگر به فرم اضافه کنید. یک روش سریع برای انجام این کار، انتخاب کادرهای متنی و برچسب‌ها، کپی و سپس Paste کردن آن‌ها با استفاده از قاب Clipboard در زبانه Home است.



۴. برای انتخاب چند کنترل، کافی است کلید Shift را نگه‌داشته و روی کنترل موردنظر کلیک کنید. با نگه داشتن کلید Shift و کلیک مجدد روی کنترل می‌توانید آن را از مجموعه انتخاب شده خارج نمایید.

۵. برچسب‌ها را از سمت چپ و کادرهای متنی را از سمت راست تراز کنید.



می‌خواهیم فاصله میان کنترل‌ها را تنظیم کنیم.

۶. کادرهای متنی و برچسب‌ها را انتخاب کنید.

۷. در زبانه Arrange و قاب Position روی دکمه Make Vertical Spacing Equal کلیک کنید تا فاصله کادرهای متنی از هم، یک اندازه شود.

۸. با کلیک کردن روی دکمه‌های Increase/Decrease Verical Spaceing می‌توانید این فاصله را کاهش/ افزایش دهید.

۵-۲-۴ بهبود ظاهر فرم

۱. در جعبه ابزار روی دکمه Tilte کلیک کنید تا یک ســـربرگ به فرم اضافه شود و به صورت پیش‌فرض، نام فرم درون یک کادر قرار گیرد.

۲. عبارت دلخواه را درون این کادر وارد کنید.

۳. کادر را به وسط سربرگ منتقل نمایید.

۴. با حرکت دادن خط جداکننده فرم از سربرگ، اندازه سربرگ را تعیین کنید.

۵. در جعبه ابزار، ابزار مستطیل را انتخاب نموده و دور کنترل‌ها یک کادر مستطیلی رسم کنید.

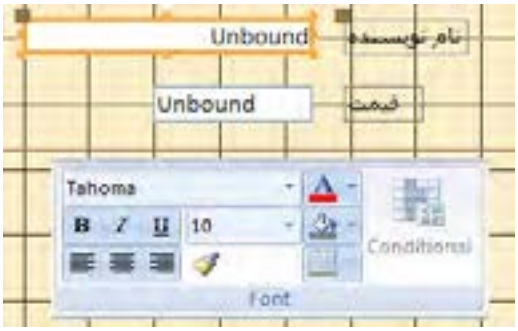
۶. روی مستطیل رسم شده کلیک و در جعبه ابزار، رنگ و ضخامت آن را تعیین نمایید.

۷. در زبانه Arrange به سراغ قاب AutoFormat بروید و منوی آن را باز کنید.

۸. روی یکی از قالب‌بندی‌های پیش‌ساخته کلیک کنید تا روی فرم اعمال شود.

۹. درون برچسب‌ها دوبار کلیک کنید تا محتوای آن‌ها انتخاب شود. حالا عبارت دلخواه را وارد نموده در پایان کلید Enter را بزنید تا ثبت شود.

۱۰. کنترل‌های موردنظر را انتخاب و در قاب Font از زبانه Home، تغییرات موردنظر را روی نوع قلم، اندازه قلم و نوع چینش متن اعمال کنید. برای مثال کادر متنی «نام نویسنده» باید راست‌چین باشد اما فیلدهای عددی می‌توانند چپ‌چین باشند.



۶-۲-۴ ذخیره‌سازی فرم

۱. منوی نمای فرم را باز کرده و روی حالت Form View کلیک کنید.

۲. برای ایجاد تغییر روی فرم باید به نمای طراحی (Design View) برگردید.
۳. اگر از نتیجه کار رضایت دارید، روی دکمه Save کلیک و فرم را با نام Frm_Books_1 ذخیره کنید.

۴-۳ تنظیم خصوصیت اجزاء فرم و کنترل‌ها

هر فرم و کنترل‌هایی که به آن اضافه کرده‌ایم دارای مجموعه‌ای از خصوصیت‌ها[1] هستند که ظاهر و نوع عمل‌کرد آن عنصر را مشخص می‌کنند. با تغییر این خاصیت‌ها می‌توانید ظاهر و عمل‌کرد فرم و کنترل‌های موجود روی آن را به شکل دلخواه تنظیم نمایید.

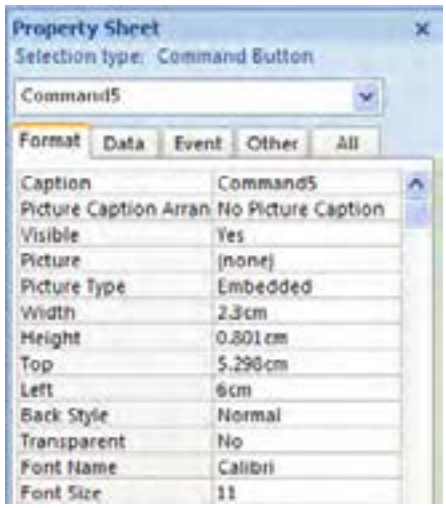
۴-۳-۱ کار با برگه خصوصیات

۱. فرم Frm_Books_1 را که در بخش قبل ایجاد و ذخیره کردید، در نمای طراحی باز نمایید.
۲. در زبانه Desing و قاب Tools روی دکمه Property Sheet کلیک کنید تا برگه تنظیم خصوصیات، در سمت راست پنجره برنامه ظاهر شود. کلیک روی این دکمه، برگه را پنهان خواهد کرد.

1 . Properties

۳. در بالای این برگه، یک لیســـت بازشونده وجود دارد که می‌توانید فرم و اجزاء و کنترل‌های موجود در آن را انتخاب کنید. این کار با کلیک کردن روی عنصر موردنظر هم قابل انجام اســـت. روی یکی از دکمه‌ها کلیک کنید.

۴. برگه خصوصیات این کنترل ظاهر می‌شود. تعداد و نوع این خصوصیات با خصوصیات یک کنترل دیگر (مثلاً برچسب) تفاوت قابل توجهی دارد.



برگه خصوصیات پنج زبانه دارد که هر یک حاوی موارد زیر است:

الف) زبانه Format : خصوصیات مربوط به قالب‌بندی کنترل مانند ابعاد، نوع و اندازه قلم، رنگ و ... در این زبانه قابل تنظیم اســـت. در بخش قبل این کار را از روش معمول نرم‌افزارهای Office یعنی قاب Font انجام دادیم.

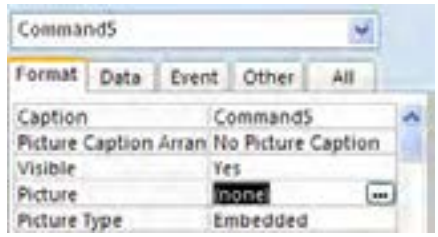
ب) زبانه Data : خصوصیاتی مانند فیلد متصل به کنترل، مقادیر پیش‌فرض، قواعد محدودســـازی ورود اطلاعات و ... در این زبانه تنظیم می‌شود.

ج) زبانه Event : همه رویدادهای کنترل و فرم از طریق فیلدهای این زبانه تنظیم می‌شوند. برای مثال تعیین می‌کنید که با وقوع رویداد کلیک روی کنترل (On Click) چه عملیات یا کدی اجرا شود.

د) زبانه Other : حاوی تنظیماتی مانند نام کنترل و ... است.

ه) زبانه All : همه خصوصیات یک عنصر که در چهار زبانه Format، Data، Event و Other قرار دارند در این زبانه به صورت یک‌جا قابل تنظیم هستند.

۵. در زبانه Format روی عبارت Picture کلیک کنید.



۶. در ستون سمت راست، روی دکمه تنظیم تصویر کلیک نمایید تا پنجره انتخاب تصویر ظاهر شود.

۷. یکی از تصاویر موجود را انتخاب و روی دکمه OK کلیک کنید تا تصویر روی دکمه قرار گیرد.

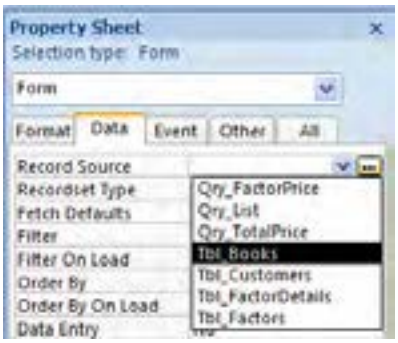
۲-۳-۴ متصل کردن داده‌ها

هدف مـــا از طراحی فرم Frm_Books_1 نمایش و ویرایش اطلاعات جـــدول کتاب‌ها بود بنابراین باید کنترل‌های موجود در این فرم را به فیلدهای جدول Tbl_Books متصل[1] کنیم.

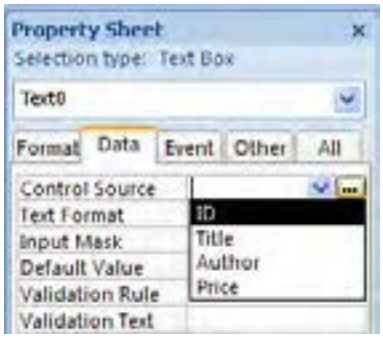
۱. در بالای برگه خصوصیات، لیست بازشونده تعیین عنصر را باز و گزینه Form را انتخاب کنید. این کار با کلیک روی بخش آبی‌رنگِ خارج از بخش شبکه‌ای هم قابل انجام است.

۲. در زبانـــه Data، خصوصیت Record Source را روی Tbl_Books تنظیم کنید. با این کار به فرم اعلام می‌کنید که باید داده‌ها را از جدول کتاب‌ها بخواند. همه پرس‌وجوها و جداول طراحی شـــده در پایگاه داده جاری از طریق این منو قابل انتخاب هستند.

1 . Bind or Bound



۳. حـــالا به ســـراغ اولین کادر متنی برویـــد و در زبانه Data، خصوصیـــت Control Source را روی فیلد موردنظر تنظیم نمایید. در این منو، همه فیلدهای جدول یا پرس‌وجویی که در قسمت قبل انتخاب کرده‌اید قابل انتخاب هستند.

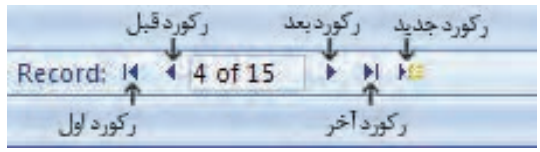


۴. همین کار را برای سه کادر متنی دیگر هم انجام دهید. عبارت Unbound (غیرمتصل) درون این کادرها با نام فیلد جایگزین می‌شود چون کنترل به یک فیلد متصل (Bound) شده است.

۵. فرم را به نمای Form View ببرید.

۶. اولین رکورد جدول نمایش داده می‌شود. با کلیک روی دکمه‌های پیمایش[1] که در پایین فرم قرار دارند می‌توانیـــد رکوردها را ببینید و فیلدهای آن‌هـــا را تغییر دهید. همچنین با کلیک روی دکمه New record، یک رکورد خالی ایجاد می‌شود تا اطلاعات کتاب جدیدی را وارد جدول نمایید.

دکمه‌های پیمایش به صورت زیر عمل می‌کنند.



۳-۳-۴ ایجاد دکمه

می‌خواهیم برای فرم، دکمه‌هایی طراحی کنیم تا کار با اجرای عملیات موردنظر روی فرم با ســـرعت و سهولت بیش‌تری انجام گیرد. برای ایجاد دکمه‌های پیمایشی یا عملیاتی می‌توانیم دکمه‌ای را از درون جعبه ابزار روی فرم قرار داده و ســـپس در رویداد On Click، دســـتورات موردنظر را به صورت کد یا ماکرو اضافه کنیم اما چون این مباحث در فصول بعدی کتاب تدریس می‌شـــوند، فعلاً از ویزارد تنظیم دکمه اســـتفاده خواهیم کرد.

۱. با انتخاب دکمه‌های قبلی و فشار دادن کلید Delete، آن‌ها را حذف کنید.

---

1 . Navigation Button

۲. در نوار ابزار، دکمه ویزارد کنترل‌ها را فعال نمایید.

۳. ابـــزار دکمه را انتخاب و یک دکمه زیر کادرهای متنی اضافه کنید. بلافاصله ویزارد تنظیم خصوصیات دکمه فعال می‌شود.

۴. در ســـتون سمت چپ، دکمه‌ها برحسب عمل‌کردی که دارند دسته‌بندی شده‌اند. با کلیک روی نام هر دسته، دکمه‌های قابل انتخاب در سمت راست نشان داده می‌شوند.

۵. دکمه Go To First Record را انتخاب و روی دکمه Next کلیک کنید.

۶. پس از تعیین تصویرِ روی دکمه، Finish را کلیک کنید تا دکمه روی فرم قرار گیرد.

دکمه‌های قابل تنظیم در این ویزارد، در گروه‌های اصلی زیر دسته‌بندی شده‌اند.

1- Navigation Record : برای جابه‌جایی در میان رکوردها و اصطلاحاً پیمایش میان آن‌ها کاربرد دارند.

2- Record Operation : عملیات‌هایی مانند حذف یا اضافه‌کردن رکورد را انجام می‌دهند.

3- Form Operations : برای اجرای دستورات موردنظر بر روی فرم به کار می‌روند.

4- Report Operations :برای اجرای دستورات موردنظر بر روی گزارش‌ها کاربرد دارند.

5- Miscellaneous : حاوی دستوراتی نظر چاپ جدول یا اجرای پرس‌وجو است.

مهم‌ترین دکمه‌های موجود در این ویزارد، درون جدول زیر توضیح داده شده‌اند.

گروه	دکمه	کارکرد
۱	Go To First Record	اولین رکورد را نمایش می‌دهد.
	Go To Last Record	آخرین رکورد را نمایش می‌دهد.
	Go To Next Record	رکورد بعدی را نمایش می‌دهد.
	Go To Previous Record	رکورد قبلی را نمایش می‌دهد.
۲	Add New Record	یک رکورد جدید ایجاد می‌کند.
	Delete Record	رکورد جاری را حذف می‌کند.
	Print Record	رکورد جاری را چاپ می‌کند.
	Undo Record	تغییرات رکورد را لغو می‌کند.
۳	Apply Form Filter	یک فیلتر روی رکوردهای فرم اعمال می‌کند.
	Close From	فرم را می‌بندد.
	Open Form	فرم موردنیاز را باز می‌کند.
	Print a Form	فرم موردنظر را چاپ می‌کند.
	Print Current Form	فرم جاری را چاپ می‌کند.
۴	Preview Report	پیش‌نمایش گزارش را نشان می‌دهد.
	Print Report	گزارش را چاپ می‌کند.
۵	Run Macro	ماکروی تعیین شده را اجرا می‌کند.
	Run Query	پرس‌وجوی تعیین شده را اجرا می‌نماید.

۷. دکمه‌های موردنیاز را به فرم اضافه کنید.

شماره	دکمه	شماره	دکمه	شماره	دکمه
۱	اولین رکورد	۲	رکورد قبل	۳	رکورد بعد
۴	آخرین رکورد	۵	جستجوی رکورد	۶	حذف رکورد جاری
۷	ایجاد رکورد جدید	۸	چاپ رکورد جاری	۹	بستن فرم

۸. عملیات درج رکورد جدید، ویرایش و حذف یک رکورد موجود و نیز چاپ آن را انجام دهید.

۴-۳-۴ ساخت فرم با ویزارد

در این بخش با روش ساخت یک فرم و قرار دادن کنترل‌های موردنظر روی آن آشنا شدید. به عنوان آخرین مطلب این بخش، یادآوری می‌کنم که برای ساخت سریع‌تر فرم‌های ساده در اکسس، یک ویزارد کارآمد وجود دارد که می‌توانید تمام یا بخشی از فرایند طراحی یک فرم را به آن واگذار نمایید.

۱. در ستون سمت چپ پنجره، جدول Tbl_Books را انتخاب کنید.

۲. در زبانه Create و قاب Forms، منوی More Forms را باز و روی گزینه Form Wizard کلیک نمایید.

۳. در پنجـــره اول ویزارد، جدول یـــا پرس‌وجوی موردنظر را انتخاب کنید. ســـپس در بخش Available Filelds، روی فیلد دلخواه کلیک کنید و سپس با فشاردادن دکمه > آن را به بخش فیلدهای انتخاب شده (Selected Fields) ببرید. پس از اتمام کار روی دکمه Next کلیک کنید.

نکته

با کلیک روی دکمه << همه فیلدها به بخش Selected Fields منتقل می‌شوند.

۴. در پنجره بعد باید یکی از چهار چیدمان[1] موجود را برای نمایش فیلدها انتخاب نمایید.

1 .Layout

این چهار نوع چیدمان عبارتند از :

۱. ستونی (Columnar)

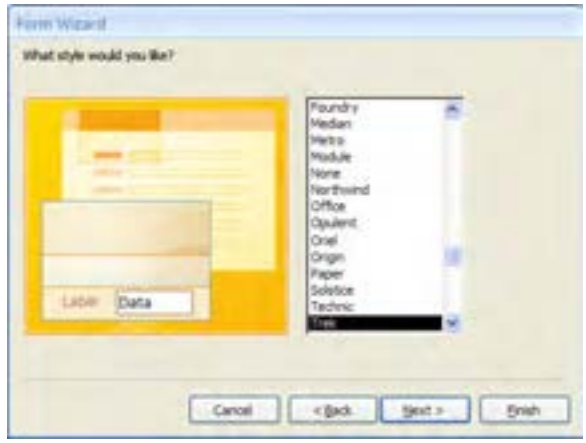
۲. همتراز (Justified)

۳. جدولی (Tabular)

۴. داده برگی (Datasheet)

۵. سپس یکی از سبک‌های[1] گرافیکی را برای قالب‌بندی فرم انتخاب کنید.

1 . Style



۶. پنجره آخر هم مخصوص وارد کردن عنوان فرم است. با کلیک روی دکمه Finish، فرم ساخته می‌شود و می‌توانید با آن کار کنید یا در نمایش طراحی، تغییراتی را روی فرم اعمال نمایید.

۴-۴ استفاده از زیرفرم[1] در فرم اصلی

وقتی پایگاه داده شـــما حاوی چند جدول مرتبط با هم باشـــد، طراحی فرم‌های مجزا برای هر کدام از جداول و ورود اطلاعات به صورت مســـتقل از هم راه‌حل چندان مناســـبی نیست چراکه جداول با کلیدهای

1 .Subform

خارجی (مثل شـــماره کتاب، شماره فاکتور و ...) به هم متصل هستند و برای وارد کردن هر رکورد باید این شماره‌ها را درج کنید که عملاً بسیار وقت‌گیر است و صحت ورود اطلاعات را هم تحت تأثیر قرار می‌دهد.

اکســـس برای غلبه بر چنین مشـــکلی، امکان استفاده از یک یا چند زیرفرم (فرم فرعی) را در فرم اصلی فراهم آورده اســـت. در مثالی که دنبال می‌کنیم قصد داریم فرمی ایجاد کنیم که مشـــخصات مشـــتری و فاکتورهای صادر شده برای وی را نشان دهد و با کلیک روی هر فاکتور، جزییات فاکتور یعنی اقلام موجود در آن نشـــان داده شـــود. پس فرم اصلی حاوی جدول Tbl_Customers اســـت و جداول Tbl_Factors و Tbl_FactorDetails به عنوان زیرفرم، کار ورود و بررسی داده‌ها را تسهیل خواهند کرد.

۱. پیش از راه‌اندازی ویزارد، میان جداول و پرس‌وجوهایی که قصد دارید در فرم از آن‌ها اســـتفاده کنید ارتباطات موردنیاز را برقرار نمایید.

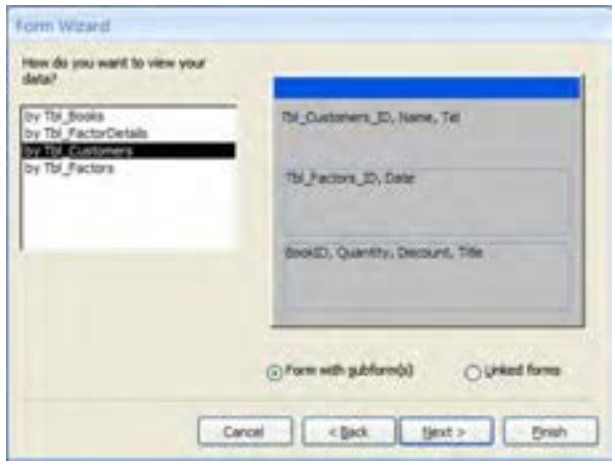
۲. در زبانه Create و قاب Forms، منوی More Forms را باز و روی گزینه Form Wizard کلیک نمایید.

۳. جدول Tbl_Customers را انتخاب و فیلدهای موردنظر را به بخش Selected Fields اضافه کنید.

۴. این کار را برای جداول Tbl_Factors و Tbl_FactorDetails هم (در همین پنجره) انجام دهید.

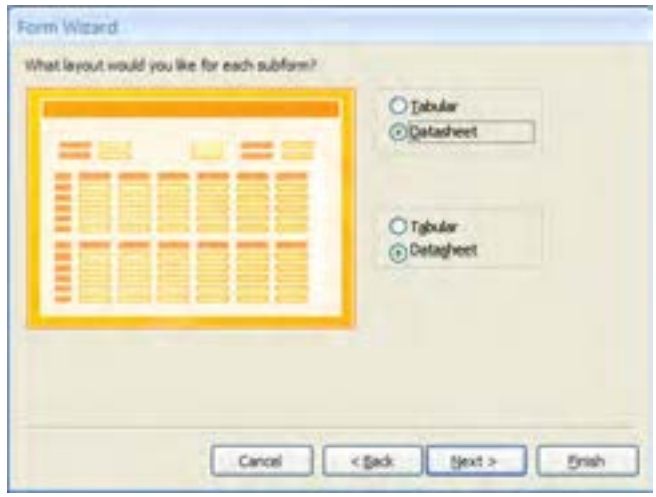
۵. می‌خواهیـــم در زیرفرم جزییات فاکتور، علاوه بر شـــماره کتاب، نام آن هم درج شـــود بنابراین جدول Tbl_Books را انتخاب و فیلد Title آن را به بخش فیلدهای انتخابی اضافه کنید. پس از تعیین همه فیلدها روی دکمه Next کلیک کنید.

۶. در پنجره بعد باید تعیین کنید که اطلاعات بر مبنای کدام جدول نمایش داده شوند. می‌خواهیم اطلاعات فاکتورها را بر اساس نام مشتریان نمایش دهیم بنابراین، گزینه by Tbl_Customers را انتخاب کنید.



۷. بـــا انتخاب گزینه Form with subform(s) پرس‌وجوی بازیابـــی اطلاعات برای فرم اصلی و زیرفرم‌ها نشان داده می‌شود. در این مثال فرم اصلی حاوی دو زیرفرم است. روی دکمه Next کلیک نمایید.

۸. در پنجره بعد باید طرح‌بندی هر کدام از زیرفرم‌ها را مشـــخص کنید. هر دو گزینه را روی Datasheet تنظیم و روی دکمه Next کلیک نمایید.

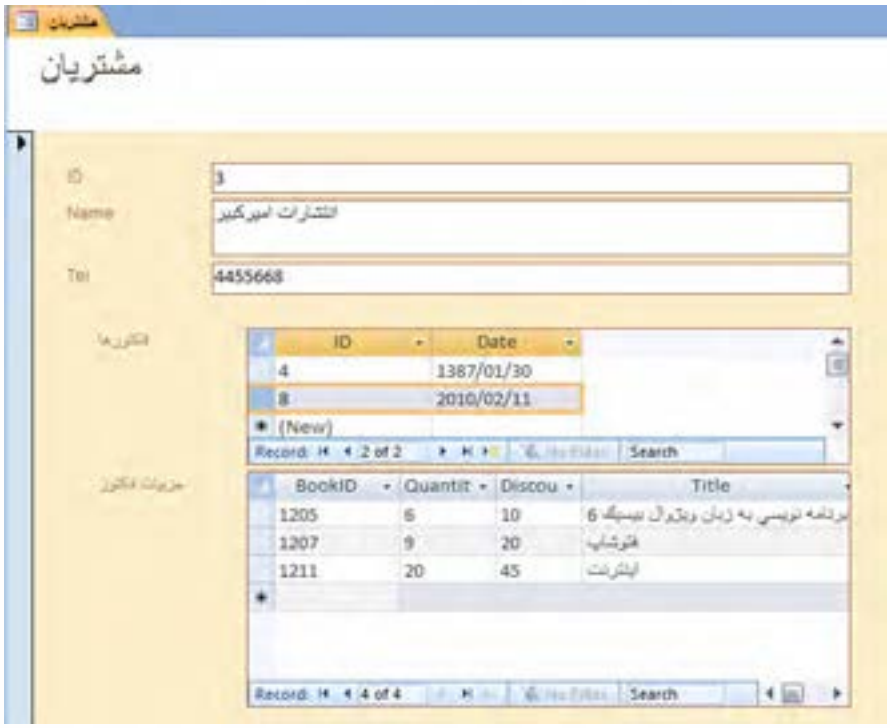


۹. پس از تعیین ســـبک فرم، به آخرین پنجره ویزارد می‌رسیم. برای هر یک از بخش‌های فرم یک عنوان مناسب وارد و روی دکمه Finish کلیک کنید.

۱۰. به لیست فرم‌های پایگاه داده نگاه کنید. سه فرم جدید با اسامی انتخابی، ایجاد شده است.

۱۱. فرم اصلی را در حالت Form View ببینید و با آن کار کنید.

با تغییر رکورد مشـــتری، فاکتورهای او در زیرفرم «فاکتورها» ظاهر می‌شوند و با کلیک روی هر فاکتور، اقلام موجود در آن درون زیرفرم «جزییات فاکتور» نمایش داده می‌شـــود. به این ترتیب می‌توانید تغییرات دلخواه را روی مشخصات مشتری، فاکتور و اقلام آن ایجاد کنید.



۴-۵ تغییر خواص زیرفرم‌ها

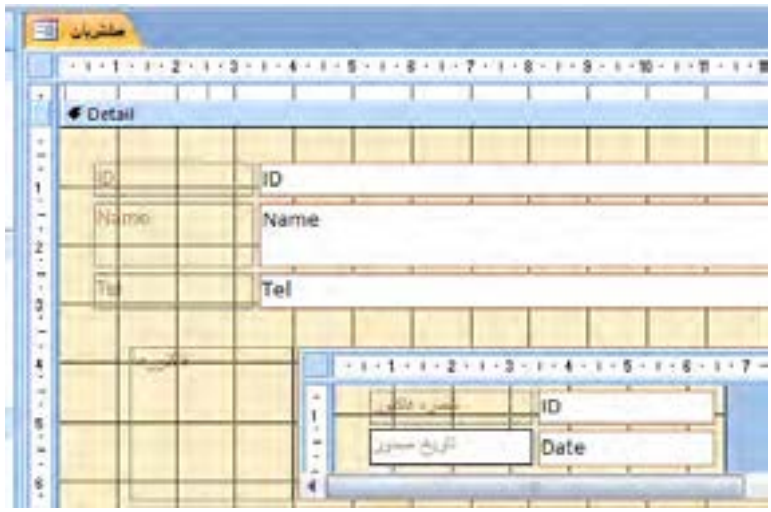
اغلب اوقات لازم می‌شود تغییراتی را روی زیرفرم‌های ایجاد شده اعمال نمایید تا ظاهر فرم نهایی بهبود پیدا کند و کارکردن با آن ساده‌تر شود. تغییر خواص زیرفرم به صورت مستقل یا از طریق نمای طراحیِ فرم اصلی باعث انعکاس روی زیرفرم و نهایتاً فرم اصلی می‌شود.

۱. در بخش Forms برنامه روی زیرفرم «جزییات فاکتور» راســت‌کلیک نموده و گزینه Design View را انتخاب کنید.

۲. فرم در نمای طراحی باز می‌شــود. نام برچســب‌ها را به عبارات فارسی تغییر دهید. تغییرات را ذخیره نموده و این فرم را ببندید.

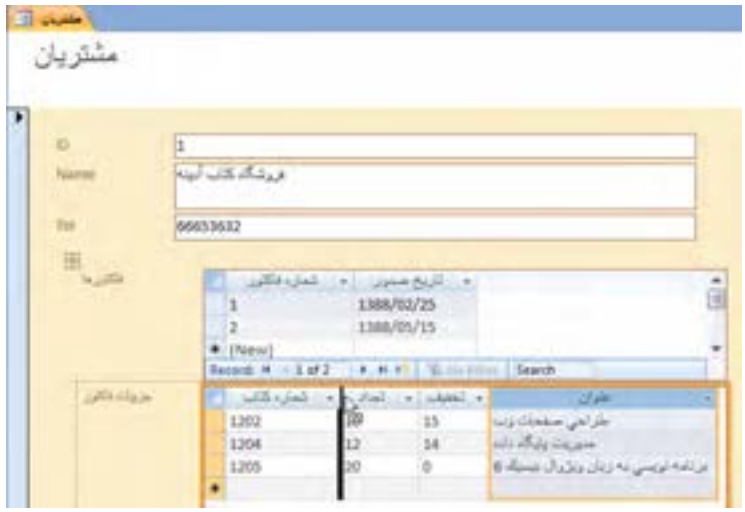
۳. فرم اصلی (مشتریان) را در نمای فرم باز کنید. همان‌طور که می‌بینید، تغییرات روی این فرم منعکس شده است.

۴. فرم اصلی را در نمای طراحی باز و برچسب‌های زیرفرم فاکتورها را فارسی کنید. تغییرات را ذخیره نمایید. این تغییرات روی زیرفرم اعمال می‌شود.



۵. فرم اصلی را در نمای Layout باز و در زیرفرم «جزییات فاکتور» با استفاده از روش کشیدن و رها کردن[1]، فیلد عنوان کتاب را کنار شماره کتاب ببرید. با ذخیره کردن تغییرات، این جابه‌جایی روی فرم فرعی اعمال می‌شود.

1. Drag and Drop


مشتریان
ID
1
Name
فروشگاه کتاب آینده
Tel
66653632
1
1388/02/25
2
1388/01/15
(New)
Search
1202
15
1204
12
14
1205
20
0

Create a form that contains two subforms

This procedure creates a form and two subforms with the following characteristics:

- The main form has a one-to-many relationship with the first subform.
- The first subform has a one-to-many relationship with the second subform.
- The main form contains both the subform controls.

Create the form

1. On the Create tab, in the Forms group, click More Forms, and then click Form Wizard.

2. On the first page of the wizard, in the Tables/Queries drop-down list, select the table or query for the main form. For example, suppose you want to create a Customers form that has two subforms — an Orders subform and an Order Details subform. Select the Customers table (the “one” side of the first one-to-many relationship).

3. Double-click the fields that you want to include from this table or query.

4. On the same page of the wizard, in the Tables/Queries drop-down list, select the table or query for the first subform. For this example, click the Orders table (the “many” side of the first one-to-many relationship), and then double-click the fields that you want to include from this table or query.

5. On the same page of the wizard, in the Tables/Queries drop-down list, select the table or query for the second subform. For this example, select the Order Details table (the “many” side of the second one-to-many relationship), and then double-click the fields that you want to include from this table or query.

6. When you click Next, assuming that you set up the relationships correctly before starting the wizard, the wizard asks How do you want to view your data? — that is, by which table or query. For this example, to create the Customers form, click By Customers.

7. Select the Form with subform(s) option.

8. Follow the directions on the remaining pages of the wizard. After you click Finish, Access creates a main form that contains two subform controls and also creates two other form objects — one for each subform.

۱. روش ایجاد یک فرم را که حاوی دو زیر فرم باشد شرح دهید.

۲. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. انواع کنترل‌های موجود در اکسس را توضیح دهید.

۲. بدون استفاده از ویزارد، یک فرم برای ویرایش اطلاعات مشتریان ایجاد کنید.

۳. فعال کردن دکمه Snap to Grid چه تأثیری روی جابه‌جایی کنترل‌ها دارد؟

۴. با استفاده از ویزارد، فرمی ایجاد کنید که مشخصات فاکتورها را نشان دهد و با کلیک روی هر فاکتور، جزییات آن در یک زیرفرم ظاهر شود.

فصل پنجم

فصل پنجم : کار با ماکروها

کار با ماکروها[1]، یکی از مباحث پیشرفته و کاربردی در استفاده از پایگاه داده اکسس محسوب می‌شود. با تسلط بر این قابلیت شما می‌توانید بدون نیاز به یادگیری زبان برنامه نویسی VBA[2]، نیازهای خود را در زمینه خودکارسازی کارها رفع نمایید.

۵-۱ اصول ایجاد و ویرایش ماکروها

پیش از شروع کار با ماکروها ابتدا باید با تعریف این ابزار کارآمد اکسس آشنا شوید و بدانید در چه مواردی از آن استفاده می‌شود.

۵-۱-۱ ماکرو چیست؟

ماکروی اکسس ابزاری است که به شما کمک می‌کند تا مجموعه‌ای از اعمال[3] را به صورت خودکار درون یک فرم یا گزارش و نیز در پاسخ به وقوع رویدادی روی یک کنترل اجرا نمایید؛ برای مثال ماکرویی بنویسید که با کلیک کردن روی یک دکمه، یکی از گزارش‌های موجود در پایگاه داده، بر حسب یکی از داده‌ها، مرتب‌سازی و چاپ شود.

---

1 . Macro

2 . Visual Basic for Applications

3 . Actions

ماکرو را می‌توانید یک زبان برنامه‌نویسی ساده تصور کنید که لیستی از اعمال را در اختیار شما قرار می‌دهد تا بدون نیاز به تسلط بر یک زبان برنامه‌نویسی مانند VBA، کارکردهای موردنظرتان را به فرم‌ها، گزارش‌ها و کنترل‌ها اضافه نمایید. اعمال قابل اجرا درون یک ماکرو، از طریق زبان VBA هم قابل پیاده‌سازی هستند اما اغلب افراد ترجیح می‌دهند تا حد امکان خود را درگیر کدنویسی نکرده و نیازهای خود را با ماکروها برطرف نمایند.

هر ماکرو می‌تواند حاوی یک یا چند عمل باشد و کارهایی مانند موارد زیر را انجام دهد:

۱. باز و بسته کردن فرم‌ها و گزارش‌ها
۲. تعیین نحوه مرتب‌سازی داده‌ها در گزارش‌ها
۳. تنظیم ویژگی‌های یک کنترل بر اساس مقادیر سایر کنترل‌ها در حین اجرای برنامه
۴. اعتبارسنجی داده‌های وارد شده و صدور پیغام‌های مناسب در صورت ورود داده نامعتبر
۵. انجام یک عملیات در شروع کار با پایگاه داده
۶. خودکار سازی کارهای تکراری در محیط پایگاه داده

ماکروها را می‌توان به صورت یک فایل مستقل ایجاد و ذخیره‌سازی نمود و سپس آن را به رویدادی از یک کنترل (مانند رویداد OnClick یک دکمه) نسبت داد. علاوه بر این، ایجاد ماکروهای تعبیه شده[1] هم امکان‌پذیر است که هر چند در فایل پایگاه داده ذخیره می‌گردند اما فایل مستقلی محسوب نمی‌شوند. در ادامه این فصل با هر دو روش تولید ماکرو آشنا خواهید شد.

۵-۱-۲ ایجاد ماکرو

وقتی ماکرویی را ایجاد می‌کنید باید در مورد دو موضوع تصمیم مناسبی اتخاذ کنید :

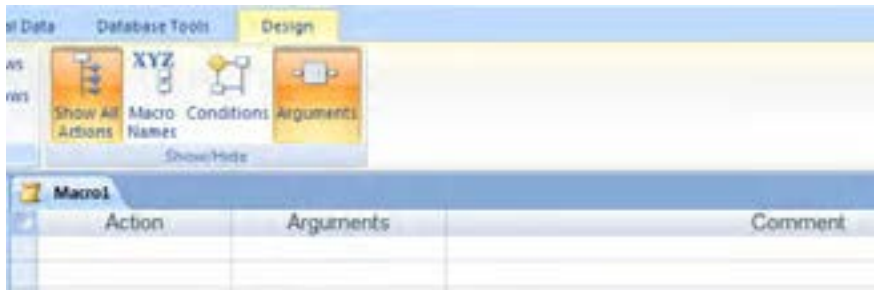
الف) قرار است چه عمل یا اعمالی در قالب ماکرو انجام شود. ضمناً هر عمل چه آرگومان‌هایی داشته باشد و با چه شرطی اجرا گردد.

ب) ماکرو در صورت وقوع چه رویدادی اجرا گردد.

در این بخش، موضوع اول را با جزییات کامل مورد بررسی قرار خواهیم داد.

۱. در زبانه Create و قاب Others روی دکمه زردرنگ Macro کلیک کنید.
۲. پنجره‌ای با سه ستون Action، Arguments و Comment ظاهر می‌شود.

1 . Embedded



کارکرد هر ستون به صورت زیر است :

الف) Action : فرمان انتخاب شده برای اجرا را نشان می‌دهد.

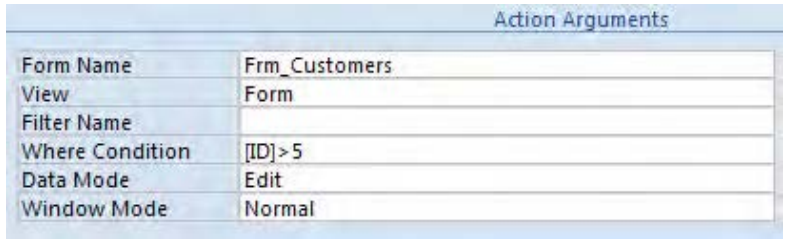
ب) Arguments : بـــا انتخاب هر فرمان باید مجموعـــه‌ای از آرگومان‌ها را مقداردهی کنید تا جزییات اجرای دستور مشخص شود.

ج) Comment : این ســـتون برای درج توضیحات در مورد فرمان درنظر گرفته شـــده اما استفاده از آن اجباری نیست.

۳. روی دکمه Show All Actions کلیک کنید تا همه فرمان‌ها در لیست Action قابل مشاهده باشند.

۴. در ستون Action کلیک نموده و از لیست فرمان‌های موجود، گزینه OpenForm را انتخاب کنید.

۵. در ســـتون Arguments، آرگومان‌های پیش‌فرض ظاهر می‌شـــود و باید بـــا مراجعه به بخش Action Arguments که در پایین پنجره قرار دارد، آرگومان‌ها را روی مقادیر موردنظر تنظیم کنید.

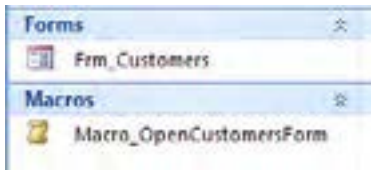


همان‌طـــور که قبلاً اشـــاره کردم هر فرمان، آرگومان‌های مخصوص به خـــود دارد؛ برای مثال در فرمان بازکردن فرم، تعیین نام فرم الزامی است اما در مقداردهی آرگومان‌هایی مثل حالت داده‌ها و شرط جداسازی داده‌ها اجباری وجود ندارد.

۶. برای مشـــاهده نحوه کار ماکرو، در زبانه Desingn و قاب Tools روی دکمه Run کلیک کنید تا نتیجه اجرای ماکرو که در این مثال باز شدن فرم مشتریانِ دارای شماره بالای ۵ است ظاهر شود.

۷. روی آیکن  کلیک نموده و با وارد کردن نام مناسب، ماکروی ساخته شده را ذخیره نمایید.

۸. این ماکرو در قالب یک فایل مستقل ذخیره شده و در نوار پیمایش نشان داده می‌شود.



یک روش سریع برای اجرا و آزمایش ماکرو این است که در نوار پیمایش روی نام ماکرو دوبار کلیک کنید.

۳-۱-۵ فرمان‌های ماکرو

وقتی دکمه Show All Actions را فعال می‌کنید همه فرمان‌های قابل استفاده در ماکرو درون لیست Action دیده می‌شوند. کارکرد تعدادی از این فرمان‌ها در جدول صفحه‌ی بعد شرح داده شده است:

اعمال ماکرو	کارکرد
ApplyFilter	روی داده‌های موجود در جـــدول، فرم یا گزارش، بر اساس یک معیار خاص عمل جداسازی یا مرتب‌سازی را انجام می‌دهد.
FindNext, FindRecord, GoToControl, GoToPage, GoToRecord	امکان جابه‌جایـــی بین کنترل‌ها، رکوردها و صفحات را فراهم می‌آورند.
OpenQuery, Run SQL, RunApp, RunMacro	پرس‌وجو، برنامه یا ماکروی دیگری را اجرا می‌کنند.
CancelEvent, StopAllMacros, StopMacro	بدون خروج از محیط اکسس، اجرای ماکرو را متوقف می‌کنند.
TransferDatabase, TransferSharePoittList, Transfer-Spreadsheet, TransferSQLDatabase, TransferText	امکان انتقال داده‌ها را میان برنامه‌های مختلف فراهم می‌آورند.
Close, OpenDataAccesspage, OpenForm, Open-Query, OpenReport, OpenTable, OpenView	برای باز و بســـته کردن اشیاء پایگاه داده و دستکاری داده‌ها کاربرد دارند.
SetValue, SetProperty	مقـــدار یک فیلد یا کنترل و نیز ویژگی‌های کنترل را تنظیم می‌کنند.
Echo, MsgBox	برای نشان دادن یک پیغام به کاربر استفاده می‌شوند.
PrintOut	چاپ یک فرم، گزارش و ... را امکان‌پذیر می‌کند.
Beep	بوق سیستم را به صدا در می‌آورد
Quit	برنامه اکسس را می‌بندد.

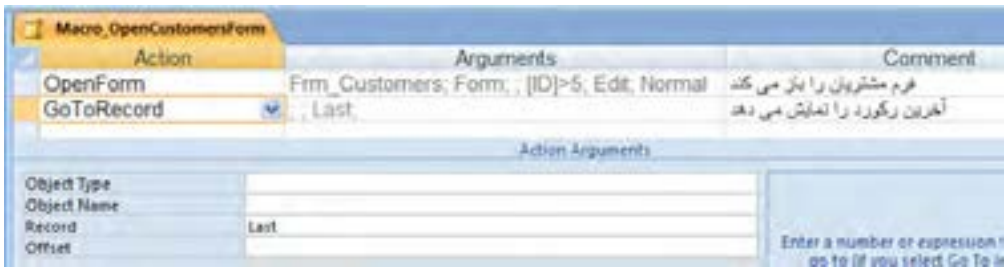
۵-۱-۴ ویرایش ماکرو

قصد داریم فرمان دیگری به ماکرو اضافه کنیم، تا پس از باز شدن فرم، آخرین رکورد نشان داده شود.

۱. در نوار پیمایش اکسس روی ماکرویی که در بخش قبل ساختید، راست کلیک نموده و گزینه Design View را انتخاب کنید تا ماکرو در حالت ویرایش قرار گیرد.

۲. در ستون Action و ردیف دوم، فرمان GoToRecord را انتخاب نمایید.

۳. در بخش Argument، آرگومان Record را روی Last تنظیم کنید.



۴. همه تنظیماتی که روی آرگومان‌ها ایجاد می‌کنید، به صورت خاکستری رنگ در ستون Arguments نشان داده می‌شوند.

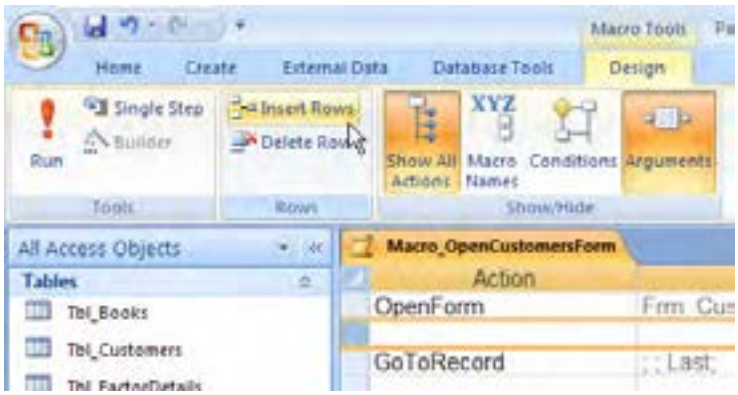
۵. تغییرات ماکرو را ذخیره و با کلیک روی دکمه Run آن را اجرا کنید.

اگر در قاب Tools از زبانه Design دکمه Single Step را فعال نمایید، با هربار کلیک روی دکمه Run، فقط یک سطر از فرمان‌های ماکرو اجرا می‌شود.

می‌خواهیم بین دو فرمان موجود در ماکرو یک فرمان فیلتر کردن جدید اضافه کنیم. برای انجام این کار :

۱. روی فرمان دوم کلیک کنید تا انتخاب شود.

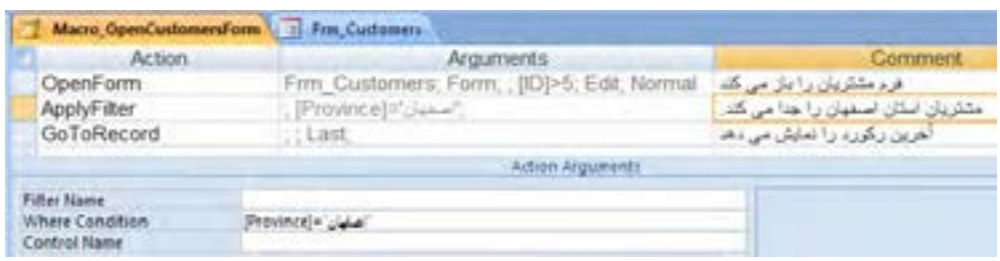
۲. در زبانه Design و قاب Rows روی دکمه Insert Rows کلیک کنید تا یک سطر خالی به بالای ردیف انتخاب شده اضافه گردد.



۳. در ستون Action دستور ApplyFilter را انتخاب نموده و در بخش Argument Actions در کادر Where Conditions شرط موردنظر را وارد کنید.

۴. برای این‌که در فرم، فقط مشتریان استان اصفهان نشان داده شود باید عبارت SQL زیر را در کادر Where Condition وارد کنیم.

Province='اصفهان'



۵. تغییرات ماکرو را ذخیره و با کلیک روی دکمه Run آن را اجرا کنید.

۵-۲ استفاده از ماکروها در فرم‌ها

هنگام طراحی پایگاه داده، اجرای ماکرو را مشروط به وقوع یک رویداد[1] می‌کنند. باز شدن پایگاه داده، کلیک کردن روی یک دکمه، باز شدن یک گزارش، تغییر دادن محتوای یک کنترل و ... نمونه‌ای از این رویدادها هستند. همان‌طور که قبلاً اشاره کردم شما می‌توانید ماکرو را به صورت یک فایل مستقل ذخیره نموده و سپس آن را به یک رویداد نسبت دهید؛ کاری که در ادامه با آن آشنا خواهید شد.

۵-۲-۱ انتساب ماکرو به رویداد

۱. در زبانه Forms روی دکمه Form Design کلیک کنید تا یک فرم خام در محیط طراحی باز شود.

۲. روی فرم یک دکمه قرار داده و در صورت ظاهر شدن ویزارد تنظیم عمل‌کرد آن، روی دکمه Cancel کلیک کنید تا ویزارد بسته شود.

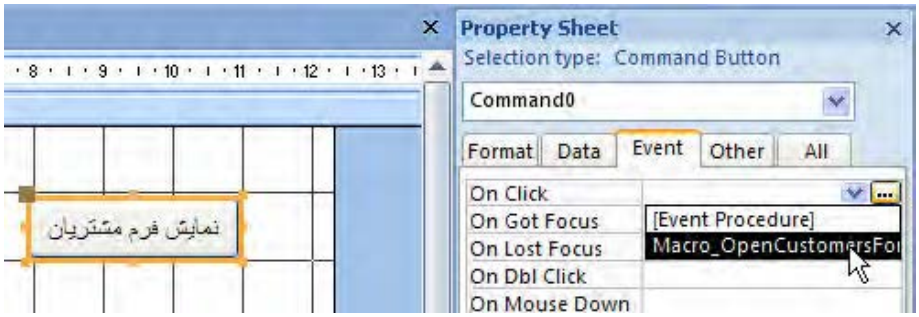
۳. در زبانه Design و قاب Tools، دکمه Property Sheet را فعال کنید تا برگه تنظیم ویژگی‌های کنترل ظاهر شود.

۴. روی دکمه کلیک نموده و سپس در زبانه Format از پنجره تنظیم ویژگی‌ها، ویژگی Caption را به عبارت «نمایش فرم مشتریان» تغییر دهید.

---

1 . Event

۵. برای انتســاب ماکرو به یکی از رویدادهای دکمه، به ســراغ زبانه Events رفته و منوی روبروی عبارت On Click را باز کنید.



۶. لیســت ماکروهای ساخته شــده در پایگاه داده نشان داده می‌شود. روی نام ماکرویی که در بخش قبل ساختیم کلیک کنید.

با انجام این کار به اکسس اعلام کردیم که اگر این دکمه کلیک شد (یا به بیان بهتر، رویداد On Click این کنترل به وقوع پیوست)، ماکروی انتخاب شده اجرا شود.

۷. فرم را با نام Frm_Menu ذخیره نموده و آن را باز کنید. با کلیک روی دکمه، فرامین موجود در ماکرو اجرا می‌شود و فرم مشتریان با فیلترها و شرط‌هایی که اعمال کردیم به نمایش در می‌آید.

۲-۲-۵ انواع رویدادها

فرم‌هــا، گزارش‌ها و هر یــک از کنترل‌ها، رویدادهای مخصوص به خود دارند کــه البته تعدادی از این رویدادها بین آن‌ها مشترک است. برای هر یک از این اشیاء، رویدادی که بیش‌تر مورد استفاده قرار می‌گیرد به عنوان رویداد پیش‌فرض[1] تعریف شده است؛ برای مثال، رویداد پیش‌فرض کنترل دکمه، On Click است اما در مورد گزارش، On Open رویداد پیش‌فرض محسوب می‌شود.

رویدادهایی که بیش‌ترین کاربرد را در طراحی برنامه‌های پایگاه داده دارند در جدول صفحه بعد مشاهده می‌کنید.

1 . Default

نام	زمان وقوع رویداد	کاربرد
On Click	کلیک شدن کنترل یا فرم	اجرای فرمان‌های ماکرو
On Open	هنگام باز شدن فرم قبل از نمایش رکوردها در صفحه	باز کردن، بستن، یا کمینه‌کردن فرم‌های دیگر و بیشینه‌کردن فرم جاری
On Enter	پیش از این که مکان‌نما از کنترل قبلی به کنترل حاوی رویداد منتقل شود.	نمایش اطلاعاتی در مورد داده‌ها حین ورود به کنترل یا درخواست کلمه عبور از کاربر
Before Update	هنگامی که مکان‌نما یک رکورد یا کنترل را ترک می‌کند و قبل از ثبت تغییرات در پایگاه داده	پیغامی را برای تأیید تغییرات صادر یا داده‌های وارد شده را اعتبارسنجی می‌کند.
After Update	پس از ثبت تغییر رکوردها در پایگاه داده یا خارج شدن مکان‌نما از کنترل	بروزرسانی داده‌ها در کنترل‌ها، فرم‌ها و گزارش‌ها. همچنین انتقال داده‌های جدید به سایر برنامه‌های کاربردی

۳-۲-۵ ایجاد ماکروی شرطی

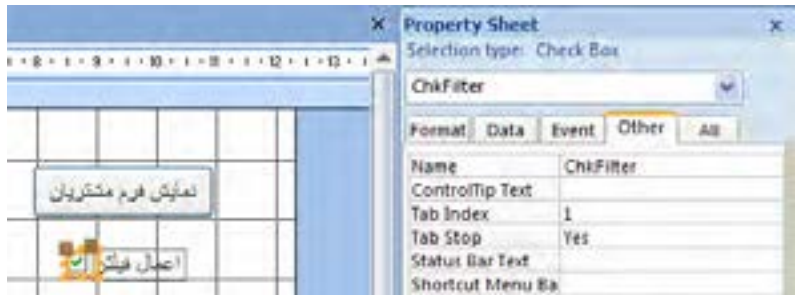
یکی از ویژگی‌های ماکروها که قدرت آن‌ها را در ایجاد برنامه‌های پایگاه داده افزایش می‌دهد، امکان شرطی کردن اجرای فرمان‌هاست. برای مثال می‌توانید اجرای هر یک از فرمان‌های موجود در ماکرو را به وقوع یک حالت خاص مشروط کنید. می‌خواهیم در فرم Frm_Menu یک کادر تأیید[1] قرار دهیم تا کاربر با تیک زدن آن نشان دهد تمایل به اعمال فرمان ApplyFilter در ماکرویی که ساخته‌ایم دارد.

۱. فرم Frm_Menu را در حالت طراحی باز کنید.

۲. یک کنترل کادر تأیید روی آن قرار دهید.

۳. با دوبار کلیک روی برچسب، عبارت «اعمال فیلتر» را تایپ نمایید.

۴. روی کادر تأیید کلیک نموده و در زبانه Other از برگه ویژگی‌ها، ویژگی Name را به ChkFilter تغییر دهید.



1 . CheckBox

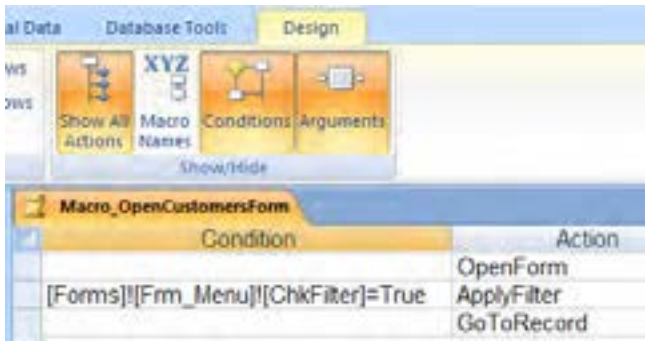
۵. فرم Frm_Menu را ذخیره کنید و ببندید.

۶. این بار ماکرویی را که در بخش‌های قبل ایجاد کردیم، در محیط طراحی باز کنید.

۷. در زبانه Design و قاب Show/Hide روی دکمه Conditions کلیک کنید تا ستون Condition هم به ستون‌های قبلی اضافه شود.

۸. در ردیف فرمان ApplyFilter عبارت زیر را در ستون Condition اضافه کنید.

[Forms]![Frm_Menu]![ChkFilter]=True



این عبارت یک شرط است و اعلام می‌کند اگر کادر تأیید در فرم Frm_Menu تیک خورده است (مقدار آن True است) فرمان ApplyFilter را اجرا کند.

۹. ماکرو را ذخیره نموده و آن را ببندید.

۱۰. حال فرم Frm_Menu را باز و بدون تیک زدن گزینه «اعمال فیلتر» روی دکمه «نمایش فرم مشتریان» کلیک کنید. همان‌طور که می بینید چون کادر تأیید تیک نخورده و شرط نادرست است، فرمان مرتبط با آن اجرا نمی‌شود و مشتریان همه شهرها نشان داده می‌شوند.

نمونه‌هایی از شرط‌های قابل پیاده‌سازی در ماکرو را در جدول زیر مشاهده می‌کنید.

حالتی که ماکروی مرتبط اجرا می‌شود	شرط
مقدار فیلد Tilte در فرمی که ماکرو را اجرا کرده تهی باشد.	IsNull([Title])
مقدار فیلد City در فرمی که ماکرو را اجرا نموده برابر با 'تهران' باشد.	City=''تهران'
مقدار فیلد City در فرم Frm_Customers برابر با 'اصفهان' باشد.	Forms![Frm_Customers]!City='اصفهان'
فاکتور نمایش داده شده در فرم Frm_Factors در سال ۱۳۸۸ صادر شده باشد.	Forms![Frm_Factors]![Date] BETWEEN #1388/01/01#AND#1388/12/29#
فیلد City برابر با 'تهران' و تعداد نویسه‌های شماره تلفن برابر ۸ باشد.	[City]='تهران' AND LEN([Tel])=8
در کادر محاوره‌ای که پیغام «تغییرات ذخیره شود؟» را نمایش می‌دهد، روی دکمه OK کلیک کنید.	MsgBox(«تغییرات ذخیره شود؟»,1)=1
تعداد ورودی‌های فیلد ID در جدول Tbl_Factors بیش‌تر از ۱۰۰ باشد.	DCount([ID],"Tbl_Factors")>100

۵-۲-۴ ایجاد ماکروی تعبیه شده

در مباحث قبلی با روش ایجاد یک ماکرو به صورت فایل مستقل و انتساب آن به یک رویداد کنترل یا فرم آشنا شدید. در این بخش روش تولید ماکروهای تعبیه شده را فرا می‌گیرید؛ ماکروهایی که در نوار پیمایش به صورت یک شیء مستقل نشان داده نمی‌شوند و جزیی از فایل پایگاه داده هستند.

یکی از کاربردهای مهم ماکروها، اعتبارسنجی[1] داده‌های ورودی در فرم‌های برنامه است تا از ثبت داده‌های غیرمعتبر در پایگاه داده جلوگیری گردد یا به کاربر هشدار مناسب در مورد اشتباهات داده شود. به عنوان نمونه، تلفن‌های شهر تهران ۸ رقمی است و اگر در فرم ورود اطلاعات مشتریان، فیلد City برابر «تهران» باشد اما طول عبارت وارد شده در فیلد Tel، ۸ نویسه نباشد باید پیغام مناسبی روی فرم درج شود.

توضیح : در این مثال فرض کرده‌ایم مشتریان ما در شهر تهران از شماره‌های خاص (مثلاً چهار رقمی) استفاده نمی‌کنند.

1 . Validation

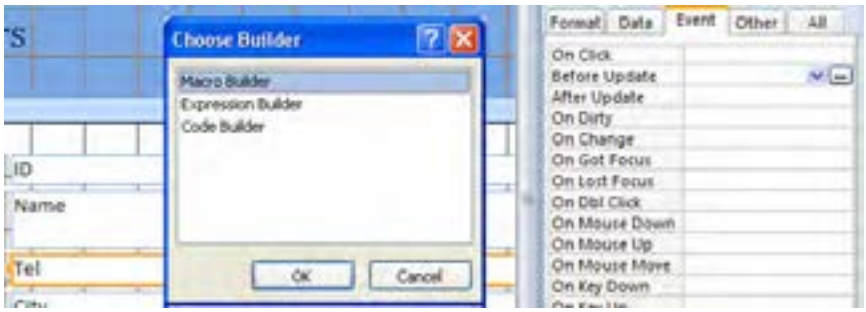
۱. فرم Frm_Customers را در نمای طراحی باز کنید.

۲. یک کادر متنی (Text Box) به پایین فرم اضافه نموده و برچسب آن را به عبارت «پیغام» تغییر دهید.

۳. روی کادر متنـــی کلیک نموده و در زبانه Other از برگه تنظیم ویژگی‌ها، ویژگی Name را به txtMsg تغییر دهید.

۴. روی کادر متنـــی Tel کلیک نموده و در زبانـــه Event از برگه ویژگی‌ها، روی عبارت Before Update کلیک کنید تا دکمه‌ای کوچک در انتهای سطر این رویداد ظاهر شود.

۵. روی این دکمه کلیک نموده و در پنجره ظاهر شـــده، گزینه Macro Builder را انتخاب و روی دکمه OK کلیک کنید.



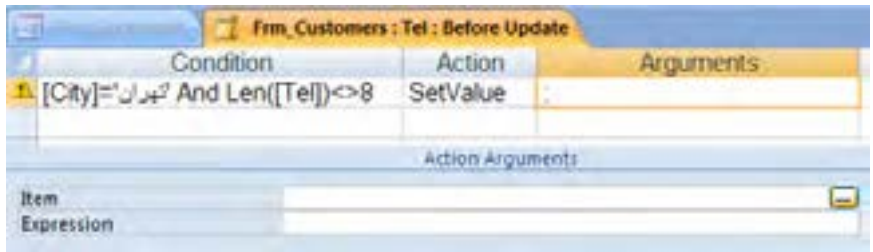
۶. پنجره ساخت ماکرو ظاهر می‌شود. دکمه‌های Show All Actions و Conditions را فعال کنید.

۷. در ســـتون Condition عبارت زیر را وارد کنید. به این معنی که در صورت مســـاوی بودن مقدار درون فیلد City با واژه «تهران» و هشت حرفی نبودن مقدار درون فیلد Tel، فرمان اجرا شود.

[City]='تهران' And Len([Tel])<>8

۸. در ستون Action عبارت SetValue را انتخاب کنید. این فرمان برای تغییر مقدار یک کنترل در حین اجرای برنامه کاربرد دارد.

به سراغ بخش Action Arguments بروید تا آرگومان‌های فرمان را هم تنظیم کنیم.

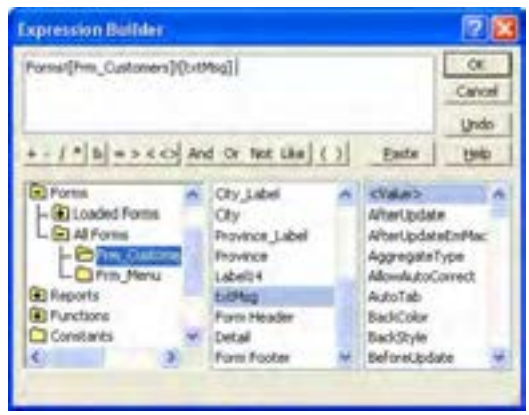


۹. درون کادر Item کلیک و روی دکمه‌ای که در انتهای آن ظاهر می‌شود کلیک کنید.

۱۰. پنجره Expression Builder که حاوی ابزار لازم جهت ساخت عبارات مورد استفاده در محیط اکسس است ظاهر می‌شود.

۱۱. از ستون سمت چپ، فرم موردنظر را انتخاب و در ستون وسط روی کنترل txtMsg کلیک کنید.

۱۲. دکمه Paste را بزنید تا عبارت قابل فهم برای اکسس جهت شناسایی کنترل موردنظر شما ساخته شود. روی دکمه OK کلیک کنید تا این پنجره بسته شود.



۱۳. درون کادر Expression هم عبارت موردنظر برای نمایش در کادر پیغام را وارد کنید. دقت داشته باشد

که در اطراف این عبارت، علامت نقل قول قرار دهید.

Action Arguments	
Item	[Forms]![Frm_Customers]![txtMsg]
Expression	'خطا در ورود اطلاعات: شماره تلفن باید 8 رقمی باشد'

۱۴. برای حالت صحیح بودن مقدار وارد شـــده هم فرمان پاک کردن کادر متنی txtMsg را به ماکرو اضافه کنید تا پیغام باقی‌مانده از درج قبلی، پاک شود.

۱۵. ماکرو را ذخیره کنید و پنجره طراحی را ببندید. همان‌طور که متوجه شـــدید نیازی به تعیین نام برای ماکرو نیست چون در فرم ادغام و همراه با آن ذخیره می‌شود.

۱۶. تغییرات ایجاد شده روی فرم را ذخیره نموده و آن را ببندید.

۱۷. با دوبار کلیک روی نام فرم در نوار پیمایش، فرم را در حالت ویرایش اطلاعات باز کنید.

۱۸. اگر در فیلد City، عبارت «تهران» وجود داشـــته باشـــد و تعداد ارقام شـــماره تلفن مخالف ۸ باشد، به محض خروج مکان‌نما از فیلد Tel پیغام هشدار در کادر متنی ظاهر می‌شود.

اگر هنگام درج یک رکورد جدید، ابتدا یک تلفن ۷ رقمی وارد و ســـپس عبارت «تهران» را در فیلد City بنویسیم آیا پیغام خطا ظاهر می‌شود؟

پاســخ منفی اســت. ماکرو روی کادر متنی Tel نوشته شــده و هنگام خروج مکان‌نما از آن، شرط اجرای ماکرو بررسی می‌شود. چون فیلد City در آن لحظه خالی است طبق جدول :

مقدار	شرط
نادرست	[City]='تهران'
درست	Len([Tel])<>8

نتیجه AND منطقی نادرست است و فرمان اجرا نخواهد شد.

برای تکمیل کار باید همین ماکرو را روی کادر متنی City هم بنویســید تا به محض خروج مکان‌نما از هر کدام از فیلدها، اجرای ماکرو بررســی شــود. در این تمرین با روش دیگری برای ایجاد ماکروهای تعبیه شده آشنا خواهید شد.

۱. فرم Frm_Customers را در حالت طراحی قرار دهید.

۲. این بار به جای استفاده از برگه خصوصیات، روی کادر متنی City راست کلیک نموده و گزینه Build Event را انتخاب کنید.

۳. گزینه Macro Builder را انتخاب و روی دکمه OK کلیک کنید.

در این حالت، ماکرو روی رویداد پیش‌فرض کنترل نوشته می‌شود. با رویدادهای پیش‌فرض در مباحث قبل آشنا شدید؛ حالا این نکته را هم یاد بگیرید که رویداد پیش‌فرض کادر متنی، Before Update است.

۴. به این ترتیب پنجره ماکرونویسی باز می‌شود و فرمان‌ها و آرگومان‌های موردنیاز مانند تمرین قبل است.

اگر به جای شــماره تلفن۸ رقمی یک کلمه ۸ حرفی وارد شود آیا ماکرو قادر به

تشخیص خطا هست؟

خیر! چون در کد نوشـــته شده از تابع تعیین طول رشته استفاده شده و این تابع تفاوتی میان نویســـه‌های حرفی و عددی قایل نیست. اگر بخواهیم صحت شماره تلفن وارد شـــده را با دقت بیش‌تری مورد بررســـی قرار دهیم باید از شـــرط زیر استفاده کنیم:

[Tel] LIKE “[2-8][0-9][0-9][0-9][0-9][0-9][0-9][0-9]”

در این شرط مشخص کرده‌ایم که شماره وارد شده باید شامل ۸ رقم باشد که رقم اول فقط می‌تواند بین ۲ تا ۸ و سایر رقم‌ها بین ۰ تا ۹ باشند.

ماکرونویســـی مبحث پیشرفته‌ای است که برای تسلط بر آن باید به کتاب‌های نوشته شده در این زمینه یا راهنمای نرم‌افزار اکســـس مراجعه کنید. تســـلط بر این ابزار می‌تواند شما را به یک کاربر حرفه‌ای اکسس تبدیل کند.

What is a macro?

A macro is a tool that allows you to automate tasks and add functionality to your forms, reports, and controls. For example, if you add a command button to a form, you associate the button's OnClick event to a macro, and the macro contains the commands that you want the button to perform each time it is clicked.

In Access, it is helpful to think of macros as a simplified programming language that you write by building a list of actions. (action: The basic building block of a macro; a self-contained instruction that can be combined with other actions to automate tasks. This is sometimes called a command in other macro languages.) to perform. When you build a macro, you select each action from a drop-down list and then fill in the required information for each action. Macros enable you to add functionality to forms, reports, and controls without writing code in a Visual Basic for Applications (VBA) (Visual Basic for Applications) VBA is a macro-language version of Microsoft Visual Basic that is used to program Microsoft Windows-based applications and is included with several Microsoft programs.)

For example, suppose that you want to start a report directly from one of your data entry forms. You can add a button to your form and then create a macro that starts the report. The macro can either be a standalone macro (a separate object in the database), which is then bound to the OnClick event of the button, or the macro can be embedded directly into the OnClick event of the button itself — a new feature in Office Access 2007. Either way, when you click

the button, the macro runs and starts the report.

Understand macros

The term macro is often used to refer to standalone macro objects (that is, the objects that you see under Macros in the Navigation Pane), but in reality, one macro object can contain multiple macros. In that case, it is referred to as a macro group. A macro group is displayed in the Navigation Pane as a single macro object, but a macro group actually contains more than one macro. Of course, it is possible to create each macro in a separate macro object, but often it makes sense to group several related macros into a single macro object. The name in the Macro Name column identifies each macro.

A macro consists of individual macro actions. Most actions require one or more arguments. In addition, you can assign names to each macro in a macro group, and you can add conditions to control how each action is run.

۱. متن بالا را مطالعه کرده و در مورد آن توضیح دهید.

۲. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. موارد کاربرد ماکروها را نام ببرید.

۲. تفاوت میان ماکروهای تعبیه شده با ماکروهای مستقل در چیست؟

۳. ماکروی مستقلی بنویسید تا شماره فاکتورهایی را نشان دهد که در آن‌ها حداقل یک عنوان کتاب با تخفیف بیش‌تر از 40% فروخته شده باشد.

۴. فرمی حاوی یک دکمه و یک کادر متنی برای ورود نام یک شهر ایجاد کنید. سپس ماکرویی تعبیه شده روی دکمه بسازید تا با کلیک شدن دکمه، نام مشتریان ساکن در شهر وارد شده لیست شود.

فصل ششم

فصل ششم : محافظت از پایگاه داده

محافظت از فایـــل پایگاه داده به دلیل ارزش داده‌هایی که در خود ذخیره می‌کند، اهمیت فوق‌العاده‌ای دارد. مشـــکلات نرم‌افزاری و ســـخت‌افزاری متنوعی وجود دارد که ممکن است به این فایل آسیب رسانیده و همه یا بخشـــی از داده‌های موجود را غیرقابل دســـترس نماید. علاوه بر این ممکن است افرادی که مجوز مشـــاهده یا تغییـــر داده‌های درون پایگاه داده را ندارند، به فایل دسترســـی پیدا کـــرده و و اطلاعات آن را دســـت‌کاری یا از آن‌ها سوءاستفاده کنند. از این رو، آشـــنایی با اصول محافظت از پایگاه داده برای کاربران اکسس کاملاً ضروری است.

۱-۶ فشرده‌سازی و ترمیم پایگاه داده

با افزایش رکوردها و اشـــیاء موجود در پایگاه داده، ممکن است اشکالاتی در کار فایل اکسس بروز کند؛ حتی گاهی اوقات اتفاق ســـاده‌ای مثل قطع برق در حین باز بودن پایگاه داده باعث می‌شـــود بخش‌هایی از داده‌ها و اشـــیاء درون آن در اجرای بعدی با مشـــکلاتی مواجه شـــوند. از این رو برنامه اکسس در هنگام باز کردن فایل پایگاه داده، عملیاتی را برای حصول اطمینان از سلامت فایل به اجرا درمی‌آورد و در صورت وجود مشکل، برای رفع آن تلاش می‌کند؛ هرچند همیشه قادر به رفع مشکل نیست.

حتی اگر هیچ اشـــکال مهمی در پایگاه داده بروز نکرده باشـــد، اســـتفاده معمول از فایل و درج و حذف رکوردها بر ساختار داخلی فایل اکسس تأثیر می‌گذارد و هم نظم فیزیکی را که در ابتدای ایجاد فایل وجود داشت به هم می‌ریزد و هم فضاهای بلااستفاده (حافظه هرز) درون فایل ایجاد می‌کند. از این رو گاهی اوقات به نظر می‌رسد کارایی و سرعت کار پایگاه داده به نحو قابل ملاحظه‌ای کاهش یافته است؛ مسأله‌ای که اغلب از افزایش بیهوده حجم پایگاه داده ناشی می‌شود.

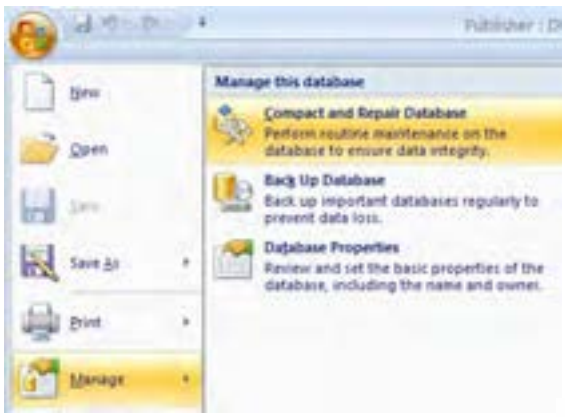
همه این مسایل باعث شده، طراحان اکسس قابلیتی به نام فشرده‌سازی و ترمیم[1] را در نرم‌افزار تعبیه کنند تا کاربران را برای غلبه بر خرابی‌ها یا کاهش کارایی یاری کند.

۱. با کلیک روی دکمه آفیس[2]، منوی برنامه را باز کنید.

۲. اشاره‌گر را روی گزینه Manage ببرید تا دستورات موجود در آن ظاهر شوند.

۳. روی عبارت Compact and Repair DataBase کلیک کنید.

۴. عملیات فشرده‌سازی و ترمیم در عرض چند ثانیه انجام می‌شود و ضمن رفع مشکلات احتمالی در جداول، فرم‌ها و گزارش‌ها، حجم فایل اکسس نیز به نحو چشم‌گیری کاهش می‌یابد.



نکته

توصیه می‌شود در بازه‌های زمانی منظم، این عملیات را روی فایل اجرا کنید.

۲-۶ پشتیبان‌گیری از پایگاه داده

پشتیبان‌گیری منظم از داده‌ها، یکی از مهم‌ترین وظایف کاربران رایانه محسوب می‌شود؛ چراکه تهدیدات نرم‌افزاری و سخت‌افزاری فراوانی در کمین اطلاعات موجود روی هارددیسک است و ممکن است یک اشتباه سهوی یا مشکل نرم‌افزاری یا سخت‌افزاری باعث از دست رفتن اطلاعات ارزشمند شما شود.

ایجاد نسخه‌های پشتیبان از فایل‌ها برای همه اطلاعاتِ حساس امری کاملاً ضروری است و اهمیت آن برای فایل‌های پایگاه داده بسیار بیش‌تر است؛ چراکه از دست دادن یک فایل اکسس ممکن است به از بین

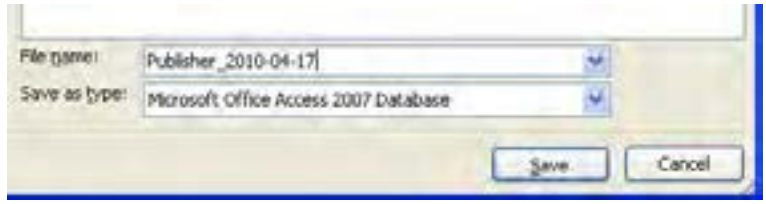
1 . Compact and Repair
2 . Office Button

رفتن اطلاعات حیاتی یک شرکت یا سازمان منجر شود. روش پشتیبان‌گیری از پایگاه داده در اکسس بسیار ساده و سریع است و توصیه می‌شود برای پایگاه داده‌هایی که تراکنش زیادی دارند، به صورت روزانه انجام شود.

۱. روی دکمه آفیس کلیک و از منوی Manage روی دستور Back Up Database کلیک کنید.

۲. پنجره ذخیره‌سازی ظاهر می‌شود و در بخش File name آن، نام پایگاه داده همراه با تاریخ جاری درج می‌شود.

۳. محل ذخیره‌سازی را تعیین و روی دکمه Save کلیک کنید.



توصیه می‌شود فایل ایجاد شده را روی CD رایت کنید یا در صورت استفاده از شبکه، آن را به یکی دیگر از رایانه‌های موجود منتقل نمایید. در هر صورت، نگه‌داری نسخه‌های پشتیبان روی همان رایانه روش مناسبی نیست چون در صورت بروز مشکل سخت‌افزاری (مثل آسیب‌دیدن هارددیسک بر اثر نفوذ ویروس، ضربه یا آتش‌سوزی) فایل پشتیبان هم از بین خواهد رفت.

۳-۶ بررسی کارایی[1] اجزاء پایگاه داده

میزان کارایی، یکی از مسایلی است که در طول عمر پایگاه داده باید به صورت مداوم مورد بررسی قرار گیرد و در صورت نیاز، اصلاحاتی بر روی ساختار جداول ایجاد گردد. در اکسس قابلیتی وجود دارد که کارایی پایگاه داده را بررسی نموده و به طراحان و کاربران در خصوص راه‌های بهبود عمل‌کرد مشاوره می‌دهد.

۱. برنامه اکسس را باز و در صفحه شروع آن، گزینه Sample را انتخاب کنید.

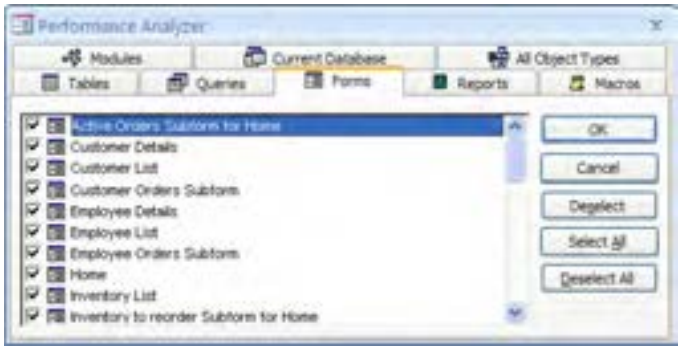
۲. روی عبارت Northwind کلیک و در ستون سمت راست، محل ذخیره‌سازی فایل پایگاه داده را تعیین نمایید.

1 . Performance

۳. با کلیک روی دکمه Create فایل Northwind ۲۰۰۷.accdb ســـاخته می‌شود که مثالی از پایگاه داده یک شرکت فعال در زمینه تجارت است.

۴. در منوی Database Tools و قاب Analyze روی دستور Analyze Performance کلیک کنید تا پنجره انتخاب اجزاء ظاهر شود.

۵. با کلیک روی دکمه Select All همه اجزاء پایگاه داده را انتخاب و روی دکمه OK کلیک کنید.



۶. پس از چند لحظه، پنجره تحلیل پایگاه داده نمایان می‌شـــود. نتایج به دســـت آمده از تحلیل در قالب ردیف‌های زیر نشان داده می‌شوند :

الف) توصیه (Recommendation)

ب) پیشنهاد (Suggestion)

ج) نظر (Idea)

د) اصلاح شده (Fixed)

بـــا کلیک روی هر نتیجـــه، جزییات آن در پایین پنجره ظاهر می‌شـــود. برای مـــواردی نظیر توصیه‌ها می‌توانید روی دکمه Optimize کلیک کنید تا تغییرات موردنیاز در پایگاه داده اعمال شـــود و آیکن نتیجه به حالت اصلاح شده تغییر شکل دهد.

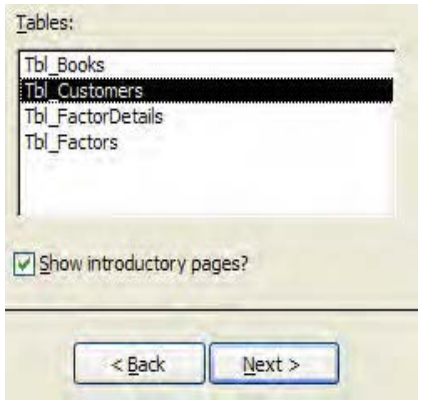
۶-۴ بررسی جدول‌ها

در ابتدای این کتاب با روش‌های علمی موجود برای طراحی یک پایگاه داده اســـتاندارد و کارآمد آشـــنا شدید. جالب است بدانید در اکسس قابلیتی پیش‌بینی شده که با بررسی ساختار جداول و داده‌های ذخیره شده درون آن‌ها می‌تواند برای بهبود طراحی، پیشنهادهایی را ارایه کند.

۱. پایگاه داده‌ای را که در طول کتاب روی آن کار کرده‌ایم باز کنید.

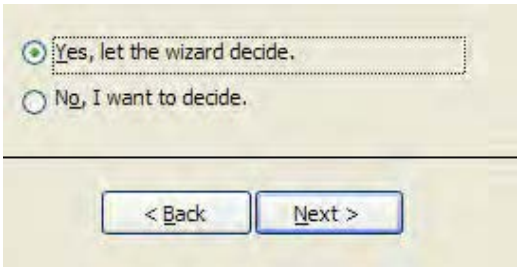
۲. در زبانه Database Tools و قاب Analyze روی دکمه Analyze Table کلیک کنید.

۳. ویزاردی ظاهر شد و نحوه تحلیل جداول را در قالب یک مثال توضیح می‌دهد. روی دکمه Next کلیک نمایید.

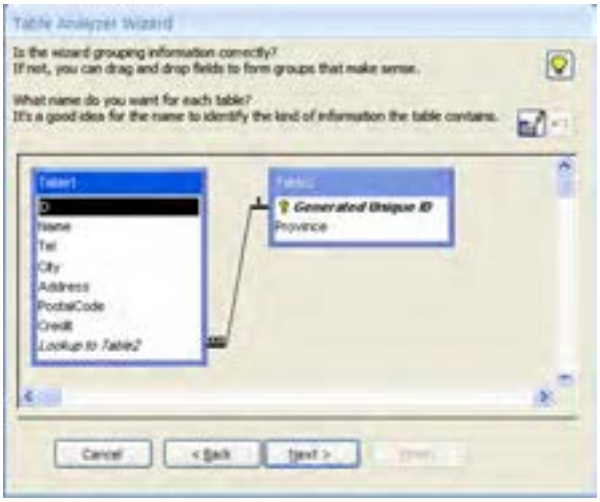


۴. پس از کلیک روی دومین دکمه Next، به پنجره انتخاب جدول می‌رســـید. جدول مشـــتریان (Tbl_Customers) را انتخاب و روی دکمه Next کلیک کنید.

۵. در پنجـــره بعد گزینه اول را انتخاب و روی دکمه Next کلیک کنید. این گزینه، تصمیم‌گیری در مورد ایجاد تغییرات در ساختار جدول را به ویزارد واگذار می‌کند.



۶. ویزارد با بررسی جدول مشتریان به این نتیجه رسیده که نام استان‌ها در فیلد Province تکرار می‌شود. بنابراین پیشـــنهاد می‌کند یک جدول مستقل ایجاد و نام استان‌ها با شناسه منحصربفرد در آن ذخیره شود. سپس در جدول مشتریان، به جای نام استان، شناسه متناظر درج شود.



۷. با دوبار کلیک روی نام جداول، اسامی موردنظر را به آن‌ها نسبت داده و روی دکمه Next کلیک کنید.

۸. در پنجره بعد باید کلید اصلی جداول فاقد کلید اصلی را تعیین و روی دکمه Next کلیک نمایید.

۹. یکی از مراحل مهم در کار با این ویزارد، اصلاح اشـــتباهات تایپی در ورود داده‌های فیلدی اســـت که از جدول اصلی جدا می‌شـــود. برای مثال اگر دو مشـــتری از استان «چهارمحال و بختیاری» داشته باشید و

اشتباهاً در یکی از رکوردها اشتباه تایپی وجود داشته باشد، پنجره زیر ظاهر می‌شود که ابتدا باید با انتخاب گزینه Leave as is حالت صحیح را انتخاب و سپس سایر رکوردها را با انتخاب نام استان از لیست بازشونده، تصحیح کنید.

۱۰. با کلیک روی دکمه Next و نهایتاً Finish، دو جدول جدید ساخته می‌شود. جدول قبلی هم با اضافه شدن عبارت OLD_ در انتهای نام هم‌چنان در دسترس است.

۵-۶ مستندسازی[1] پایگاه داده

هنگامی که نرم‌افزاری را برای استفاده در یک سازمان تولید می‌کنید، سفارش‌دهنده برنامه از شما می‌خواهد تا مستندات آن را نیز همراه با فایل‌های اجرایی تحویل دهید تا اگر در آینده، فرد دیگری بخواهد تغییراتی را در نرم‌افزار ایجاد کند، روش طراحی و پیاده‌سازی به صورت کامل در اختیار او قرار داده شود؛ این کار باعث جلوگیری از سردرگمی توسعه‌دهنده نرم‌افزار خواهد شد.

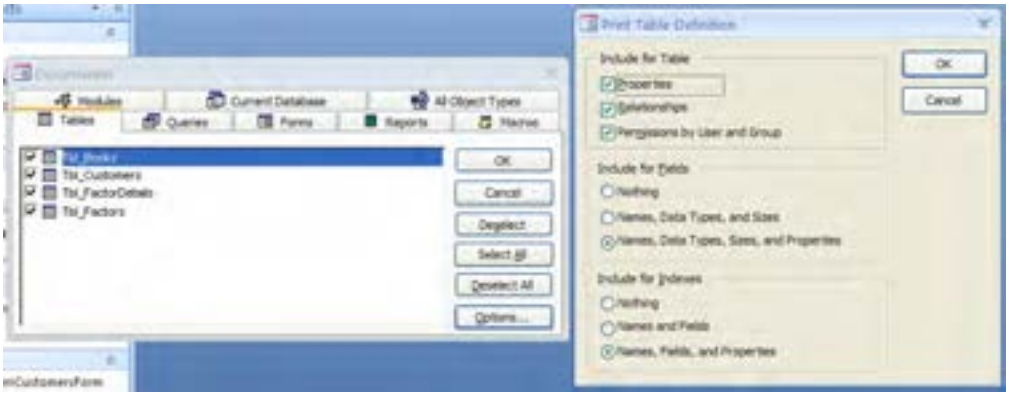
در اکسس برای مستندسازی پایگاه داده، یعنی مشخص کردن ویژگی‌های جداول، فرم‌ها، گزارش‌ها و ... و نیز روابط میان جداول روش سریعی پیش‌بینی شده است.

۱. در زبانه Database Tools و قاب Analyze روی دکمه Database Documenter کلیک کنید.

۲. پنجره انتخاب عناصر پایگاه داده ظاهر می‌شود. هر عنصر زبانه‌ای دارد که در آن می‌توانید موارد دلخواه را تیک بزنید. علاوه بر این می‌توانید با کلیک روی دکمه Select All همه عناصر موجود در پایگاه داده را انتخاب نمایید.

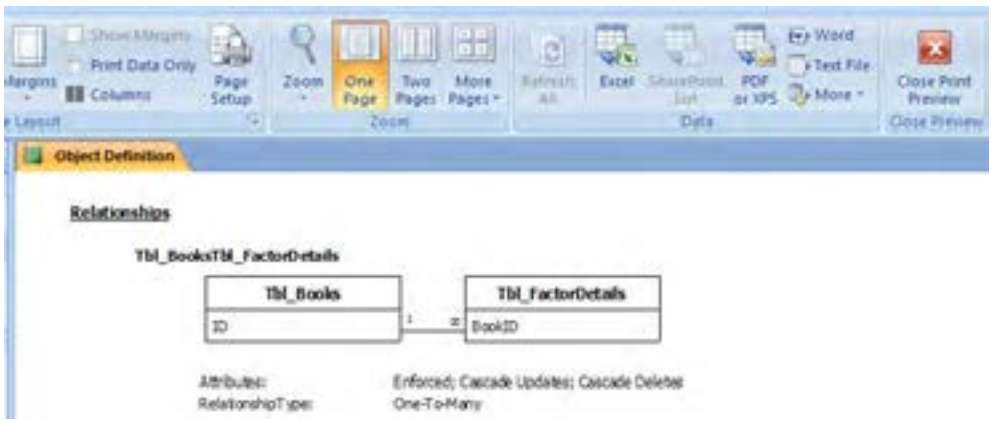
۳. روی دکمه Options کلیک کنید. پنجره‌ای ظاهر می‌شود که در آن امکان تنظیم جزییات وجود دارد. برای مثال با تیک زدن گزینه Relationships، روابط میان جداول نیز در مستندات درج خواهد شد.

1 . Documentation



۴. در پنجره Documenter روی دکمه OK کلیک کنید تا مستندات پایگاه داده تولید شود. با کلیک روی دکمه‌های پیمایش، همه صفحات آن را ببینید.

۵. فایل تولید شـــده را نمی‌توانید در پایگاه داده ذخیـــره کنید اما امکان چاپ یا تبدیل آن به فرمت‌هایی مثل docx (برای اســـتفاده در Word)، pdf (برای نمایش در Adobe Acrobat) و نیز html (جهت مشاهده در مرورگر اینترنتی) وجود دارد. Export کردن فایل تولید شـــده از طریق کلیک روی دکمه‌های موجود در قاب Data انجام می‌پذیرد.



۶. با کلیک روی دکمه قرمزرنگ Close Print Preview، مستندات ساخته شده را ببندید.

۶-۶ قرار دادن رمز عبور[1] برای پایگاه داده

تأمین امنیت پایگاه داده و جلوگیری از دسترسی‌های غیرمجاز به داده‌های درون آن از مهم‌ترین مباحث تولید برنامه‌های کاربردی محسوب می‌شود. در اکسس برای تحقق این مسأله دو راه‌حل پیش‌بینی شده است :

الف) قرار دادن رمزعبور روی فایل اکسس که بیش‌تر در محیط‌های تک‌کاربره کاربرد دارد.

ب) تعیین سطوح دسترسی به اطلاعات در محیط‌هایی که چند کاربر به صورت هم‌زمان از پایگاه داده استفاده می‌کنند.

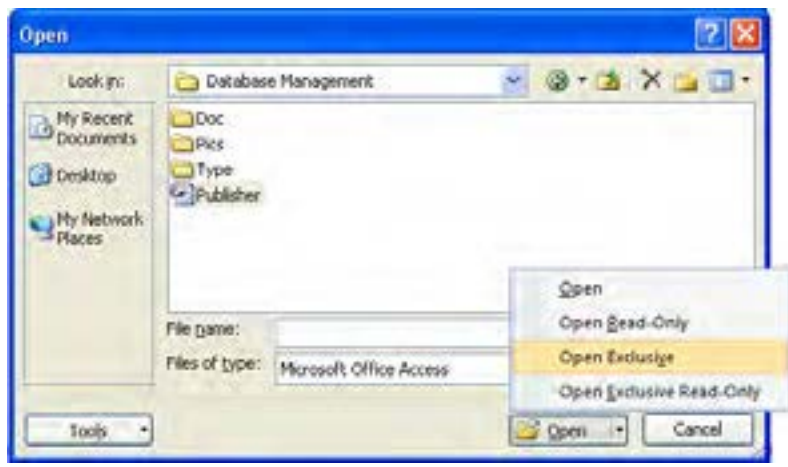
در این کتاب شما با روش اول آشنا خواهید شد. در این روش، کاربر برای باز کردن پایگاه داده باید رمز عبور را وارد کند.

۱. برنامه اکسس را باز کنید.

۲. در منوی آفیس روی دستور Open کلیک کنید یا کلیدهای میان‌بر Ctrl+O را فشار دهید.

۳. با محل ذخیره‌سازی پایگاه داده رفته و روی آن کلیک کنید.

۴. با کلیک روی مثلث کنار دکمه Open، منوی آن را باز و دستور Open Exclusive را انتخاب کنید تا پایگاه داده برای کاربرد انحصاری (در این‌جا تنظیم رمز عبور) باز شود.



۵. در زبانه و قاب Database Tools روی دکمه Encrypt with Password کلیک کنید.

---

1 . Password

۶. رمز عبور و تأیید آن را وارد نموده و دکمه OK را بزنید. از این به بعد برای باز کردن فایل پایگاه داده، وارد کردن رمزعبور ضروری است.

نکته

در انتخـــاب رمز عبور، حتی‌الامکان از عبارات طولانی، حاوی عدد و نویســـه‌های غیرحرفی (مثل *، @، ! و ...) اســـتفاده نمایید. از آن‌جا که اغلب نرم‌افزارهای تولید شده برای کشف رمز عبور فایل‌های اکسس در شناسایی نویسه‌های یونیکد[1] (مثل حروف فارسی) ناتوان هستند، ورود عبارات فارسی باعث بالا رفتن امنیت پایگاه داده می‌شود.

۷. وقتی یک پایگاه داده دارای رمز عبور باشـــد، در زبانـــه و قاب Database Tools، دکمه Encrypt with Password به Decrypt Database تبدیل می‌شود. با کلیک روی این دکمه و وارد کردن رمز فعلی می‌توانید رمز عبور پایگاه داده را حذف کنید. برای برداشـــتن رمز عبور هم باز کردن فایل در حالت کاربرد انحصاری[2] ضروری است.

۷-۶ تنظیمات شروع به کار[3] پایگاه داده

هنگامی که یک فایل اکســـس را باز می‌کنید، به صورت پیش‌فرض صفحه‌ای ظاهر می‌شـــود که حاوی منوهای برنامه و نوار پیمایش اســـت؛ یعنی پایـــگاه داده در وضعیتی قرار دارد که بیش‌تر مخصوص طراحی است. اما اگر بخواهید پایگاه داده را تبدیل به یک برنامه کاربردی نموده و در اختیار کاربرانی قرار دهید که فقط وظیفه ورود اطلاعات و گزارش‌گیری را برعهده دارند باید شـــروع به کارِ پایگاه داده را به‌گونه‌ای تنظیم کنید که در زمان باز شدن فایل، یک فرم ظاهر شود یا ماکرویی اجرا گردد.

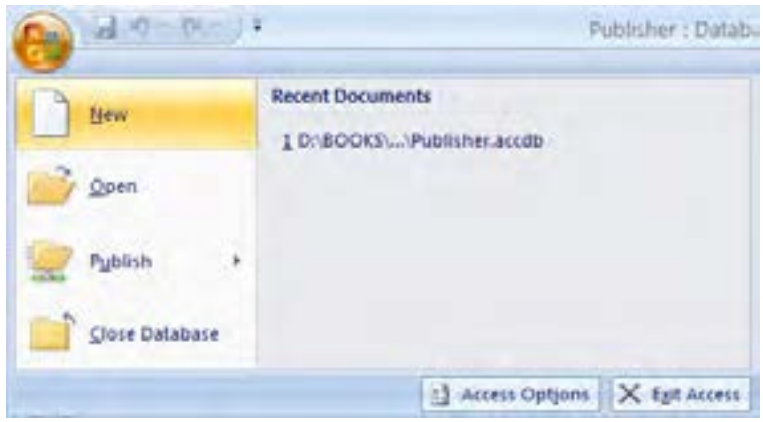
---

1 . Unicode
2 . Exclusive
3 . Startup

روش معمول برای شـــروع به کار پایگاه داده، نمایش یک فرم با چند گزینه قابل انتخاب اســـت تا کاربر دســـتورات موردنیاز را اجرا و به صفحات دلخواه منتقل شود. این فرم، «صفحه انتخاب»[1] نامیده می‌شود و با مراجعه به کتاب‌های مرجع اکسس می‌توانید نحوه ایجاد آن را یاد بگیرید.

در ایـــن تمرین قصد داریم یکی از فرم‌های برنامه را که در بخش‌هـــای قبل تولید کردیم به عنوان فرم شروع به کار پایگاه داده تنظیم کنیم.

۱. در انتهای منوی آفیس روی دکمه Access Options کلیک کنید تا پنجره تنظیمات برنامه ظاهر شود.



۲. در ستون سمت چپ روی گزینه Current Database کلیک کنید.

۳. در کادر متنی Application Title عبارتی را وارد کنید که نشان دهنده کارکرد برنامه باشد.

1 . Switch Board

۴. منوی Display Form را باز و فرمی را که می‌خواهید در زمان راه‌اندازی پایگاه داده نشـــان داده شـــود انتخاب کنید.

۵. در ایـــن صفحه گزینه‌هـــای متعددی برای تعیین نحوه نمایش اجزاء صفحـــه وجود دارد. برای مثال با برداشـــتن تیک گزینـــه Navigation Pane می‌توانید نوار پیمایش را از دیـــد کاربر پنهان کنید تا در حالت نمایش اولیه، قادر به تغییر طراحی جداول و فرم‌ها نباشد.

۶. روی دکمه OK کلیک کنید. پیغامی ظاهر می‌شـــود مبنی بر این که برای مشاهده تغییرات، برنامه را بسته و فایل را مجدداً باز کنید.

۷. از این پس هنگام باز کردن فایل، فرم Frm_Menu نشـــان داده می‌شـــود و عبارت وارد شـــده در کادر Application Title جایگزین عبارت Access 2007 در بالای پنجره برنامه می‌شود.

Use a database password to encrypt an Office Access 2007 database

The encryption tool in Office Access 2007 combines and improves on two older tools - encoding and database passwords. When you use a database password to encrypt a database, you make all data unreadable by other tools, and you force users to enter a password to use the database. The encryption applied in Office Access 2007 uses a stronger algorithm than was used in earlier versions of Access.

Encrypt by using a database password

1. Open the database that you want to encrypt in Exclusive mode.

    1. Click the Microsoft Office Button , and then click Open.
    2. In the Open dialog box, browse to the file that you want to open, and then select the file.
    3. Click the arrow next to the Open button, and then click Open Exclusive.

2. On the the Database Tools tab, in the Database Tools group, click Encrypt with Password.

The Set Database Password dialog box appears.

3. Type your password in the Password box, and then type it again in the Verify field.

NOTE

Use strong passwords that combine uppercase and lowercase letters, numbers, and symbols. Weak passwords don't mix these elements. Strong password: Y6dh!et5. Weak password: House27. Passwords should be 8 or more characters in length. A pass phrase that uses 14 or more characters is better.

It is critical that you remember your password. If you forget your password, Microsoft cannot retrieve it. Store the passwords that you write down in a secure place away from the information that they help protect.

4. Click OK.

۱. متن بالا را مطالعه کرده و در مورد آن توضیح دهید.

۲. یک عبور قوی با یک رمز عبور ضعیف چه تفاوتی دارد؟

۳. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. فشرده‌سازی و ترمیم پایگاه داده چه ضرورتی دارد؟

۲. روش ایجاد نسخه پشتیبان از فایل اکسس را توضیح دهید. آیا روش دیگری برای انجام این کار وجود دارد؟

۳. یک پایگاه داده برای ذخیره‌سازی اطلاعات دانش‌آموزان، دروس و دبیران مدرسه ایجاد و در آن داده‌های آزمایشی وارد کنید. سپس عملیات بررسی جداول، مستندسازی و قرار دادن رمز عبور را روی آن اجرا کنید.

۴. رمزعبورهای زیر را از نظر قدرت، رتبه‌بندی کنید.

الف) alireza89

ب) alireza

ج) ali*reza89

د) reza

فصل هفتم

فصل هفتم : آشنایی با SQL Server

برای افرادی که به دنیای برنامه‌نویسی و به خصوص تولید نرم‌افزارهای مبتنی بر پایگاه داده وارد می‌شوند، یادگیری کار با Access اولین گام محسوب می‌شود و نیازهای فنی آن‌ها را تا حد زیادی برطرف می‌کند اما Access برای تولید نرم‌افزارهای بزرگ، محدودیت‌های زیادی از لحاظ کارایی و امنیت دارد. این‌جاست که برنامه‌نویسان باید به سراغ سیستم‌های قدرتمندتری مثل SQL Server، اوراکل[1] ، MySQL، DB2 و ... بروند و کار با حداقل یکی از آن‌ها را یاد بگیرند.

در میان این سیستم‌های مدیریت پایگاه داده، SQL Server جایگاه ویژه‌ای دارد چون نسبت خوبی میان سه شاخص سادگی، کارایی و قیمت آن برقرار است. در ادامه این کتاب نحوه کار با جدیدترین نسخه این سیستم مدیریت پایگاه داده را فرا می‌گیرید.

---

1 .ORACLE

۱-۷ SQL Server و کاربرد آن

SQL Server یکی از رایج‌ترین سیستم‌های مدیریت پایگاه داده رابطه‌ای است که برای نخستین بار در سال ۱۹۸۹ میلادی توسط شرکت‌های سای‌بِیس[1] و مایکروسافت عرضه شد و در طول یک دهه و به خصوص زمانی که شرکت مایکروسافت به صورت مستقل مسؤولیت توسعه آن را بر عهده گرفت توانست در دنیای نرم‌افزار جایگاه قابل قبولی پیدا کند. این سیستم بر اساس مدل سرویس‌دهنده – سرویس‌گیرنده[2] بنا شده که در آن درخواست سرویس‌گیرنده برای سرویس‌دهنده ارسال می‌شود و پس از انجام پردازش‌های موردنیاز روی داده‌ها، نتیجه برای سرویس‌گیرنده فرستاده می‌شود.

به این ترتیب شما می‌توانید برنامه‌های تحت ویندوز یا تحت وب خود را به این سیستم متصل نموده و مدیریت ذخیره و بازیابی داده‌ها را برعهده SQL Server بگذارید. قابلیت‌های گسترده این سیستم در پشتیبانی از مدل رابطه‌ای، تعریف سطوح امنیتی و خودکارسازی فعالیت‌ها باعث شده SQL Server به قلب تپنده طیف وسیعی از نرم‌افزارهای کاربردی و بزرگ در سطح دنیا تبدیل شود.

در یک بررسی اجمالی می‌توان موارد زیر را به عنوان ویژگی‌های برجسته این سیستم مدیریت پایگاه داده برشمرد :

الف) توانایی کار با پایگاه داده‌های حجیم تا حد صدها گیگابایت

ب) امکان دسترسی هم‌زمان هزاران کاربر به پایگاه داده

ج) سازگاری بالا با سیستم‌عامل‌ها و محیط‌های برنامه‌نویسی گوناگون

د) برخورداری از سطوح امنیتی بالا

ه) ایجاد محیط یکپارچه برای مدیریت پایگاه داده در نسخه‌های اخیر

SQL Server در سیر پیشرفت خود راهی طولانی را پیموده و تغییرات زیادی در ظاهر، امکانات و کارایی آن ایجاد شده است. شاید بتوان نسخه ۲۰۰۰ این نرم‌افزار را از لحاظ تکمیل سرویس‌های ذخیره و بازیابی اطلاعات یک نقطه کامل محسوب نمود. شرکت مایکروسافت در نسخه‌های بعدی یعنی ۲۰۰۵ و ۲۰۰۸ کوشید علاوه بر ایجاد محیط یکپارچه برای مدیریت پایگاه داده، سرویس‌های جدیدی مانند، گزارش‌گیری[3] ،تحلیل[4] ، اطلاع‌رسانی[5] و ... را در نرم‌افزار خود تعبیه کند و برای حفظ فاصله خود با رقبای سرسختی چون اوراکل یک گام بلند دیگر بردارد.

---

1 .Sybase

2 .Server-Client

3 . Reporting Service

4 . Analyze Service

5 . Notification Service

در این کتاب شـــما با قابلیت‌ها و امکانات SQL Server 2008 آشـــنا خواهید شـــد. این نسخه در چهار ویرایش[1] به بازار عرضه شده که اغلب آن‌ها بر روی Windiow Server قابل نصب هستند. اما اگر می‌خواهید این نرم‌افزار را روی سیســـتم‌عامل‌های غیر ســـرور مانند Windows XP نصب کنید باید به ســـراغ ویرایش Developer بروید.

SQL Server مانند بســـیاری دیگر از سیســـتم‌های مدیریت پایگاه داده رابطـــه‌ای از زبان پرس‌وجوی ســـاختاریافته (SQL) اســـتفاده می‌کند. SQL که در اوایل دهه ۸۰ میلادی در آزمایشگاه‌های شرکت IBM متولد شد زبانی برای تعریف ساختارهای داده‌ای (مثل جداول) و نیز دست‌کاری داده‌ها (مانند عملیات درج) محسوب می‌شود و در سیستم‌های مدیریت پایگاه داده مختلف، با تفاوت‌های جزیی قابل پیاده‌سازی است.

هرچند اغلب سیســـتم‌های مدیریت پایگاه داده حاوی ابزارهایی برای تولید خودکار پرس‌وجو هستند اما تسلط بر نگارش دستی دستورات SQL به شما برای ایجاد پایگاه داده‌های کارا و خوانا کمک زیادی خواهد کرد.

۲-۷ آشنایی با Management Studio

هنگامی که نرم‌افزار SQL Server 2008را به صورت کامل روی سیستم خود نصب می‌کنید، سرویس‌ها و امکانات متعددی در اختیار شـــما قرار می‌گیرد. در این میان، Management Studio به عنوان یک واسط گرافیکی[2]، امکان مدیریت آسان اجزاء و داده‌های موجود در پایگاه داده را فراهم می‌آورد.

اهمیت این واســـط گرافیکی به خصوص برای کاربران مبتدی زمانی مشـــخص‌تر می‌شود که بدانند در برخی از سیســـتم‌های مدیریت پایگاه داده امروزی، محیط یکپارچه‌ای برای مدیریت همه سرویس‌ها و اجزا پایگاه داده وجود ندارد و هنوز هم بســـیاری از کارها بـــا وارد کردن کد در خط فرمان یا انجام می‌گیرد؛ در حالی که Management Studio محیط ساده اما کارآمدی را برای مدیریت پایگاه داده فراهم آورده است.

۱. در منوی شروع ویندوز، بر روی عبارت SQL Server 2008 و سپس گزینه SQL Server Management Studio کلیک کنید.

۲. پنجره ورود به محیط برنامه ظاهر می‌شـــود. اما قبل از کلیک روی دکمه Connect باید مشـــخص کنید قصد دارید به کدام ســـرور و چه سرویسی از آن متصل شـــوید. در منوی Server type، سرویس‌های نصب شـــده روی رایانه قابل انتخاب هستند. برای ورود به محیط ساخت و مدیریت پایگاه داده، گزینه Database Engine را انتخاب کنید.

---

1 . Edition

2 .GUI (Graphic User Interface)

۳. در لیســـت بازشونده Server name می‌توانید نام رایانه‌ای را که مشغول کار با آن هستید انتخاب کنید یا اگر متصل به یک شبکه هستید، با انتخاب گزینه <Browse for more…> به سراغ سایر رایانه‌هایی بروید که SQL Server روی آن‌ها نصب شده است.

۴. در لیســـت Authentication که روش اعتبارسنجی و مجوز ورود کاربر را تعیین می‌کند دو گزینه زیر را می‌بینید:

- **Windows Authentication :** اعتبارســـنجی توسط خود ویندوز انجام می‌شود؛ یعنی فرض بر این است که کاربرِ وارد شده به ویندوز، اجازه دسترسی به Management Studio را هم دارد.
- **SQL Server Authentication :** بـــا انتخاب این گزینه، کادرهای User name و Password فعال می‌شود و کاربر باید نام کاربری و رمزعبور معتبری را برای ورود به محیط برنامه وارد کند.

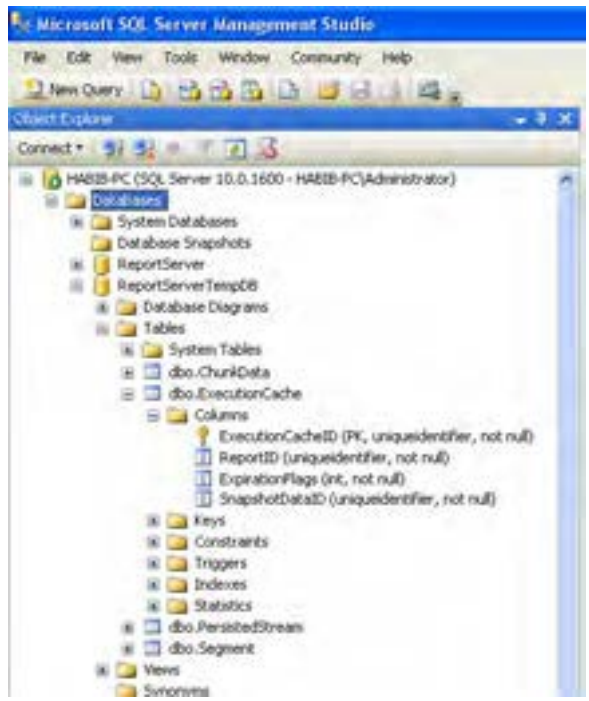
تعیین روش اعتبارسنجی کاربران در هنگام نصب برنامه قابل تنظیم است.

۵. روی دکمه Connect کلیک کنید تا اتصال به سرور برقرار و پنجره Management Studio ظاهر شود. این پنجره دارای یک منو و نوار دسترســـی ســـریع در بالا و یک قاب مرورگر اشـــیاء[1] در سمت چپ صفحه

1 . Object Explorer

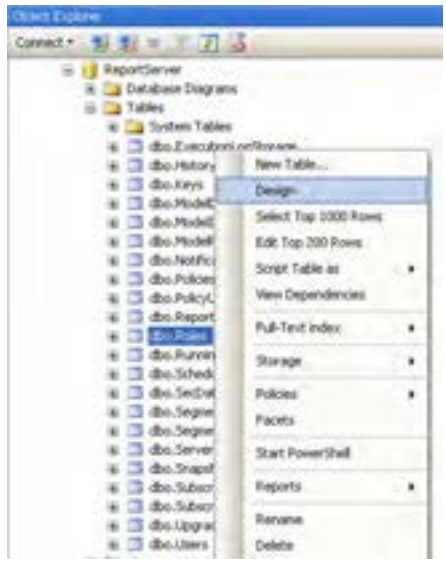
اســـت که در آن اجزاء موجود در پایگاه داده درون یک نمودار درختی مرتب شـــده‌اند و با باز کردن پوشه‌ها و راســـت‌کلیک روی عناصر درون آن‌ها می‌توانید دستورات موردنظر را اجرا کنید؛ این قاب جزو مهم‌ترین و کاربردی‌تری بخش‌های نرم‌افزار است.

۶. پوشـــه Databases را باز کنید. در پوشه System Databases، پایگاه داده‌های سیستمی مثل master، model و ... قرار دارند که غالباً خودِ SQL Server با آن‌ها سر و کار دارد. پایگاه داده‌هایی هم که توسط کاربر ساخته می‌شود با آیکن یک استوانه زردرنگ در پوشه Databases قرار می‌گیرند.



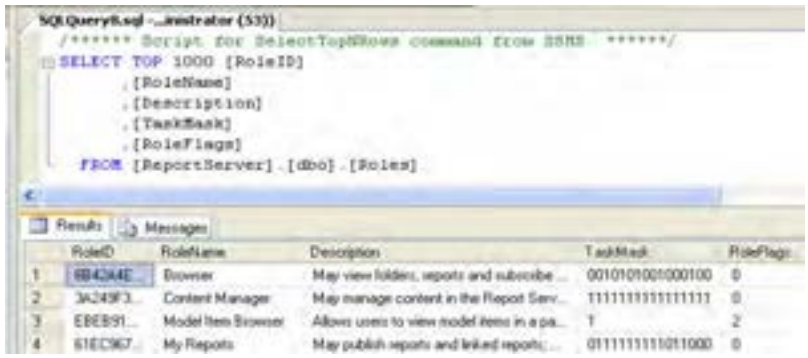
۷. پایگاه داده ReportServerTempDB را که در صورت نصب سرویس گزارش‌گیری، در لیست پایگاه‌های داده دیده می‌شـــود باز کنید. همان‌طور که در تصویر بالا می‌بینید همه اجزاء پایگاه داده از جداول و نماها گرفته تا ســـتون‌ها و کلیدها در قالب یک ســـاختار درختی مرتب شده‌اند و با باز کردن پوشه‌ها می‌توانید به همه بخش‌ها و امکانات پایگاه داده دسترسی داشته باشید.

۸. به ســـراغ پایـــگاه داده ReportServer بروید. پوشـــه Tables را باز نموده و روی جـــدول dbo.roles راســـت‌کلیک کنید. منویی حاوی چندین گزینه ظاهر می‌شـــود که در ادامه با کار تعدادی از آن‌ها آشـــنا خواهید شد.



Design : جدول را در نمای طراحی باز می‌کند و امکان تغییر نام و نوع فیلدها فراهم می‌آید.

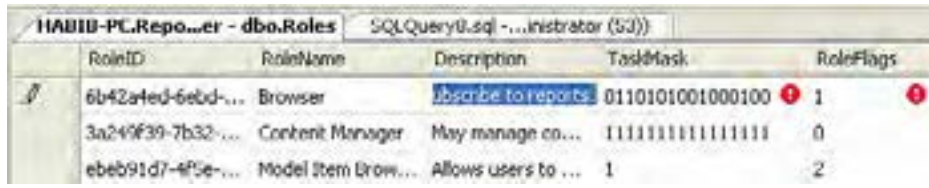
Select Top 1000 Rows : این دستور، فضای سمت راست مرورگر اشیاء را به دو بخش تقسیم می‌کند. در قسمت بالا، پرس‌وجوی موردنیاز برای استخراج هزار ردیف اول جدول نوشته می‌شود و در قسمت پایین، رکوردهای اســـتخراج شده درون یک شبکه[1] نشان داده می‌شوند. دقت داشته باشید که در این حالت امکان تغییر رکوردها وجود ندارد.



Edit Top 200 Rows : دویســـت ردیف اول جدول را درون یک شـــبکه نمایش می‌دهد و امکان تغییر

1 . Grid

اطلاعـــات رکورد را فراهم می‌آورد. وقتی فیلدی را تغییر می‌دهید یک علامت تعجب قرمز رنگ در کنار آن ظاهر می‌شود؛ به این معنی که مقدار فیلد تغییر کرده اما هنوز تغییرات در پایگاه داده ثبت نشده است.



با زدن کلید Enter یا کلیک کردن درون یکی از ردیف‌های دیگر (به شرط همخوانی مقادیر وارد شده با ضوابط جدول) تغییرات ثبت می‌شوند. فشار دادن کلید Esc هم باعث لغو تغییرات خواهد شد.

Rename : امکان تغییر نام جداول را فراهم می‌آورد.

Delete : چنان‌چه محدودیتی برای حذف جدول وجود نداشته باشد، آن را حذف می‌کند.

همان‌طور که مشـــاهده کردید با اجرای هر دستور، زبانه جدیدی در بخش میانی پنجره Management Studio ظاهر می‌شود که با کلیک روی علامت ضربدرِ انتهای قاب می‌توانید آن زبانه را ببندید.

در SQL Server برای اجرای اغلب دستورات، دو روش وجود دارد که اولی استفاده از منوها و ویزاردهای موجود در نرم‌افزار و دیگری کدنویســـی در محیط Management Studio است. هر چند ممکن است روش کدنویســـی در نگاه اول پیچیده و زمان‌بر به نظر برســـد، اما تســـلط بر آن برای برنامه‌نویسان حرفه‌ای کاملاً ضروری است زیرا :

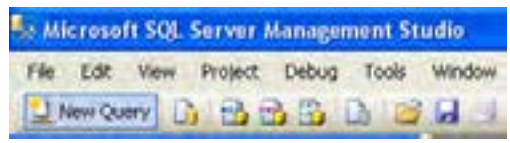
الف) امکان دسترسی به همه گزینه‌ها و تنظیمات اجرای دستور را فراهم می‌آورد.

ب) دستورات را با سرعت بیش‌تری اجرا می‌کند.

ج) برای تولید رویه‌های ذخیره شده[1]، تراکنش‌ها[2] و برخی دیگر از اجزاء پایگاه داده تنها راه موجود است.

برای نوشتن کدهای SQL در محیط Management Studio به روش زیر عمل کنید:

۱. در نوار منوهای برنامه روی دکمه New Query کلیک نمایید.



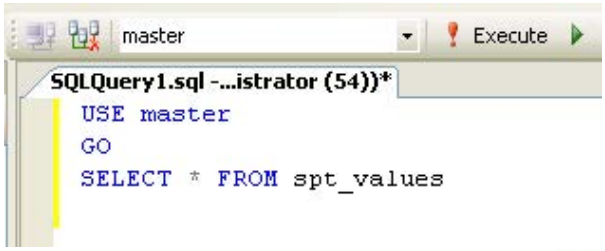
1 . Stored Procedures

2 . Transactions

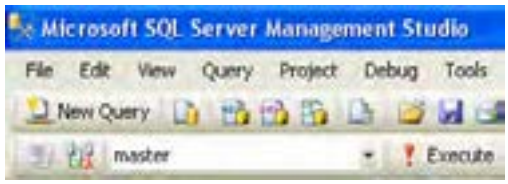
۲. زبانه جدیدی به قاب میانی برنامه اضافه می‌شود که حاوی صفحه سفیدی برای کدنویسی است.

۳. کد زیر را در صفحه وارد کنید.

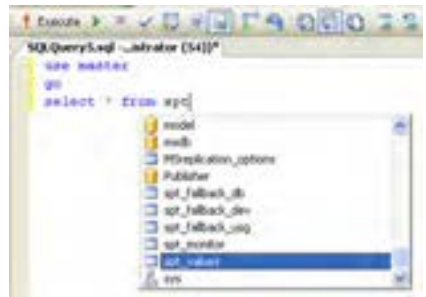
```
USE master
GO
SELECT * FROM spt_values
```



- عبـــارت USE پایـــگاه داده‌ای را که می‌خواهید با آن کار کنید مشـــخص می‌کنـــد. در این تمرین می‌خواهیم با پایگاه داده master که همراه با Management Studio نصب می‌شود کار کنیم. اگر در نوار تعیین پایگاه داده فعلی، پایگاه داده موردنظر انتخاب شـــده باشد دیگر نیازی به استفاده از دستور USE نیست.

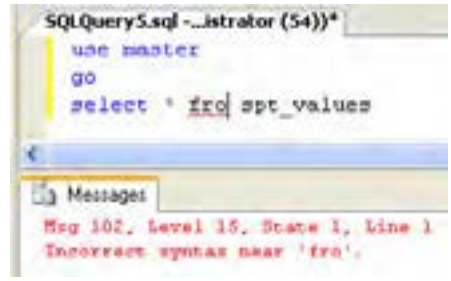


- عبارت go دستورات SQL را از هم جدا می‌کند و صرفاً برای افزایش خوانایی کدها استفاده می‌شود.
- عبارت SELECT همه رکوردهای جدول spt_values را برمی‌گرداند.
- کلمه‌های کلیدی با رنگ آبی نشان داده می‌شوند و این مسأله به شما کمک می‌کند سریع‌تر متوجه اشتباهات تایپی خود شوید.
- هنگام تایپ نام پایگاه داده یا جدول، به صورت خودکار لیستی ظاهر می‌شود و موارد مشابه با حروف تایپ شـــده را نشان می‌دهد. به این لیست IntelliSense گفته می‌شود و کمک زیادی به تایپ سریع دســـتورات می‌کند. با فشـــار دادن کلید Enter می‌توانید عبارت انتخاب شده را وارد صفحه کنید. یک روش سریع برای ظاهر کردن این لیست، نگه‌داشتن کلید Ctrl و فشار دادن دکمه Space است.



۴. روی علامت قرمز رنگ تعجب (دکمه Execute) کلیک کنید تا کد اجرا و نتیجه در قاب نتایج نشـــان داده شـــود. این قاب دو زبانه دارد که رکوردهای استخراج شـــده در زبانه Results و درون یک شبکه نشان داده می‌شـــوند. قاب Messages هم حاوی پیغام‌های سیستم در مورد اجرای دستور است و در مورد دستور SELECT، تعداد رکوردهای بازیابی شده را نشان می‌دهد.

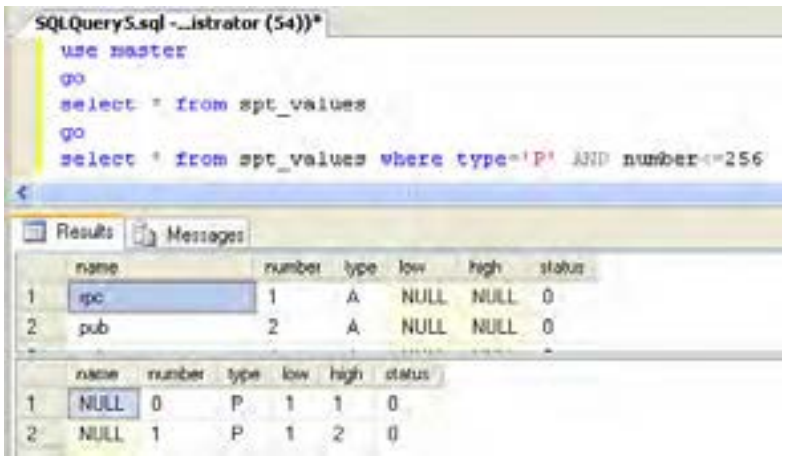
۵. یکی از کلمات کلیدی را طوری تغییر دهید که از نظر املایی اشتباه شود. این بار با کلیک روی دکمه اجرا، در زبانه Messages یک پیغام قرمزرنگ ظاهر می‌شود که اعلام می‌کند دستور وارد شده نادرست است.



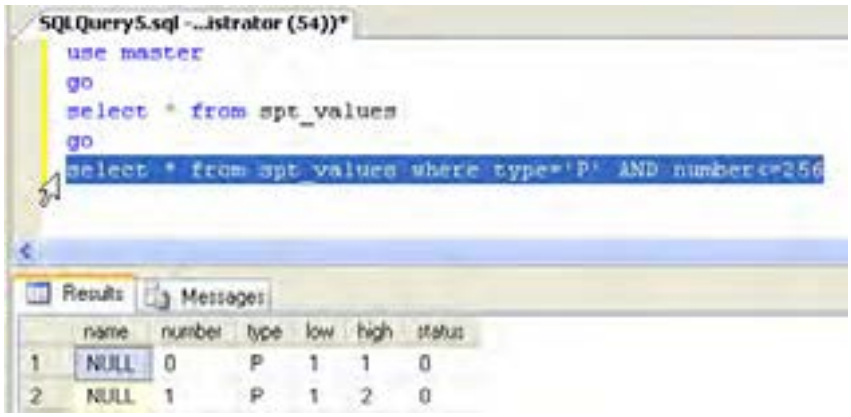
۶. عبارت را تصحیح نموده و کد زیر را در ادامه پرس‌وجوی قبلی تایپ کنید.

```
GO
SELECT * FROM spt_values WHERE type='P' AND number<=256
```

۷. روی دکمه اجرا کلیک کنید. در قاب نتایج، دو شبکه نشان داده می‌شود که حاوی نتایج دو پرس‌وجوی موجود در صفحه است. واژه type آبی‌رنگ می‌شود و نشان می‌دهد جزو کلمات رزرو شده SQL است. توصیه می‌شـــود نام فیلدهایی را که جزو کلمات رزرو شـــده هستند درون کروشه قرار دهید تا مشکلی برای اجرای پرس‌وجو ایجاد نشود.

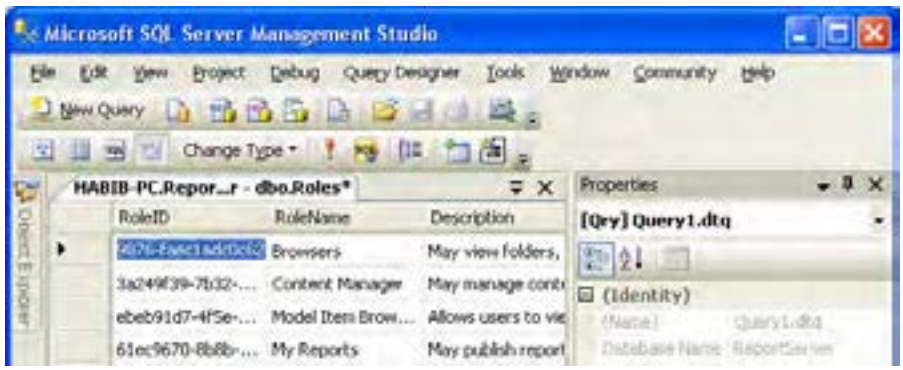


۸. اشـــاره‌گر را به ابتدای یکـــی از خطوط (بین نوار زردرنگ و ابتدای پرس‌وجـــو) ببرید و کلیک کنید تا دســـتور انتخاب شود. حالا روی دکمه اجرا کلیک کنید. همان طور که می‌بینید از میان دستورهای موجود در صفحه، فقط کد انتخاب شده اجرا می‌شود. این روش راهی سریع برای اجرای یک یا چند دستور از میان کدهای نوشته شده است.



۹. در نوار منوی برنامه روی آیکن دیسکت کلیک کنید. محل ذخیره‌سازی و نام پرس‌وجو را وارد کرده و روی دکمه Save کلیک کنید. کد نوشـــته شـــده با پسوند sql ذخیره می‌شود و می‌توانید در ادامه کار از آن استفاده کنید.

به صورت پیش‌فرض در ســـمت چپ پنجره، قاب مرورگر اشیاء و در سمت راست قاب خصوصیات[1] قرار دارد. اگر سنجاق[2] بالای این قاب‌ها به حالت عمودی باشد، قاب در جای خود ثابت می‌شود اما اگر با کلیک روی این سنجاق آن را به حالت افقی در بیاورید، قاب فقط در حالتی که اشاره‌گر روی آن قرار داشته باشد نمایان است و پس از کنار رفتن اشاره‌گر تبدیل به یک دکمه در نوار کناری صفحه خواهد شد.



با قرار گرفتن اشاره‌گر روی دکمه، قاب مجدداً ظاهر می‌شود.

1 . Properties Pane
2 . Pin

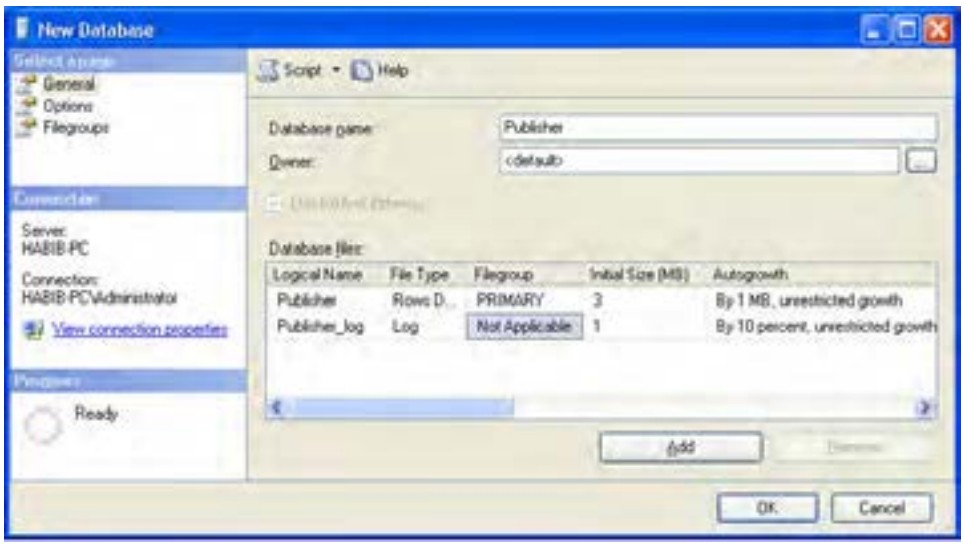
۷-۳ ایجاد پایگاه داده و جداول

اگر اصول طراحی پایگاه داده را فرا گرفته باشـــید، پیاده‌ســـازی آن که شامل ایجاد پایگاه داده، ساخت جداول و برقراری روابط میان آن‌ها می‌شـــود کار چندان دشـــواری نیست. در این بخش با نحوه ایجاد پایگاه داده و جداول به دو روش متفاوت آشنا خواهید شد.

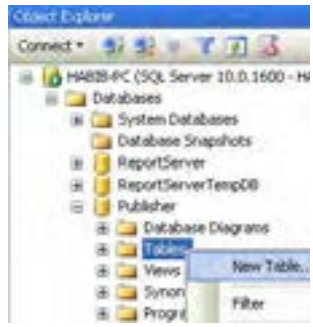
۷-۳-۱ استفاده از واسط گرافیکی

۱. در مرورگر اشیاء روی پوشه Databases راست‌کلیک نموده و گزینه New Database را انتخاب کنید.

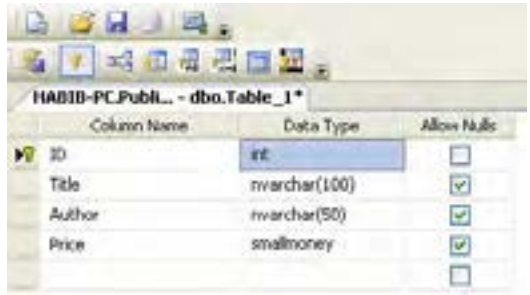
۲. در پنجـــره‌ای که ظاهر می‌شـــود درون کادر Database name نام موردنظـــر را وارد و روی دکمه OK کلیک کنید تا پایگاه داده خام ساخته شود.



۳. پوشـــه پایگاه داده ساخته شده را باز و روی پوشه Tables راســـت‌کلیک کنید. با انتخاب گزینه New Table، پنجره میانی برنامه در حالت طراحی جدول قرار می‌گیرد.



۴. نام ستون، نوع داده‌ای و امکان تهی بودن را برای فیلد اول تعیین و این کار را برای همه فیلدها تکرار کنید. تنظیم کلید اصلی هم با انتخاب فیلد و کلیک روی آیکن کلید انجام می‌شود.



۵. با کلیک روی آیکن  و تعیین نام برای جدول، آن را ذخیره کنید.

۶. جدول به پوشه Tables پایگاه داده اضافه می‌شود و در ساختار درختی می‌توانید کلیدها، خصوصیات ستون‌ها و سایر اجزاء جدول را یک‌جا مشاهده کنید.

با استفاده از واسط گرافیکی، جدول tbl_Customers را ایجاد کنید.

۲-۳-۷ استفاده از کدهای SQL

۱. در نوار منوی برنامه روی دکمه New Query کلیک کنید.

۲. کد زیر را در قاب پرس‌وجو وارد نمایید.

```
use Publisher
go
CREATE TABLE Tbl_Factors
(
 ID int IDENTITY(1,1)PRIMARY KEY NOT NULL,
 CustomerID int NOT NULL,
 [Date] char(10) NULL
)
```

بررسی کد :

دستور CREATE TABLE برای ایجاد جدول به کار می‌رود و بعد از آن باید نام جدول قید شود.

نام فیلدها، نوع داده‌ای آن‌ها و پذیرش یا عدم پذیرش مقادیر تهی باید برای همه فیلدها مشخص شود. مشخصات هر فیلد با علامت کاما از فیلد بعدی جدا می‌شود و مجموعه این مشخصات باید درون یک جفت پرانتز قرار گیرند.

عبارت IDENTITY(a,b) مشخص می‌کند که این فیلد از نوع شناسه است و به صورت خودکار تولید می‌شود. ضمناً شناسه‌های این فیلد از عدد a شروع شده و هربار معادل b اضافه می‌شوند.

عبارت PRIMARY KEY تعیین می‌کند که این فیلد کلید اصلی است.

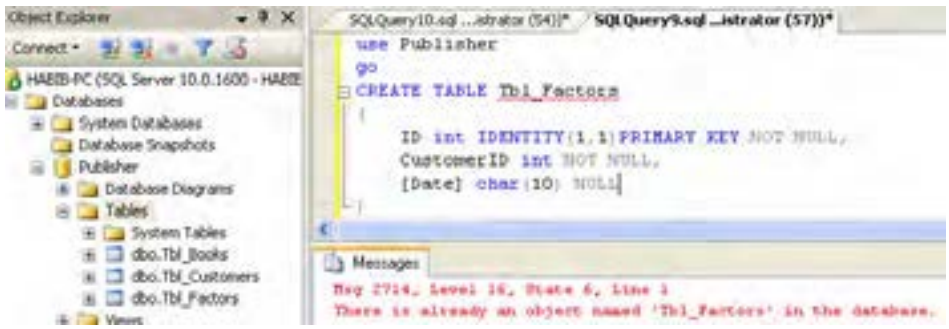
اگر نام فیلد، یکی از کلمات رزرو شده SQL است باید نام فیلد را درون کروشه قرار دهید؛ مانند Date در کد بالا.

۳. روی دکمه اجرا کلیک کنید تا جدول ساخته شود. ظاهر شدن پیغام Command(s) completed

successfully نشان دهنده اجرای موفقیت‌آمیز دستور است.

۴. روی پوشـــه Table از پایگاه داده Publisher راســـت‌کلیک نموده و گزینه Refresh را انتخاب کنید تا جدول ساخته شده ظاهر شود.

۵. مجدداً کد را اجرا کنید.



۶. یک پیغام خطا ظاهر می‌شود و اعلام می‌کند که جدولی با همین نام در پایگاه داده وجود دارد.

۷. قصد داریم در جدولی که ساخته‌ایم تغییری ایجاد کنیم تا یک فیلد جدید به آن اضافه شود. کد درج شده در صفحه‌ی بعد را در قاب پرس‌وجو وارد و اجرا کنید.

```
use Publisher
go
ALTER TABLE Tbl_Factors
ADD NewColumn int NULL
```

بررسی کد :

دستور ALTER برای ایجاد تغییر در اجزاء پایگاه داده به کار می‌رود.

برای اضافه کردن یک فیلد جدید، باید پس از دســـتور ADD نام ســـتون و سپس نوع داده‌ای آن را قید کنید.

۸. برای حذف یکی از فیلدهای جدول باید از قطعه کد زیر استفاده کنید.

```
use Publisher
go
ALTER TABLE Tbl_Factors
DROP COLUMN NewColumn
```

بررسی کد :

- دستور DROP برای حذف اجزاء پایگاه داده به کار می‌رود.
- برای حذف کردن یک فیلد ، باید پس از دستور DROP COLUMN نام ستون را درج کنید.

۳-۳-۷ انواع داده‌ای در SQL Server

همان‌طور که مشـــاهده کردید هنگام اســـتفاده از واسط گرافیکی، روش ســـاخت جداول SQL Server تفاوت چندانی با اکسس ندارد، اما انواع داده‌ای در محیط این دو نرم‌افزار تاحدی متفاوت از هم هستند. در ادامه، انواع داده‌ای در SQL Server 2008 و کاربرد هر یک را بررسی می‌کنیم.

char : برای ذخیره‌ســـازی رشته‌هایی با طول ثابت به کار می‌رود. برای مثال char(6) عبارتی ۶ حرفی را ذخیره می‌کند و اگر رشـــته 'aaa' را در آن ذخیره کنید، سمت راست رشته با فواصل خالی پر می‌شود؛ به صورت : ' aaa'

nchar : شـــبیه به char است با این تفاوت که نویســـه‌های یونیکد را هم ذخیره می‌کند. بنابراین برای ذخیره‌سازی عبارات فارسی با طول ثابت باید از این نوع داده‌ای استفاده کنید. n از کلمه national به معنی «ملی» گرفته شـــده و نشان دهنده امکان ذخیره‌سازی زبان ملیت‌های مختلف است. دقت داشته باشید که nchar در مقایســـه با char حجم بیش‌تری را اشـــغال می‌کند بنابراین اگر رشته فقط حاوی حروف انگلیسی است، استفاده از انواع داده‌ای غیر national ارجحیت دارد.

varchar : مانند char برای ذخیره‌ســـازی عبارات متنی کاربرد دارد با این تفاوت که طول فیلد متغیر است؛ یعنی اگر فیلدی را به صورت varchar(50)تعریف کنید حداکثر ۵۰ نویسه را ذخیره می‌کند با این‌حال عبارتی مثل 'aaa' فقط ۳ نویسه را اشغال خواهد کرد.

nvarchar : کارکرد آن شبیه به varchar است اما نویسه‌های یونیکد را هم ذخیره می‌کند.

حداکثر طول قابل تعریف برای این چهار نوع داده‌ای، ۸۰۰۰ نویسه است و برای طول‌های بیش‌تر از این باید به جای عدد، از عبارت MAX استفاده کنید مانند nvarchar(MAX)

text و **ntext :** بـــرای نگه‌داری داده‌های متنی با طول بیش‌تر از ۸۰۰۰ نویســـه کاربرد دارد اما با توجه به این که در نســـخه‌های بعدی SQL Server حذف خواهد شـــد، توصیه می‌شـــود از varchar(MAX) و

nvarchar(MAX) استفاده نمایید.

اعداد صحیح : چهار نوع داده‌ای زیر برای نگه‌داری مقادیر عددی صحیح وجود دارد.

نوع داده‌ای	محدوده
tinyint	۰ تا ۲۵۵
smallint	-۳۲۷۶۸ تا ۳۲۷۶۷
int	-۲۱۴۷۴۸۳۶۴۸ تا ۲۱۴۷۴۸۳۶۴۷
bigint	$-۲^{۶۳}$ تا $۲^{۶۳}-۱$

real و **float :** این دو نوع داده‌ای، اعداد اعشاری را ذخیره می‌کنند.

bit : مقدار ۰ را برای False و ۱ را برای True ذخیره می‌کند.

time، date، smalldatetime، datetime، datetime2: هریــک محدوده‌ای از تاریخ و زمان را با دقت خاصی به صورتی که در جدول زیر نشان داده شده است، پوشش می‌دهند.

نوع داده‌ای	قالب	محدوده	دقت
time	hh:mm:ss[.nnnnnnn]	00:00:00.0000000 تا 23:59:59.9999999	100 nanoseconds
date	YYYY-MM-DD	0001-01-01 تا 9999-12-31	1 day
smalldatetime	YYYY-MM-DD hh:mm:ss	1900-01-01 تا 2079-06-06	1 minute
datetime	YYYY-MM-DD hh:mm:ss[.nnn]	1753-01-01 تا 9999-12-31	0.00333 second
datetime2	YYYY-MM-DD hh:mm:ss[.nnnnnnn]	0001-01-01 00:00:00.0000000 تا 23:59:59.9999999 9999-12-31	100 nanoseconds

binary و **varbinary :** داده‌های دودویی[1] (صفر و یک) را ذخیره می‌کند.

1 . Binary

نـــوع داده‌ای image هـــم برای داده‌های باینری طراحی شـــده و می‌توان تصاویـــر را درون آن ذخیره کرد. اما این نوع داده‌ای در نســـخه‌های بعدی SQL Server حذف خواهد شـــد بنابراین توصیه می‌شـــود از varbinary(MAX) استفاده کنید.

money و **smallmoney :** برای نگه‌داری مقادیر پولی مناسب است.

uniqueidentifier : یکـــی از انواع داده‌ای اســـت کـــه برای ذخیره یک شناســـه منحصربفرد جهانی[1] به کار می‌رود. این شناسه با توجه به مشخصات نرم‌افزاری و سخت‌افزاری سیستم تولید می‌شود و برای همه رایانه‌های موجود در شبکه منحصربفرد است مانند :

bf5a2c6f-b0d9-4837-9375-f1f443e23f4b

۷-۴ یکپارچگی داده‌ها

یکپارچگی داده‌ها[2] که در برخی منابع از آن به عنوان «جامعیت پایگاه داده» یاد می‌شـــود یعنی این‌که داده‌ها در هر لحظه از حیات پایگاه داده، ســـازگاری و اعتبار داشته باشند. این مسأله مستلزم آن است که با اعمال برخی محدودیت‌ها روی پایگاه داده، کیفیت داده‌های درون آن را تضمین کنیم.

برای مثال اگر در جدول اطلاعات دانشجویان، دانشجویی با شماره ۸۹۰۱۲۱۴۰ وجود داشته باشد پایگاه داده نباید اجازه دهد دانشـــجوی دیگری با همین شـــماره در جدول ثبت شود. همچنین اگر یکی از نمرات دانشـــجو به اشتباه بالاتر از ۲۰ ثبت شده باشـــد می‌توانیم نتیجه‌گیری کنیم که پایگاه داده در اعتبارسنجی داده‌های ورودی دچار ضعف است و نمی‌تواند صحت داده‌های جداول را تضمین کند.

۷-۴-۱ انواع یکپارچگی

در هنگام طراحی جداول باید مشـــخص کنید که اولاً برای هر ســـتون چه مقادیری معتبر اســـت و ثانیاً چگونـــه می‌خواهید پایگاه داده را ملزم به حفظ اعتبار و یکپارچگی داده‌ها کنید. حفظ یکپارچگی داده‌ها در چهار زمینه زیر قابل بررسی است :

الف) یکپارچگی موجودیتی[3]: پایگاه داده را ملزم می‌کند هر ردیف از جدول، یک فیلد (یا مجموعه‌ای از

---

1 . Global Unique Identifier (GUI)

2 . Data Integrity

3 . Entity Entegrity

فیلدهای) منحصربفرد داشته باشد. مثلاً شماره کتاب در جدول کتاب‌ها منحصربفرد باشد.

ب) یکپارچگی دامنه‌ای[1]: با تعیین نوع داده‌ای برای فیلد و محدود کردن قالب یا محدوده مقادیر باعث تضمین داده‌های وارد شـــده به جدول می‌شـــود. برای مثال پایگاه داده را از پذیرش مقدار منفی برای تعداد کتاب‌های فروخته شده منع می‌کند.

ج) یکپارچگی ارجاعی[2]: وقتی جدولی از طریـــق کلید خارجی با جدول اصلی رابطه دارد، نباید در آن ردیفی وجود داشـــته باشد که کلید آن در جدول اصلی، حذف یا عوض شده باشد. برای مثال اگر در جدول جزییات فاکتور، فروشی با شماره فاکتور ۱۰۱ وجود دارد، در جدول فاکتورها، این شماره موجود باشد.

جدول فاکتورها

شماره فاکتور	شماره مشتری	تاریخ فروش
۱۰۰	۱۲۰۱	۱۳۸۹/۱۱/۰۱
۱۰۱	۱۲۰۵	۱۳۸۹/۰۵/۱۱

جدول جزییات فاکتور

شماره فاکتور	شماره کتاب	تعداد	تخفیف
۱۰۰	۵۰۱۰	۳۵	۲۵
۱۰۰	۵۰۱۲	۱۰	۳۰
۱۰۱	۵۰۰۹	۳۶	۳۰
۱۰۰	۵۰۱۳	۲۰	۲۰
۱۰۱	۵۰۲۰	۸۰	۰

د) یکپارچگی تعریفِ کاربر[3]: طراح با تعریف برخـــی قواعد در محیط پایگاه داده، یکپارچگی داده‌ها را تعریف می‌کند؛ مثلاً اگر قیمت کتابی کم‌تر از ۲۰۰۰ تومان باشد، تخفیف بیش‌تر از 20% برای آن غیرمجاز است.

۲-۴-۷ روش‌های پیاده‌سازی

در SQL Server بـــرای حفظ یکپارچگـــی پایگاه داده و تضمین صحت و ســـازگاری داده‌ها، روش‌های متنوعی پیش‌بینی شده است که از جمله آن‌ها می‌توان به موارد زیر اشاره کرد:

الف) محدودیت‌ها[4]: با اســـتفاده از محدودیت‌ها (که گاهی قید هم گفته می‌شـــوند) می‌توانید سیستم مدیریـــت پایگاه داده را ملزم کنید در ورود داده‌ها، محدودیت‌های موردنظر شـــما را اعمال نماید. در SQL

1 . Domain Integrity
2 . Referential Integrity
3 . User Defined
4 . Constraints

Server محدودیت‌های زیر قابل پیاده‌سازی هستند :

NOT NULL : هنگام تعریف جدول مشخص می‌کند که فیلد حتماً باید حاوی مقدار باشد.

کلید اصلی : فیلد یا فیلدهایی که تبدیل به کلید اصلی شـــده‌اند نباید مقادیر تکراری یا NULL داشته باشند.

کلید خارجی : ارتباط میان رکوردهای یک جدول را با رکورد متناظر در جدول اصلی حفظ می‌کند.

محدودیـــت یکتایی[1]: محدودیت یکتایـــی مانند کلید اصلی عمل می‌کند یعنـــی از تکرار یک مقدار در فیلدهای مشـــابه از دو رکورد جلوگیری می‌کند و می‌تواند مورد ارجاع کلید خارجی واقع شـــود، اما مقدار NULL را هم می‌پذیرد و می‌توان بیش از یک محدودیت یکتایی را روی یک جدول اعمال کرد.

محدودیت‌های وارسی[2]: مقدار یک ورودی را بررسی و در صورت مطابقت با محدودیت تعریف شده روی فیلد، به آن اجازه ذخیره سازی در پایگاه داده می‌دهد.

مثال : می‌خواهیم جدول جزییات فروش را با استفاده از کدهای SQL ایجاد کنیم. در این جدول:

الف) فیلدهای FactorID و BookID به صورت مشترک کلید اصلی هستند.

ب) فیلد FactorID با فیلد ID از جدول فاکتورها در ارتباط است.

ج) باید محدودیتی تعریف کنیم که درصد تخفیف بین صفر تا هفتاد درصد باشد.

جدول فوق به روش زیر قابل پیاده‌سازی است:

۱. در نوار منوی Management Studio روی دکمه New Query کلیک کنید.

۲. کد زیر را در قاب پرس‌وجو وارد و آن را اجرا کنید.

```
use Publisher
go
CREATE TABLE Tbl_FactorDetailss
(
  FactorID int NOT NULL,
```

1 . UNIQUE Constraints

2 . Check Constraints

```
    BookID int NOT NULL,
    Quantity smallint,
    Discount tinyint
    CONSTRAINT PK_Tbl_FactorDetails PRIMARY KEY (FactorID,BookID)
    CONSTRAINT FK_Tbl_FactorDetails_Tbl_Factors FOREIGN KEY (FactorID)
    REFERENCES Tbl_Factors(ID)
        ON DELETE CASCADE
        ON UPDATE CASCADE
    CONSTRAINT chk_discount CHECK (Discount BETWEEN 0 AND 70)
)
```

۳. روی پوشه Tables راست کلیک و گزینه Refresh را انتخاب نمایید.

۴. پوشه‌های Columns، Keys و Constraints را باز کنید. همان‌طور که می‌بینید محدودیت‌های موردنظر ایجاد شده‌اند.

بررسی کد :

برای ایجاد یک محدودیت ابتدا باید کلیدواژه CONSTRAINT ، سپس نام محدودیت و بعد نوع آن را قید کنید. در ادامه هم پارامترها یا کلیدواژه‌های خاص هر محدودیت درج می‌شود.

هنگام ایجاد کلید خارجی باید مشخص کنید که تغییر یا حذف مقدار فیلد در جدول اصلی، روی

جدول مرتبط چه تأثیری می‌گذارد. برای مثال درحالت آبشـــاری یا CASCASE با تغییر شماره فاکتور در جدول فاکتورها، شماره فاکتور در رکورهای مرتبط با این فاکتور در جدول جزییات فاکتور هم تغییر خواهد کـــرد؛ در غیر این‌صورت در جدول جزییات فاکتور، رکوردهایی خواهیم داشـــت که به هیچ فاکتوری وصل نیستند و این باعث از بین رفتن یکپارچگی داده‌ها خواهد شد.

به عنوان مثال فرض کنید، شماره فاکتور ۱۰۱ در جدول فاکتورها به ۱۰۱۰ تغییر پیدا کند اما این تغییر در جدول جزییات فاکتورها اعمال نشـــود. در این حالت، ارتباط میان رکوردهای فاکتور در جدول جزییات فاکتور، با رکورد فاکتور در جدول فاکتورها از بین می‌رود.

جدول فاکتورها

شماره فاکتور	شماره مشتری	تاریخ فروش
۱۰۰	۱۲۰۱	۱۳۸۹/۱۱/۰۱
۱۰۱۰	۱۲۰۵	۱۳۸۹/۰۵/۱۱

جدول جزییات فاکتور

شماره فاکتور	شماره کتاب	تعداد	تخفیف
۱۰۰	۵۰۱۰	۳۵	۲۵
۱۰۰	۵۰۱۲	۱۰	۳۰
۱۰۱	۵۰۰۹	۳۶	۳۰
۱۰۰	۵۰۱۳	۲۰	۲۰
۱۰۱	۵۰۲۰	۸۰	۰

ب) قواعد[1] : شـــما می‌توانید برای ســـازگاری داده‌ها، قواعدی را ایجاد و آن‌ها را به فیلدها نسبت دهید. انجام این کار توسط محدودیت‌های وارسی نیز امکان‌پذیر است که البته این کار مزیت‌هایی دارد چرا که :

- برخلاف محدودیت‌های وارسی، به هر فیلد فقط می‌توان یک قاعده نسبت داد.
- محدودیت‌های وارسی جزیی از ساختار جدول هستند اما قواعد جدای از جدول ذخیره می‌شوند.
- قواعد در نسخه‌های بعدی SQL Server وجود نخواهند داشت.

ج) **پیش‌فرض‌هـــا**[2] : فیلدها را می‌توان به گونه‌ای تنظیم کرد که هنـــگام درج رکورد جدید، اگر کاربر داده‌ای برای آن فیلد وارد نکند، مقدار پیش‌فرض در فیلد ثبت شود. برای تنظیم مقدار پیش‌فرض :

۱. جدول مشـــتریان را در حالت طراحی باز کنید. این کار با راست‌کلیک روی نام جدول و انتخاب گزینه Design انجام می‌شود.

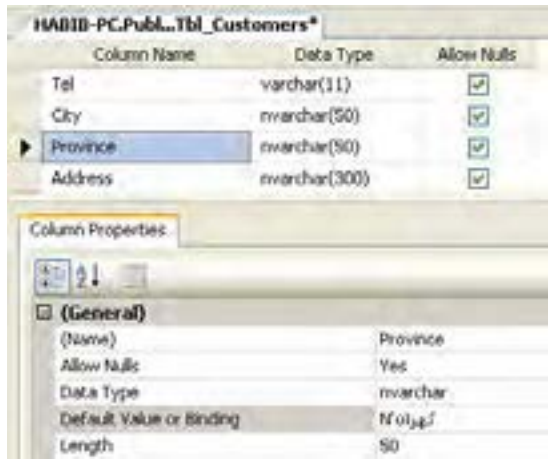
۲. درون فیلد Province کلیک نموده و در قاب Column Properties، روبروی خصوصیت Defaul Values

1 . Rule
2 . Defaults

Binding or نام یکی از استان‌ها را به فارسی درج کنید.

۳. یک حرف N به نشانه غیرانگلیسی بودن این عبارت جلوی کلمه وارد شده ظاهر می‌گردد.

۴. تغییرات را ذخیره کنید.



۵. از این پس چنان‌چه استانی را به عنوان محل سکونت مشتری وارد نکنید، به صورت پیش‌فرض مقدار وارد شده در فیلد ذخیره می‌شود.

د) **تریگرها**[1] : تریگر قطعه کدی است که در پایگاه داده اجرا می‌شود؛ عمل خاصی را انجام می‌دهد یا شرطی را بررسی می‌کند. فرض کنید می‌خواهید محدودیتی را روی پایگاه داده اعمال کنید تا کتاب‌هایی که کم‌تر از ۲۰۰۰ تومان قیمت دارند با تخفیف بیش‌تر از 20% فروخته نشوند. این کار با استفاده از محدودیت‌های وارسی امکان‌پذیر نیست چون قیمت کتاب‌ها در جدول کتاب‌ها و میزان تخفیف در جدول جزییات فروش ذخیره می‌شود و محدودیت وارسی را تنها می‌توان روی فیلدهای یک جدول اعمال کرد.

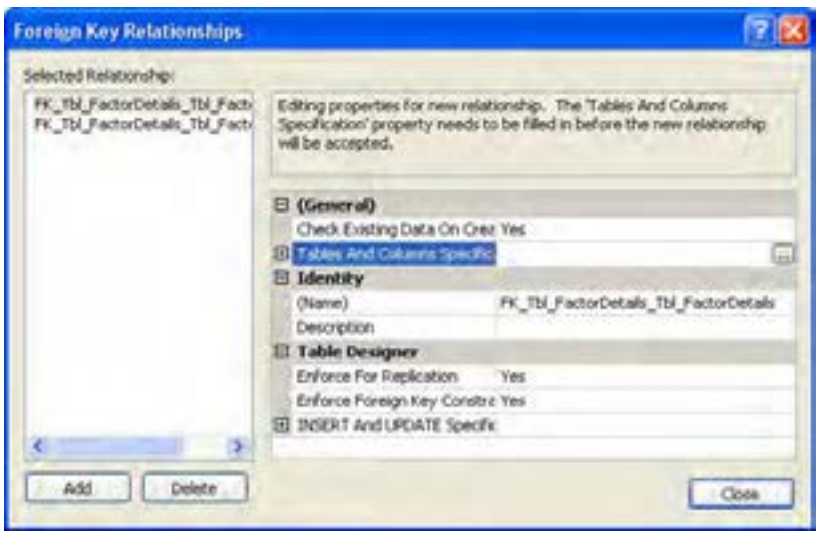
برای انجام این کار و نیز پیاده‌سازی محدودیت‌های وارسی پیچیده، استفاده از تریگرها بهترین انتخاب ممکن است.

در مثال این بخش، با استفاده از کدهای SQL جدول جزییات فروش را ایجاد و محدودیت‌هایی را روی آن ایجاد کردیم. حالا قصد داریم روش ایجاد یک محدودیت (از نوع کلید خارجی) را با استفاده از واسط گرافیکی توضیح دهیم. در این تمرین می‌خواهیم فیلد شماره کتاب را در جدول جزییات فاکتور به فیلد شماره کتاب در جدول کتاب‌ها مرتبط کنیم تا با تغییر احتمالی شماره کتاب در جدول کتاب‌ها، این شماره

1 . Trigger

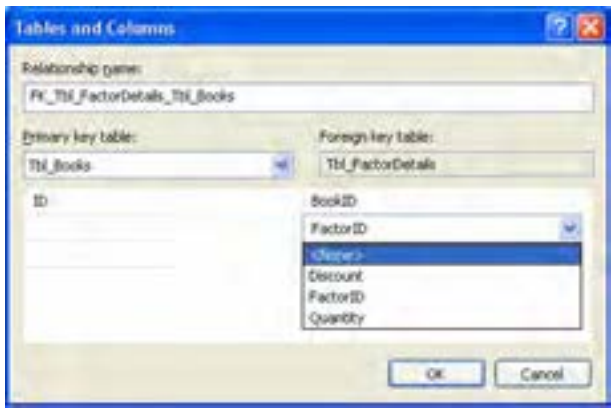
در جدول جزییات فروش هم تغییر کند. همچنین با روش ســـاخت نمودار پایگاه داده و ایجاد کلید خارجی روی آن آشنا خواهید شد.

۱. در جدول Tbl_FactorDetails روی پوشـــه Keys راست‌کلیک نموده و گزینه New Foreign Key را انتخاب کنید تا پنجره زیر ظاهر شود.



۲. روی ردیف Tables And Columns Specifications کلیک کنید تا دکمه انتهای سطر ظاهر شود. روی این دکمه کلیک کنید تا پنجره تعیین جداول و ستون‌ها نمایان شود.

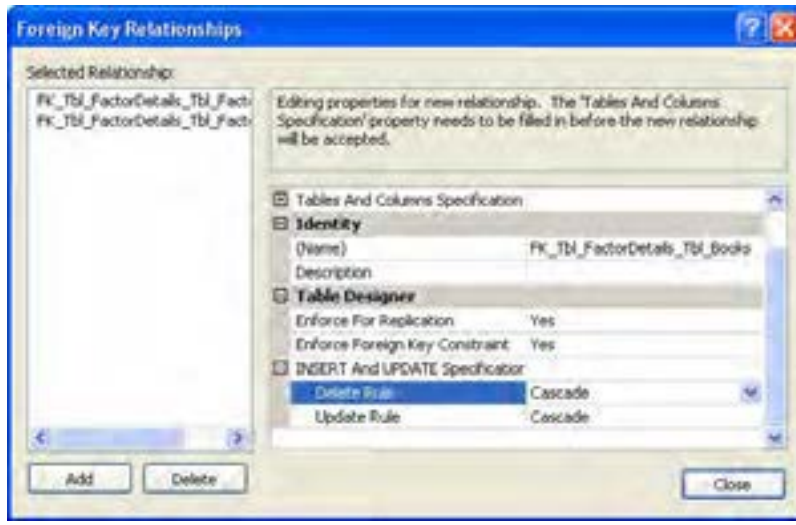
۳. در لیست Primary key table جدول Tbl_Books و فیلد ID را انتخاب کنید.



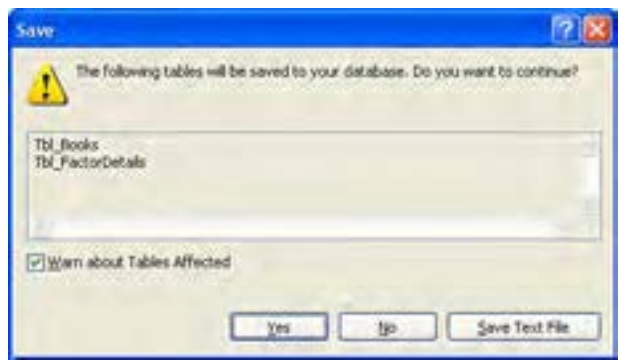
۴. در لیســـت Foreign key table، جدول Tbl_FactorDetails به صورت پیش‌فرض انتخاب شـــده است چون روی پوشه Keys این جدول راست‌کلیک کردیم. در لیست فیلدها BookID را انتخاب و در ردیفی که FactorID قرار دارد گزینه <None> را انتخاب کنید تا حذف شود.

۵. روی دکمه OK کلیک کنید تا به پنجره قبلی برگردید. نام کلید خارجی به صورت خودکار تغییر کرده است تا نشان‌دهنده جداول موجود در رابطه باشد.

۶. پوشـــه INSERT And UPDATE Specification را باز و Delete Rule و Update Rule را روی حالت CASCADE تنظیم کنید.



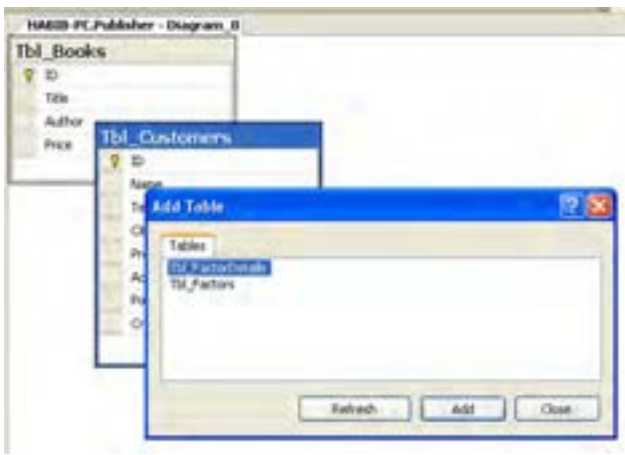
روی دکمه Close کلیک و سپس در نوار منوی برنامه روی آیکن دیسکت کلیک کنید. پیغام زیر از شما می‌پرسد که آیا می‌خواهید تغییرات را روی دو جدول ذخیره کنید.



۷. با کلیک روی دکمه Yes، محدودیت ایجاد شده در پایگاه داده ذخیره می‌شود.

۸. روی پوشـــه Database Diagram راست‌کلیک و گزینه New Database Diagram را انتخاب کنید. در صورت ظاهر شدن پیغام، آن را تأیید کنید.

۹. با کلیک روی دکمه Add، چهار جدول پایگاه داده را به نمودار اضافه کنید.



۱۰. بـــا کلیک روی نام جداول و جابه‌جا کـــردن آن‌ها، نمودار را منظم کنیـــد. روابطی که در مراحل قبل ساختید روی نمودار نشان داده می‌شود.

۱۱. روی فیلد ID از جدول مشـــتریان کلیک نموده و اشاره‌گر را روی فیلد CustomerID از جدول فاکتور بکشید و رها کنید. پنجره ایجاد رابطه که در همین تمرین با آن آشنا شدید ظاهر می‌شود.

۱۲. رابطه را ایجاد و تغییرات را ذخیره‌سازی کنید. همان‌طور که مشاهده کردید در نرم‌افزار Management Studio برای انجام هر کار چندین راه پیش‌بینی شده و این از مزایای قابل توجه محیط‌های ویژوال است.

۷-۵ آشنایی با تراکنش‌ها[1]

یک تراکنش مجموعه‌ای از دستورات است که تغییری را در پایگاه داده ایجاد می‌کند. نکته مهم در مورد تراکنش این اســـت که مجموعه دستورات موجود در آن یا به صورت کامل اجرا می‌شود یا هیچ‌کدام از آن‌ها به اجرا در نمی‌آید تا در نهایت سازگاری داده‌ها محفوظ بماند.

1 . Transactions

فرض کنید در یک دستگاه خودپرداز، دستور انتقال مبلغی را از حساب خود به حساب دیگری صادر کرده‌اید. اگر عملیات اول که کسر مبلغ از حساب شماست اجرا شود اما بنا به دلایلی عملیات وازیر به حساب مقصد صورت نگیرد، در عمل چه اتفاقی می‌افتد؟ پاسخ روشن است؛ پایگاه داده در وضعیت ناسازگار قرار می‌گیرد و صحت داده‌های موجود در آن از بین می‌رود.

برای غلبه بر چنین مشکلی، مجموعه دستورات را درون یک تراکنش قرار می‌دهند تا در صورت اجرای موفق همه دستورات، عمل تثبیت (COMMIT) صورت گیرد و در غیر این‌صورت همه تغییرات برگردانده (ROLLBACK) شود. الگوی ایجاد یک تراکنش به صورت زیر است.

BEGIN TRAN نام تراکنش

مبلغ را از موجودی حساب اول کم کن

مبلغ را به موجودی حساب دوم اضافه کن

COMMIT TRAN نام تراکنش در صورت عدم وقوع خطا

ROLLBACK TRAN نام تراکنش در صورت بروز خطا

فرض کنید، Table1 نام جدولی است که فقط یک فیلد دارد. نتیجه اجرای تراکنش زیر چیست؟

```
BEGIN TRAN T1
    INSERT INTO Table1 VALUES(1)
    INSERT INTO Table1 VALUES(2)
ROLLBACK TRAN T1
GO
INSERT INTO Table1 VALUES(3)
GO
SELECT * FROM Table1
```

با وجود این که در نگاه اول به نظر می‌رسد اعداد ۱ و ۲ و ۳ در جدول درج شده‌اند و باید برگردانده شوند اما فقط عدد ۳ برگردانده می‌شود چون دستور ROLLBACK TRAN، همه دستورات پیش از خود را تا دستور BEGIN TRAN که نام تراکنش مشابهی دارد لغو می‌کند. بنابراین اعداد ۱ و ۲ در جدول درج نمی‌شوند.

A transaction is a sequence of operations performed as a single logical unit of work. A logical unit of work must exhibit four properties, called the atomicity, consistency, isolation, and durability (ACID) properties, to qualify as a transaction.

Atomicity

A transaction must be an atomic unit of work; either all of its data modifications are performed, or none of them is performed.

Consistency

When completed, a transaction must leave all data in a consistent state. In a relational database, all rules must be applied to the transaction's modifications to maintain all data integrity. All internal data structures, such as B-tree indexes or doubly-linked lists, must be correct at the end of the transaction.

Isolation

Modifications made by concurrent transactions must be isolated from the modifications made by any other concurrent transactions. A transaction either recognizes data in the state it was in before another concurrent transaction modified it, or it recognizes the data after the second transaction has completed, but it does not recognize an intermediate state.

Durability

After a transaction has completed, its effects are permanently in place in the system. The modifications persist even in the event of a system failure.

۱. متن بالا را مطالعه کرده و در مورد آن توضیح دهید.

۲. چهار ویژگی تراکنش‌ها را توضیح دهید؟

۳. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. مشخصات سیستم پایگاه داده SQL Server را توضیح دهید.

۲. تسلط بر کدنویسی در محیط Management Studio چه ضرورتی دارد؟

۳. جدول مشـــتریان را با استفاده از کدهای SQL تولید کنید. در این جدول، شماره مشتری کلید اصلی است و ضمناً شماره مشتری در جدول فاکتورها به آن ارجاع دارد.

۴. انواع یکپارچگی را با ذکر مثال شرح دهید.

۵. استفاده از محدودیت‌ها به جای قواعد چه مزایایی دارد؟

۶. کد زیر چه مقادیری را برمی‌گرداند ؟

```
USE tempdb
CREATE TABLE Table1
( field1 int)
BEGIN TRAN T1
INSERT INTO Table1 VALUES(1)
INSERT INTO Table1 VALUES(2)
ROLLBACK TRAN T1
GO
INSERT INTO Table1 VALUES(3)
BEGIN TRAN T2
INSERT INTO Table1 VALUES(4)
ROLLBACK TRAN T2
GO
SELECT * FROM Table1
WHERE field1>3
```

فصل هشتم

فصل هشتم : پرس‌وجوهای جدول

در فصل ســـوم با نحوه نگارش پرس‌وجوهای محاسباتی و عملیاتی آشنا شدید و برخی جنبه‌های کار با دستور SELECT را در قالب مثال‌های کاربردی مشاهده کردید. در این فصل قصد داریم نحوه عملکرد این دستور بسیار مهم را با ذکر جزییات بیش‌تر و برشمردن دستورات قابل اعمال در آن مورد بررسی قرار دهیم.

۸-۱ آشنایی با اجزاء یک دستور SELECT

SELECT یک دستور فوق‌العاده قوی برای بازیابی[1] داده‌ها از یک یا چند جدول پایگاه داده است. علاوه بـــر ایـــن می‌تواند در حین بازیابی داده‌ها، آن‌ها را مرتب نموده یا محاســـباتی روی آن‌ها انجام دهد. نگارش خلاصه شده این دستور به صورت زیر است.

```
SELECT [ ALL | DISTINCT ]
[ TOP expression [ PERCENT ]]
{
* | { column_name }
[ [ AS ] column_alias ] | column_alias = expression
}
FROM table_name | view_name | alias_name
WHERE filter_criteria
ORDER BY ordering_criteria
```

1 . Retrieve

بررسی کد :

- **SELECT:** قسمت اجباری دستور – به SQL Server اعلام می‌کند دستور بازیابی اطلاعات در حال اجراست.
- **[DISTINCT | ALL] :** قسمت اختیاری دستور – All همه ردیف‌ها را بر می‌گرداند اما DISTINCT ردیف‌های غیر تکراری را بازیابی می‌کند.
- **TOP :** قسمت اختیاری دستور - برای جداسازی بخشی از داده‌ها به صورت تعداد مشخصی از ردیف‌ها یا درصدی از آن‌ها کاربرد دارد.
- *** :** قسمت اختیاری دستور - تمام ستون‌های جدول را برمی‌گرداند. بدون استفاده از علامت ستاره باید نام تک‌تک ستون‌ها را ذکر کنید.
- **column_name:** قسمت اختیاری دستور - در صورت عدم استفاده از علامت ستاره باید نام ستون‌های موردنظر را قید کنید. توصیه می‌شود در ابتدای نام ستون، نام جدول را نیز قید کرده و این دو عبارت را با نقطه از هم جدا کنید. مانند Tbl_Books.ID
- **AS** : قسمت اختیاری دستور – با استفاده از این عبارت می‌توانید نام سرستون‌ها را تغییر دهید و در واقع برای آن‌ها نام مستعار[1] تعیین کنید.
- **FROM table_name | view_name | alias_name :** قسمت اجباری دستور - مشخص می‌کند که اطلاعات از کدام جدول یا نما[2] (دیدگاه) بازیابی می‌شوند. امکان استفاده از نام مستعار آن‌ها هم وجود دارد.
- **WHERE filter_criteria :** قسمت اختیاری دستور - اگر می‌خواهید فقط ردیف‌هایی را بازیابی کنید که با معیارهای[3] خاصی تطابق دارند باید معیار موردنظر را جلوی عبارت WHERE بنویسید.
- **ORDER BY ordering_criteria :** قسمت اختیاری دستور – برای مرتب‌سازی داده‌ها از این عبارت[4] استفاده می‌شود.

ساختار جداول در این بخش مانند بخش های قبلی کتاب است.

---

1 . Alias
2 . View
3 . Criteria
4 . Clause

Tbl_Books

Column Name	Data Type	Allow Nulls
ID	int	☐
Title	nvarchar(100)	☑
Author	nvarchar(50)	☑
Price	smallmoney	☑
Pages	smallint	☑

Tbl_Customers

Column Name	Data Type	Allow Nulls
ID	int	☐
Name	nvarchar(80)	☑
Tel	varchar(11)	☑
City	nvarchar(50)	☑
Province	nvarchar(50)	☑
Address	nvarchar(300)	☑
PostalCode	char(10)	☑

Tbl_FactorDetails

Column Name	Data Type	Allow Nulls
FactorID	int	☐
BookID	int	☐
Quantity	smallint	☑
Discount	tinyint	☑

Tbl_Factors

Column Name	Data Type	Allow Nulls
ID	int	☐
CustomerID	int	☐
Date	char(10)	☑

مثال ۱: ساده‌ترین حالت استفاده از دستور انتخاب را برای برگرداندن مشخصات کتاب‌ها بنویسید.

```
SELECT * FROM Tbl_Books
```

مثال ۲: پرس‌وجویی بنویســـید که عنوان کتاب‌ها و نام نویسندگان را استخراج نموده و برای این ستون‌ها نام مستعار ایجاد کند. این پرس‌وجو را در محیط Management Studio نوشته و اجرا کنید.

دقت داشـــته باشید که اســـتفاده از عبارت AS اختیاری اســـت و لذا نگارش کد فوق به صورت زیر هم صحیح است :

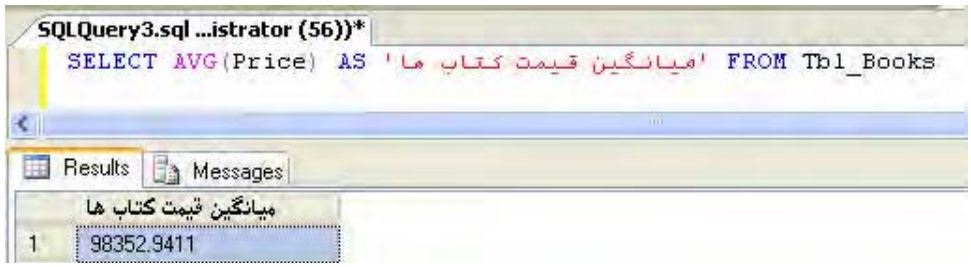
```
SELECT Title 'عنوان کتاب', Author 'نویسنده' FROM Tbl_Books
```

مثال ۳: پرس‌وجویی بنویسید که لیست نویسندگان را بدون تکرار نام آن‌ها استخراج کند.

```
SELECT DISTINCT Author FROM Tbl_Books
```

در صورت عدم اســـتفاده از عبارت DISTINCT، نام هر نویســـنده به تعداد کتاب‌هایی که نوشته تکرار می‌شود.

مثال ۴ : میانگین قیمت کتاب‌ها را محاسبه کنید.



مثال ۵ : پرس‌وجویی بنویسید که زمان و تاریخ جاری را برگرداند.

در بررسی اجزاء دستور SELECT مشاهده کردید که امکان تعیین نام مستعار برای ستون‌ها وجود دارد. این کار که برای افزایش خوانایی کد و گویاتر شدن نتایج انجام می‌گیرد، در مورد جداول هم قابل پیاده‌سازی است. با بهره‌گیری از این قابلیت می‌توانید به جای درج نام کامل جدول، از یک نام کوتاه استفاده کنید. برای نمونه می‌توانید با استفاده از پرس‌وجوی زیر به جدول کتاب‌ها نام مستعار b را تخصیص دهید :

```
SELECT b.Title , b.Price
FROM Tbl_Books b
WHERE b.Price > 50000
یا
SELECT b.Title , b.Price
FROM Tbl_Books AS b
WHERE b.Price > 50000
```

این کار در هنگام ایجاد پرس‌وجوهای ترکیبی که در آن‌ها جداولی با فیلدهای هم‌نام وجود دارند به شما کمک زیادی در خلاصه‌نویسی پرس‌وجو خواهد کرد.

نکته مهم دیگر در مورد دستور SELECT امکان ایجاد پرس‌وجوهای تودرتو[1] است. به این معنی که یک دستور SELECT می‌تواند بر روی نتایج حاصل از اجرای یک دستور SELECT دیگر اجرا شود. فرض کنید می‌خواهیم نام کتاب‌هایی را به دست آوریم که قیمت آن‌ها از میانگین قیمت کل کتاب‌ها بیش‌تر است.

```
SELECT Title FROM Tbl_Books
WHERE Price >
(SELECT AVG(Price) FROM Tbl_Books)
```

بررسی کد :

- دستور SELECT دوم که درون پرانتز قرار گرفته، میانگین قیمت کتاب‌های موجود در جدول را محاسبه می‌کند. این دستور حتماً باید درون پرانتز قرار گیرد.
- دستور SELECT اول با درنظر گرفتن میانگین قیمت کتاب‌ها، عناوینی را برمی‌گرداند که قیمتی بیش‌تر از قیمت میانگین دارند.

۸-۲ اصول انجام Filters

برای فیلتر کردن نتایج، یعنی جداسازی بخشی از رکوردها که با یک معیار خاص تطابق دارند از عبارت WHERE استفاده می‌شود که در ترکیب با عمل‌گرهای ریاضی و منطقی می‌توانند داده‌ها را به صورتی که مدنظر کاربر است جداسازی نماید. با توجه به توضیح روش‌های Filtering در فصل سوم، از تکرار آن صرف‌نظر می‌کنیم.

۸-۳ اصول کار با عمل‌گرها

در فصل سوم با کارکرد عمل‌گرهای مقایسه‌ای و منطقی آشنا شدید. این بخش ضمن معرفی سایر عمل‌گرهای موجود در SQL، نحوه کار آن‌ها را در قالب چندین مثال توضیح می‌دهد.

۸-۳-۱ عمل‌گرهای توضیح[2]

هنگام نوشتن کدهای SQL می‌توانید توضیحاتی را در مورد دستورات، درون کدها بگنجانید. این کار در هنگام ایجاد پرس‌وجوهای پیچیده کاملاً ضروری است تا اگر فرد دیگری در آینده به کدها رجوع کرد بتواند با صرف کم‌ترین وقت ممکن، بر روش کار آن‌ها تسلط پیدا کند.

---

1 . Nested

2 . Comment Operators

بـــرای نوشـــتن توضیحات چند خطی باید عبارت‌های موردنظـــر را درون جفت علامت‌های /* و */ قرار دهید اما اگر توضیحات شما تک خطی است، استفاده از عمل‌گر – کفایت می‌کند. این توضیحات در محیط Management Studio به رنگ سبز در می‌آیند و SQL Server هنگام بررسی کد، آن‌ها را نادیده می‌گیرد.

```
SQLQuery6.sql ...istrator (54))*
/* This is a multiple-line comment.
All of the text after the first comment operator
up to the closing comment operator
will be ignored */

USE Publisher
GO
SELECT * From Tbl_Books -- this query will retrieves Books

-- این توضیحی یک خطی است
```

۲-۳-۸ عمل‌گرهای محاسباتی

عمل‌گرهای زیر برای انجام محاســـبات ریاضی در کدهای SQL به کار می‌روند. تنها نکته قابل ذکر در مورد آن‌ها، توجه به هماهنگی انواع داده‌های هنگام استفاده از عمل‌گرهایی مثل تقسیم است.

عمل‌گر	عمل‌کرد	مثال برای اعداد صحیح
+	جمع	۲۶+۱۰=۳۶
–	تفریق	۲۶-۱۰=۱۶
*	ضرب	۲۶*۱۰=۲۶۰
/	تقسیم	۲۶/۱۰=۲
%	محاسبه باقیمانده	۲۶%۱۰=۲
+	مثبت	+۱۰ = ۱۰
–	منفی	-۱۰ = -۱۰

مثال ۱: پرس‌وجویی بنویســـید که حاصل تقســـیم یک عدد صحیح و یک عدد اعشاری را بر عدد صحیح و نیز باقی‌مانده یک تقسیم را نشان دهد.

```
SELECT 16/10 , 16.0/10 , 16%10
```

	(No column name)	(No column name)	(No column name)
1	1	1.600000	6

مثال ۲ : پرس‌وجویی بنویسید که قیمت کتاب‌ها را پس از اعمال ۲۵% تخفیف نشان دهد.

```
USE Publisher
GO
SELECT * , Price * 0.75 AS [New Price]
FROM Tbl_Books
```

	ID	Title	Author	Price	New Price
1	1201	Access 2007	رضا کبیری	65000.00	48750.000000
2	1202	طراحی صفحات وب	علی محمدی	35000.00	26250.000000
3	1203	برنامه نویسی به زبان دلفی	منصور کبیری	80000.00	60000.000000

مثال ۳ : پرس‌وجویی بنویسید تا مشخصات کتاب‌هایی را برگرداند که شماره آن‌ها به ۳ ختم می‌شود.

```
GO
SELECT *
FROM Tbl_Books
WHERE ID % 10 = 3
```

	ID	Title	Author	Price
1	1203	برنامه نویسی به زبان دلفی	منصور کبیری	80000.00
2	1213	مرجع کامل SQL Server	حامد رضایی	35000.00

۸-۳-۳ عمل‌گرهای بیتی

گاهی اوقات استفاده از داده‌های دودویی برای ذخیره‌سازی اطلاعات از نظر کارایی و حجم پایین بهترین انتخاب اســـت. برای مثال عددی دودویی مانند ۱۰۱۱۱۰۰۱۱۰ می‌تواند مشـــخص کند که یک کاربر به ده ســـؤال «بله» یا «خیر» چه پاســـخ‌هایی داده اســـت. در جدول زیر نحوه رفتار عمل‌گرهای بیتی را مشاهده می‌کنید.

عمل‌گر	کارکرد
&	اگر هر دو بیت یک باشند، یک برمی‌گرداند و در غیر این‌صورت صفر
|	اگر هر دو بیت صفر باشند، صفر برمی‌گرداند و در غیر این‌صورت یک
^	اگر بیت‌ها مثل هم باشند صفر برمی‌گرداند و در غیر این‌صورت یک
~	بیت‌ها را معکوس می‌کند

مثال ۱: حاصل اجرای پرس‌وجوی SELECT 7 ^ 12 چیست؟

```
0000 0111
0000 1100
---------------
0000 1011
```

بنابراین عدد ۱۱ برگردانده می‌شود.

۸-۴ مرتب‌سازی داده‌ها

وقتی دستور بازیابی داده‌ها را روی جدولی اجرا می‌کنید، در بیش‌تر موارد لازم است رکوردهای استخراج شـــده را بر اســـاس یکی از فیلدها مرتب نمایید. این کار با استفاده از عبارت ORDER BY که یکی از اجزاء اختیاری دستور SELECT است انجام می‌شود. با استفاده از این عبارت می‌توانید رکوردهای بازگشتی را بر اســـاس یک یا چند فیلد به صورت صعودی[1] یا نزولی[2] مرتب کنید. برای مرتب‌سازی صعودی باید کلیدواژه ASC را پس از نام فیلد درج نمایید. عبارت DESC هم برای مرتب‌سازی نزولی کاربرد دارد.

مثال ۱: پرس‌وجویی بنویســـید که نام کتاب‌ها را به صورت صعودی مرتب کند. این پرس‌وجو را

1 . Ascending
2 . Descending

در محیط Management Studio وارد و اجرا کنید.

```
SQLQuery1.sql ...istrator (53))*
USE Publisher
GO
SELECT * FROM Tbl_Books
ORDER BY Title ASC
```

Results | Messages

	ID	Title	Author	Price
1	1201	Access 2007	رضا اکبری	65000.00
2	1208	Dreamweaver	معصومه ناظری	120000.00
3	1214	Excel 2007	علی اکبری	200000.00
4	1216	Frontpage 2003	رضا اکبری	150000.00
5	1215	Word 2007	John Dewson	130000.00
6	1211	اینترنت	علی علوی	78000.00
7	1209	اینترنت	عاطفه مجیدی	85000.00

دستور ORDER BY به صورت پیش‌فرض داده‌ها را به شکل صعودی مرتب می‌کند، بنابراین حذف عبارت ASC تأثیری در نتایج مثال قبل ندارد.

معیار مرتب‌سازی حروف کد اسکی[1] آن‌هاست بنابراین حروف انگلیسی جلوتر از حروف فارسی قرار می‌گیرند.

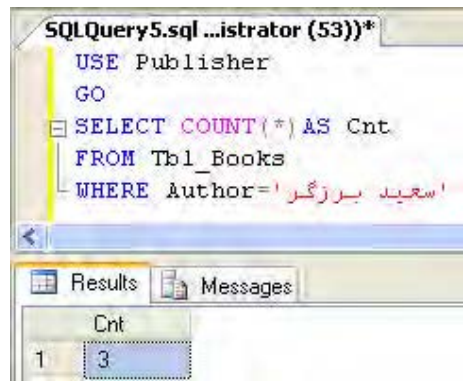
مثال ۲ : پرس‌وجویی بنویسید که داده‌های جدول کتاب را بر حسب نام کتاب‌ها به صورت صعودی مرتب کند. در صورت یکسان بودن نام چند کتاب، کتابی که قیمت بیش‌تری دارد در لیست نتایج بالاتر نشان داده شود.

1 . ASCII

۸-۵ کار با GROUP BY

در فصل ســـوم هنگام معرفی توابع تجمعی با کارکرد عبارت GROUP BY آشنا شدید. این عبارت برای گروه‌بندی تعدادی از ردیف‌های جدول و خلاصه‌سازی اطلاعات آن‌ها کاربرد دارد.

مثال ۱: پرس‌وجویی بنویسید که نشان دهد در جدول کتاب‌ها از نویسنده‌ای با نام «سعید برزگر» چند اثر وجود دارد.



مثال ۲: پرس‌وجو را به گونه‌ای تغییر دهید که مشـــخص شـــود هر نویسنده، چند اثر در جدول کتاب‌ها دارد و نتایج حاصل بر حسب تعداد اثر به صورت نزولی مرتب شود.

```
SQLQuery5.sql ...istrator (53))*
USE Publisher
GO
SELECT Author AS Name, COUNT(*) AS Cnt
FROM Tbl_Books
GROUP By Author
ORDER By Cnt DESC
```

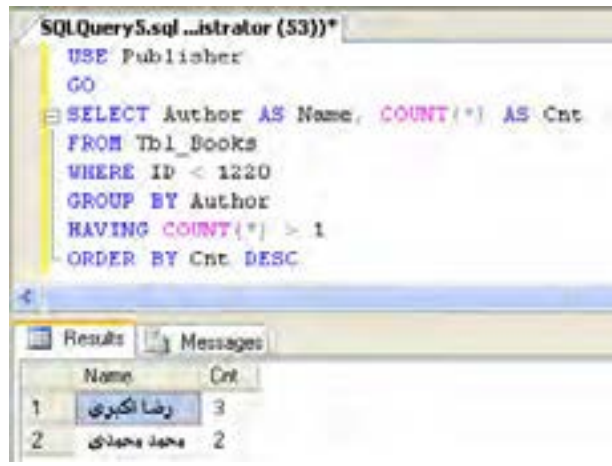
Results | Messages

	Name	Cnt
1	سعید برزگر	3
2	رضا اکبری	3
3	محمد محمدی	2
4	معصومه نظری	1
5	منصور کبیری	1
6	مهدی رضایی	1
7	رضا اکرمی	1

دستور GROUP BY به شما امکان می‌دهد روی تک‌تک رکوردهای یک جدول، عملیات موردنظر مثل شمارش، محاسبه میانگین و ... را انجام دهید و در واقع اطلاعات را بر حسب یک فیلد خلاصه کنید.

مثال ۳: پرس‌وجویی بنویسید که نشان دهد در میان کتاب‌هایی که شماره آن‌ها کم‌تر از ۱۲۲۰ اســـت، هر نویســـنده چند اثر دارد. (برخلاف نتایج مثال ۲، نام نویسنده‌ای مانند«سعید برزگر» را نمی‌بینید چون کتاب‌های وی شماره‌هایی بیش‌تر از ۱۲۲۰ دارند)

مثال ۴: پرس‌وجوی مثال قبل را به گونه‌ای تغییر دهید که فقط اســـامی نویســـندگانی را نمایش دهد که بیش از یک اثر دارند.



همان‌طور که در مثال قبل مشـــاهده کردید برای جداســـازی داده‌ها در جدول اصلی می‌توان از عبارت WHERE اســـتفاده کرد اما هنگامی که می‌خواهید شـــرطی را روی نتایج حاصل از گروه‌بندی اعمال کنید، حتماً باید از عبارت HAVING اســـتفاده نمایید. این نکته را هم به یاد داشـــته باشـــید که در شرط جلوی HAVING اســـتفاده از اســـامی مستعار فیلدها مجاز نیســـت و مثلاً در صورت استفاده از نام مستعار Cnt با پیغام خطا مواجه خواهید شد.

۸-۶ کار با NULL

هنـــگام طراحی جـــدول می‌توانید برخی از فیلدها را طوری تنظیم کنید کـــه مقدار NULL را بپذیرند. NULL مشـــخص می‌کند که مقدار فیلد «نامشـــخص» اســـت و معنی «خالی بودن» یـــا «صفر بودن» را نمی‌دهد. در مورد فیلدی با مقدار NULL می‌توانیم این قضاوت را داشـــته باشیم که مقدار آن قرار است در آینده مشخص شود.

شـــاید این سؤال برای شـــما پیش بیاید که وجود مقدار NULL در طراحی پایگاه داده چه مزیتی دارد؟ پاســـخ بسیار ســـاده است. فرض کنید اگر مجبور بودید در یک فیلد عددی به جای مقدار NULL عدد صفر را قرار دهید، مشـــخص نبود که آیا صفر نشان دهنده عدم وجود مقدار است یا مشخص می‌کند واقعاً مقدار صفر درج شده است.

مثال ۱: پرس‌وجویی بنویسید تا رکوردهایی از جدول مشتریان را برگرداند که کدپستی آن‌ها وارد نشده اما استان محل سکونت آن‌ها مشخص است.

```
SQLQuery6.sql ...istrator (53))*
USE Publisher
GO
SELECT * FROM Tbl_Customers
WHERE Postalcode IS NULL AND Province IS NOT NULL
```

Results | Messages

	ID	Name	Tel	City	Province	Address	PostalCode
1	3	انتشارات امیرکبیر	22222222	شیراز	فارس	خیابان کریمخان- پلاک ۴۲	NULL
2	4	کتابفروشی اندیشه	88552211	اسلام شهر	تهران	خیابان شهید مطهری روبروی بیمه	NULL
3	7	اداره ارشاد سپیدان	4589552	سپیدان	فارس	خیابان شهید بهشتی پلاک ۲۲	NULL

۸-۷ دستکاری نویسه‌ها

وقتی SQL Server را روی یک رایانه نصب می‌کنید می‌توانید تعیین نمایید که برنامه در مواجهه با حروف انگلیسی، حساس به متن[1] باشد یا بین حروف بزرگ و کوچک تفاوتی قایل نشود. برای مثال ممکن است نام کتاب اکسس در جدول کتاب‌ها به صورت access، Access و یا ACCESS ذخیره شده باشد. حال اگر پرس‌وجوی زیر را برای بازیابی کتاب‌های اکسس در یک سیستم مدیریت پایگاه داده حساس به متن بنویسید تنها مواردی که از نظر بزرگی و کوچکی حروف با شرط تطابق دارند برگردانده می‌شوند.

```
SELECT * FROM Tbl_Books
WHERE Title='Access'
```

حال با استفاده از توابع دستکاری نویسه‌ها، پرس‌وجو را به گونه‌ای تغییر می‌دهیم که نوع حروف از نظر بزرگی و کوچکی، خللی در اجرای صحیح پرس‌وجو و بازیابی کامل نتایج ایجاد نکند.

```
SELECT * FROM Tbl_Books
WHERE UPPER(Title) ='ACCESS'
```

و یا

```
SELECT * FROM Tbl_Books
WHERE LOWER(Title) ='access'
```

بنابراین توابع UPPER و LOWER نتایج حاصل از پرس‌وجو را به صورت موقت تبدیل به حروف بزرگ و کوچک می‌کنند تا عمل مقایسه با دقت بیش‌تری انجام گیرد.

۸-۸ دستکاری DATETIME

در فصل قبل با روش ایجاد فیلدهایی آشنا شدید که داده‌هایی از نوع تاریخ و زمان ذخیره می‌کنند. هر کدام از این انواع داده‌ای برحسب حجمی که اشغال می‌کنند، محدوده‌ای از تاریخ و زمان را با دقت خاصی پوشش می‌دهند. در SQL Server برای کار با این انواع داده‌ای توابعی پیش‌بینی شده که می‌توانند بخشی از تاریخ یا زمان را جدا نموده و یا مقدار آن را افزایش یا کاهش دهند. در ادامه تعدادی از این توابع را با ذکر مثال بررسی خواهیم کرد.

GETDATE() : تاریخ میلادی جاری را برمی‌گرداند.

DATEPART(datepart, date) : آرگومان date یک تاریخ میلادی را می‌گیرد و بسته به این‌که

1 . Case-Sensitive

آرگومان datepart چگونه مقداردهی شـــده باشـــد، یک عدد صحیح به عنوان نتیجه تابع برگردانده می‌شود. این عدد صحیح نشان‌دهنده یکی از اجزاء تاریخ مانند «شماره روز» است و می‌تواند مقادیر زیر را بپذیرد.

مقدار آرگومان	خروجی تابع	مقدار آرگومان	خروجی تابع
year	سال	week	شماره هفته
month	شماره ترتیبی ماه	weekday	شماره روز در هفته
day	روز	dayofyear	شماره ترتیبی روز در سال
hour	ساعت	minute	دقیقه
second	ثانیه	millisecond	میلی ثانیه

مثال ۱: پرس‌وجویی بنویسید که مشخص کند چند روز به پایان سال جاری باقی‌مانده است.

```
SELECT 365 - DATEPART(DAYOFYEAR, GETDATE())
```

مثال ۲: سومین روز ماه May سال ۲۰۱۰، چندمین روز هفته است؟

```
SELECT DATEPART(WEEKDAY, '2010-May-03')
```

یا

```
SELECT DATEPART(WEEKDAY, '2010-05-03')
```

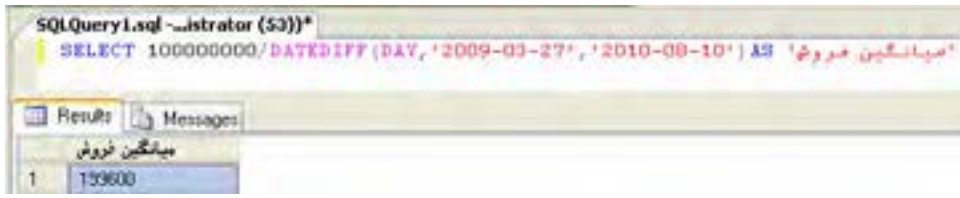
این پرس‌وجو عدد ۲ را برمی‌گرداند به این معنی که در تقویم میلادی، دومین روز هفته (سه‌شنبه) است.

DATEADD(datepart, number, date): این تابع تاریخی را که به عنوان آرگومان date داده شـــده، به میزان number و برحسب واحد مشخص شده در آرگومان datepart جلو می‌برد. مقادیر datepart همانند جدول قبل است.

مثال ۳: پرس‌وجویی بنویسید که نشان دهد ۵۰ روز بعد چه تاریخی است.

DATEDIFF(datepart, startdate, enddate) : بـــرای محاســـبه تعداد فواصل زمانی میان دو تاریخ می‌توانید از این تابع استفاده کنید. datepart مشخص کننده واحد موردنظر، startdate شروع بازه و enddate انتهای بازه زمانی است. آرگومان datepart این تابع هم از جدول قبل تبعیت می‌کند.

مثال ۴ : مجموع فروش یک انتشـــارات در بازه زمانـــی ۲۰۰۹-۰۶-۱۷ تا ۲۰۱۰-۰۸-۱۱ مبلغ صد میلیون ریال است. میانگین فروش روزانه را محاسبه کنید.



مثال ۵ : پرس‌وجوی زیر چه عددی را برمی‌گرداند؟

```
SELECT DATEDIFF(year, '2009-12-31 23:59:59.9999999'
, '2010-01-01 00:00:00.0000000');
```

پاسخ : با وجود این که تفاوت دو تاریخ فوق تنها 0.0000001 ثانیه است اما این تابع عدد یک را به عنوان فاصله زمانی این دو تاریخ بر حسب سال محاسبه خواهد کرد.

۹-۸ پرس‌وجوی Metadata

مِتادیتا یعنی داده‌هایی در مورد داده‌ها و پرس‌وجوهایی این دست به جای آن که داده‌های درون پایگاه داده را برگرداند، اطلاعاتی را پیرامون پایگاه داده و اشـــیاء درون آن در اختیار کاربر قرار می‌دهد. برای مثال با اجرای یکی از این توابع می‌توانید طول ســـتونی را به دســـت آورید که داده‌ها درون آن ذخیره شده‌اند. در ادامه کاربرد تعدادی از توابع را همراه با ذکر مثال بررسی می‌کنیم.

COL_LENGHT(‘table’, ‘column’) : طول ســـتون مشـــخص شـــده از جدول را بر حسب بایت برمی‌گرداند.

مثال : پرس‌وجوی بنویسید که طول ستون عنوان کتاب را از جدول کتاب‌ها به دست آورد.

```
SELECT COL_LENGTH('Tbl_Books','Title')
```

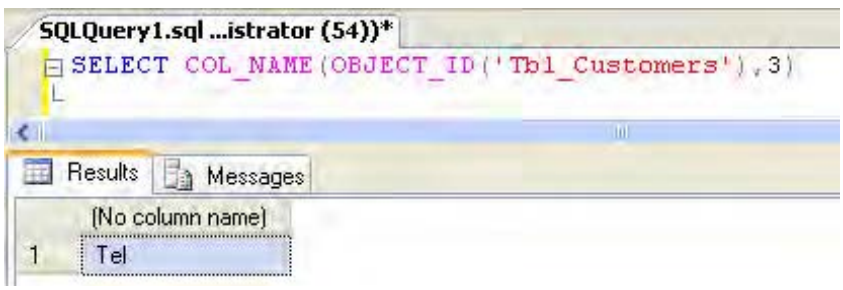
OBJECT_ID(‘object’) : هر شـــیء درون پایگاه داده یک شناسه[1] منحصربفرد دارد که مقدار آن با این تابع به دست می‌آید.

مثال : شناسه جدول کتاب‌ها را در پایگاه داده Publisher به دست آورید.

DB_ID(‘database name’) : شناسه پایگاه داده را استخراج می‌کند.

COL_NAME(table_id, column_id) : نام ستونی از جدول را برمی‌گرداند که شناسه آن به عنوان آرگومان table_id وارد شده است.

مثال : با استفاده از این تابع، نام سومین ستون از جدول مشتریان را به دست بیاورید.



COLUMNPROPERTY(table_id, ‘column_name’, ‘property’) : بســـته به مقدار آرگومان property ، مقدار متناظر با یکی از ویژگی‌های ســـتون را برمی‌گرداند. برای نمونه AllowsNULL مشخص

1 . Identifier (ID)

می‌کند که آیا ستون، مقادیر NULL را می‌پذیرد.

مثال : نتیجه اجرای پرس‌وجوی زیر چیست؟

```
SELECT COLUMNPROPERTY(OBJECT_ID('Tbl_Factors'),'ID','AllowsNull')
```

از آن‌جا که ستون ID از جدول فاکتورها مقدار NULL نمی‌پذیرد، خروجی این پرس‌وجو عدد صفر است.

DATABASEPROPERTY('database name' ,'property') : بسته به مقدار آرگومان property، مقدار متناظر با یکی از ویژگی‌های پایگاه داده را برمی‌گرداند. برای نمونه IsOffline مشخص می‌کند که آیا پایگاه داده در حالت آف‌لاین[1] قرار دارد یا خیر. در حالت آف‌لاین، پایگاه داده موقتاً از سرویس‌دهی خارج می‌گردد و امکان تغییر اشیاء و داده‌های آن وجود ندارد.

مثال : آف‌لاین بودن پایگاه داده Publisher را بررسی کنید.

چنان‌چه نتیجه اجرای این پرس‌وجوی زیر عدد یک باشد، پایگاه داده در حالت آف‌لاین است.

```
SELECT DATABASEPROPERTY('Publisher' , 'IsOffline')
```

OBJECTPROPERTY(id , 'property') : با توجه به مقدار آرگومان property، یکی از ویژگی‌های شیئی را که شناسه آن وارد شده بررسی می‌کند.

مثال : آیا شیئی با نام Tbl_Books در پایگاه داده Publisher وجود دارد؟ آیا از نوع جدول است؟

پاسخ : پرس‌وجوی زیر را اجرا و نتیجه اجرا را با جدول نتایج مقایسه می‌کنیم.

```
SELECT OBJECTPROPERTY(OBJECT_ID('Tbl_Books') , 'IsTable')
```

---

1 .Offline

نتیجه	معنی
NULL	چنین شیئی در پایگاه داده وجود ندارد.
0	این شیء در پایگاه داده وجود دارد اما جدول نیست.
1	شیء موردنظر از نوع جدول است.

۱۰-۸ کار با TOP

گاهی اوقات لازم اســت هنگام اجرای دستور SELECT تنها بخشی از داده‌های بازیابی شده را مشاهده نمایید. برای مثال هنگامی که در یکی از وب‌ســایت‌های جستجو مانند گوگل[1]، عبارتی را جستجو می‌کنید ممکن اســت هزاران نتیجه برای نمایش وجود داشــته باشد اما نمایش یک‌باره آن‌ها ضرورت چندانی ندارد چون صرف‌نظر از ایجاد مشــکلات فنی مانند مشــغول شــدن بیش از حد پایگاه داده، ممکن است نتیجه موردنظر کاربر در میان چند گزینه اول باشــد. در چنین شرایطی، تنها بخشی از نتایج نشان داده می‌شود و نمایش ادامه آن‌ها مشروط به درخواست کاربر است.

برای پیاده‌ســازی چنین حالت‌هایی، در کنار دستور SELECT از عبارت TOP استفاده می‌شود که قادر است «تعداد» یا «درصد» مشخصی از سطرهای جدول را برگرداند. نگارش این عبارت به یکی از دو صورت زیر است :

```
SELECT TOP n Column_name FROM Table_name
SELECT TOP n Percent Column_name FROM Table_name
```

مثال ۱: پرس‌وجویی بنویســید که عنوان و قیمت ۱۰ کتاب را که دارای بالاترین قیمت هســتند نشان دهد.

```
SELECT TOP 10 Title, Price FROM Tbl_Books
ORDER BY Price DESC
```

مثال ۲ : برای انجام یک نظرسنجی به اطلاعات ۲۰% درصد از مشتریان استان تهران نیاز داریم. پرس‌وجویی بنویسید که این داده‌ها را استخراج کند.

```
SELECT TOP 20 PERCENT * FROM Tbl_Customers
WHERE Province='تهران'
```

1 . www.google.com

۸-۱۱ اجرای Ranking

در SQL Server توابعی برای رتبه‌بندی داده‌های یک جدول وجود دارد که هر کدام از آن‌ها با بررسی فیلدهای یک رکورد و مقایسه آن‌ها با مقادیر فیلدهای سایر رکوردها، به روش خاصی یک رتبه را به رکورد نسبت می‌دهند.

یک مثال قابل درک از رتبه‌بندی، نمراتی است که در مدرسه به دانش‌آموزان اختصاص داده می‌شود. نمره دانش‌آموز می‌تواند عددی بین صفر تا بیست باشد اما اگر قرار باشد دانش‌آموزان را در چهار دسته «قوی، خوب، متوسط و ضعیف» قرار دهند به عنوان نمونه باید نمرات ۱۶ تا ۲۰ در یک دسته قرار گیرند.

برای اعمال رتبه‌بندی روی داده‌ها، چهار تابع وجود دارد که با هم آن‌ها را بررسی می‌کنیم.

RANK( ) : رکوردها را بر مبنای یکی از فیلدها دسته‌بندی می‌کند و به رکوردهای موجود در هر دسته، بر اساس فیلد دیگری یک رتبه اختصاص می‌دهد.

مثال : قصد داریم تعدادی کتاب را برای یک مشتری ارسال کنیم. مشتری از ما خواسته اگر از چند کتاب با عنوان‌های مشابه وجود دارد، کتابی ارسال شود که قیمت کم‌تر و صفحات بیش‌تری دارد. پرس‌وجویی بنویسید که اولویت انتخاب کتاب‌ها را مشخص کند.

```
SELECT *
,RANK() OVER (PARTITION BY Title ORDER BY Price ASC, Pages DESC) AS 'اولویت'
FROM Tbl_Books
ORDER BY Price
```

بررسی کد :

- به جدول کتاب‌ها فیلد جدیدی به نام Pages اضافه شده که تعداد صفحات کتاب را نشان می‌دهد.
- برای ایجاد یک ستون جدید به عنوان «رتبه» یا «اولویت»، تابع RANK() و کلیدواژه OVER را پس از نام ستون‌ها قرار می‌دهیم.
- عبارت PARTITION BY عمل دسته‌بندی رکوردها را بر اساس یک فیلد (در این‌جا Title) انجام می‌دهد.
- عبارت ORDER BY نحوه اختصاص رتبه به رکوردهای دسته‌بندی شده را مشخص می‌کند که در این‌جا اولویت با قیمت کم‌تر و تعداد صفحات بیش‌تر است.

با اجرای کد، نتیجه زیر ظاهر می‌شود.

	ID	Title	Author	Price	Pages	اولویت
1	1205	برنامه نویسی به زبان ویژوال بیسیک ۶	حامد تنیری	25000.00	100	1
2	1202	طراحی صفحات وب	علی محمدی	35000.00	180	1
3	1213	مرجع کامل SQL Server	حامد رضایی	35000.00	240	1
4	1223	برنامه نویسی پایگاه داده با دلفی	محمد سعیدی	35200.00	150	1
5	1222	FreeHand	سعید برزگر	52000.00	220	1
6	1219	برنامه نویسی به زبان دلفی	رضا اکبری	55000.00	200	1
7	1210	فتوشاپ	سعید برزگر	60000.00	390	1
8	1224	Access 2007	رضا باقری	65000.00	250	1
9	1201	Access 2007	رضا اکبری	65000.00	200	2
10	1221	CorelDraw	سعید برزگر	65000.00	320	1
11	1212	مهارت های پایه در برنامه نویسی	رضا اکرمی	69000.00	360	1
12	1220	Access 2007	مهدی رضا...	70000.00	300	3
13	1211	اینترنت	علی علوی	78000.00	300	1
14	1206	برنامه نویسی به زبان C#	محمد محمدی	80000.00	420	1
15	1203	برنامه نویسی به زبان دلفی	منصور گلب...	80000.00	360	1
16	1209	اینترنت	عاطفه مج...	85000.00	380	2
17	1207	فتوشاپ	اسد بیاری	110000...	510	2

برای کتاب‌های موجود در جدول، بر اساس نام کتاب‌ها دسته‌بندی به صورت زیر انجام می‌شود و سپس بر اساس معیارهای مرتب‌سازی رتبه‌بندی صورت می‌گیرد.

	دسته ۴	دسته ۳ فتوشاپ	دسته ۲ اینترنت	دسته ۱ Access 2007
رتبه ۱			قیمت ۷۸۰۰۰	قیمت ۶۵۰۰۰ صفحات ۲۵۰
رتبه ۲			قیمت ۸۵۰۰۰	قیمت ۶۵۰۰۰ صفحات ۲۰۰
رتبه ۳				قیمت ۷۰۰۰۰
...				

DENSE_RANK( ) : هنگامـــی که تابع RANK اقدام به رتبه‌بندی رکوردها می‌کند، در صورت وجود

مقادیر یکسان در فیلد، میان رتبه‌ها شکاف[1] به وجود می‌آید. اما در رتبه‌بندی توسط تابع DENSE_RANK این شکاف‌ها وجود ندارند.

مثـــال ۱: پرس‌وجوی زیر را در Management Studio اجرا و نحوه کار دو تابع را با هم مقایســـه کنید.

```
SELECT Title, Price,
  RANK() OVER (ORDER BY Price) AS 'Rank',
  DENSE_RANK() OVER (ORDER BY Price) AS 'Dense Rank'
FROM Tbl_Books
```

Results | Messages

	Title	Price	Rank	Dense Rank
1	برنامه نویسی به زبان ویژوال بیسیک ۶	25000.00	1	1
2	طراحی صفحات وب	35000.00	2	2
3	مرجع کامل SQL Server	35000.00	2	2
4	برنامه نویسی پایگاه داده با دلفی	35200.00	4	3
5	FreeHand	52000.00	5	4
6	برنامه نویسی به زبان دلفی	55000.00	6	5
7	فتوشاپ	60000.00	7	6
8	Access 2007	65000.00	8	7
9	CorelDraw	65000.00	8	7
10	Access 2007	65000.00	8	7
11	مهارت های پایه در برنامه نویسی	69000.00	11	8

در تصویـــر فوق به ســـطرهای ۲ و ۳ دقت کنید. وقتی تابع RANK به دو قیمت مشـــابه ۳۵۰۰۰ ریال رســـیده، به هر دو رکورد، رتبه ۲ اختصاص داده اما برای قیمت بعدی رتبه ۴ را منظور کرده اســـت و به این ترتیب در رتبه ۳ دچار شکاف است. در عوض تابع DENSE_RANK رتبه‌ها را پشت سرهم و بدون شکاف تخصیص می‌دهد.

مثـــال ۲: در جدول کتاب‌ها، چهار کتـــاب ۵۰۰۰۰ ریالی و یک کتاب ۶۰۰۰۰ ریالی وجود دارد. توابع RANK و DENSE_RANK به صورت صعودی چه رتبه‌ای را به آن‌ها اختصاص می‌دهند؟

---

1 . Gap

تابع / قیمت	۵۰۰۰۰	۵۰۰۰۰	۵۰۰۰۰	۵۰۰۰۰	۶۰۰۰۰
RANK	۱	۱	۱	۱	۵
DENSE_RANK	۱	۱	۱	۱	۲

NTILE (n) : عدد n را دریافت نموده و رکوردها را به n دسته تقسیم می‌کند. سپس شماره دسته هر رکورد را برمی‌گرداند.

مثال : پرس‌وجوی زیر را روی جدول کتاب‌ها اجرا می‌کنیم. چه نتیجه‌ای به دست می‌آید؟

```
SELECT Title, Price,
  NTILE(4) OVER (ORDER BY Price) AS 'Quartile'
FROM Tbl_Books
```

کتاب‌ها از لحاظ قیمت به چهار دسته تقسیم می‌شوند و هر کتاب در یکی از این دسته‌ها قرار می‌گیرد.

```
SELECT ID,Title, Price,
    NTILE(4) OVER (ORDER BY Price) AS 'چهارتایی'
FROM Tbl_Books
WHERE ID < 1215
```

Results | Messages

	ID	Title	Price	چهارتایی
1	1205	برنامه نویسی به زبان ویژوال بیسیک ۶	25000.00	1
2	1202	طراحی صفحات وب	35000.00	1
3	1213	مرجع کامل SQL Server	35000.00	1
4	1210	فتوشاپ	60000.00	1
5	1201	Access 2007	65000.00	2
6	1212	مهارت های پایه در برنامه نویسی	69000.00	2
7	1211	اینترنت	78000.00	2
8	1203	برنامه نویسی به زبان دلفی	80000.00	3
9	1206	برنامه نویسی به زبان C#	80000.00	3
10	1209	اینترنت	85000.00	3
11	1207	فتوشاپ	110000.00	4
12	1208	Dreamweaver	120000.00	4
13	1214	Excel 2007	200000.00	4

ROW_NUMBER( ) : به هر رکورد یک شماره ترتیبی اختصاص می‌دهد. در صورت استفاده از عبارت PARTITION BY رکوردها را دسته‌بندی نموده و برای هر دسته یک شماره ترتیبی در نظر می‌گیرد. شماره‌های ترتیبی از یک شروع می‌شوند.

مثال ۱ : پرس‌وجویی بنویسید که بر اساس ترتیب نزولی قیمت کتاب‌ها، به هر رکورد یک شماره ترتیبی اختصاص دهد.

مثال ۲ : ستونی ایجاد کنید که به کتاب‌های هر نویسنده به ترتیب صعودی عنوان کتاب‌ها یک شماره ترتیبی اختصاص دهد.

۱۲-۸ کار با عبارت CASE

تابع CASE یک مقدار را با چند شـــرط تطبیق می‌دهد و بسته به حالت‌هایی که برای آن تعریف شده، مقدار خاصی را برمی‌گرداند. CASE برخلاف IF به تنهایی کاربرد ندارد بلکه بخشی از یک عبارت SELECT یا UPDATE محسوب می‌شود.

این تابع به دو شکل ساده[1] و جستجویی[2] مورد استفاده قرار می‌گیرد.

الف) حالت ســـاده : در این حالت تابع CASE یک عبارت را با مجموعه‌ای از عبارات ســـاده مقایسه و نتیجه را مشخص می‌کند. شکل کلی حالت ساده به صورت زیر است :

```
CASE value
  WHEN expression_1 THEN result_expression_1
  WHEN expression_2 THEN result_expression_2
  …
  WHEN expression_n THEN result_expression_n
  [ELSE else_result_expression]
END
```

بررسی کد :

هنگام اجرای تابع، مقدار value ارزیابی می‌شود و در صورت تطبیق با هر یک از expression ها، result_expression متناظر برگردانده می‌شود.

در صورت عدم تطبیق با هیچ‌کدام از expression ها، عبارت else_result_expression برگردانده می‌شود که البته درج عبارت ...ELSE اجباری نیست.

مثال : پرس‌وجویی بنویسید که اطلاعات فاکتورها را استخراج نموده و به جای درج تاریخ فاکتور به صورت زیر عمل کند :

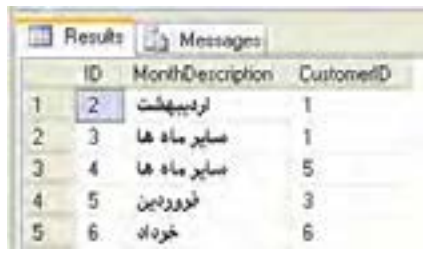
اگر فاکتور در فصل بهار صادر شـــده، نام فارسی ماه را بنویسد در غیر این‌صورت عبارت «سایر ماه‌ها» را قید کند.

```
SELECT ID, MonthDescription =
CASE SUBSTRING([DATE],6,2)
```

---

1 . Simple
2 . Searched

```
WHEN '01' THEN 'فروردین'
WHEN '02' THEN 'اردیبهشت'
WHEN '03' THEN 'خرداد'
ELSE 'سایر ماه ها'
END,
CustomerID
FROM Tbl_Factors
```



	ID	MonthDescription	CustomerID
1	2	اردیبهشت	1
2	3	سایر ماه ها	1
3	4	سایر ماه ها	5
4	5	فروردین	3
5	6	خرداد	6

بررسی کد :

تابع SUBSTRING(s , start , length) رشـــته s را می‌گیرد و از نویســـه‌ای که در آرگومان start مشخص شده به تعداد length نویسه جدا می‌کند. بنابراین:

خروجی مقدار SUBSTRING ('1388/02/15' , 6 , 2) رشـــته ’02‘ است. پیش از این توضیح داده شد که یکی از روش‌های موجود برای ذخیره‌سازی تاریخ‌های غیرمیلادی، نگه‌داری آن‌ها به صورت رشته است.

عبارت‌های WHEN بخش جدا شـــده از تاریخ را با مقادیر رشـــته‌ای مقایسه می‌کند و در صورت تطبیق، مقدار پس از THEN را برمی‌گرداند.

در صـــورت عدم تطبیق مقادیر موجود در شـــرط‌ها، مقدار موجود در عبـــارت ELSE برگردانده می‌شود.

ب) حالت جســـتجویی: در این روش که از انعطاف‌پذیری بیش‌تری برخوردار است، به جای استفاده از یک مقایسه ساده می‌توانیم در هر عبارت WHEN یک مقایسه داشته باشیم.

شکل کلی حالت جستجویی به صورت زیر است :

```
CASE
  WHEN boolean_expression_1 THEN result_expression_1
  WHEN boolean_expression_2 THEN result_expression_2
```

```
...
 WHEN boolean _expression_n THEN result_expression_n
 [ELSE else_result_expression]
END
```

بررسی کد :

هنگام اجرای تابع، هر یک از boolean_expression ها بررسی می‌شوند و در صورت درست بودن شرط مقدار result_expression متناظر برگردانده می‌شود.

در صورت عدم تطبیق با هیچ‌کدام از expression ها، عبارت else_result_expression برگردانده می‌شود و مجدداً تأکید می‌شود که درج عبارت ..ELSE اجباری نیست.

مثال : پرس‌وجویی بنویسد که اطلاعات کتاب‌ها را استخراج نموده و به جای قیمت کتاب‌ها عباراتی به صورت زیر درج کند :

قیمت بالای ۱۵۰۰۰۰: گران
قیمت بین ۸۰۰۰۰ تا ۱۵۰۰۰۰ : مناسب
قیمت زیر ۸۰۰۰۰ : ارزان

```
SQLQuery5.sql ...istrator (53))*
SELECT *, PriceEvaluation =
CASE
WHEN Price > 150000 THEN 'گران'
WHEN Price BETWEEN 80000 AND 150000 THEN 'مناسب'
WHEN Price < 80000 THEN 'ارزان'
END
FROM Tbl_Books
```

Results | Messages

	ID	Title	Author	Price	Pages	PriceEvaluation
1	1201	Access 2007	رضا اکبری	65000.00	200	ارزان
2	1202	طراحي صفحات وب	علی محمدی	160000.00	480	گران
3	1203	برنامه نويسي به زبان دلفي	منصور کبیری	80000.00	380	مناسب
4	1205	برنامه نويسي به زبان ويژوال بيسيك ۶	حامد قنبری	25000.00	100	ارزان
5	1206	برنامه نويسي به زبان C#	محمد محمدی	80000.00	420	مناسب

Evaluates a list of conditions and returns one of multiple possible result expressions.

CASE has two formats:

- The simple CASE function compares an expression to a set of simple expressions to determine the result.
- The searched CASE function evaluates a set of Boolean expressions to determine the result.

Both formats support an optional ELSE argument.

Syntax

Simple CASE function:

```
CASE input_expression
     WHEN when_expression THEN result_expression
    [ ...n ]
     [
    ELSE else_result_expression
     ]
END
```

Searched CASE function:

```
CASE
     WHEN Boolean_expression THEN result_expression
    [ ...n ]
```

```
    [
   ELSE else_result_expression
    ]
END
```

Arguments

input_expression

Is the expression evaluated when the simple CASE format is used. input_expression is any valid expression.

WHEN when_expression

Is a simple expression to which input_expression is compared when the simple CASE format is used. when_expression is any valid expression. The data types of input_expression and each when_expression must be the same or must be an implicit conversion.

n

Is a placeholder that indicates that multiple WHEN when_expression THEN result_expression clauses, or multiple WHEN Boolean_expression THEN result_expression clauses can be used.

THEN result_expression

Is the expression returned when input_expression equals when_expression evaluates to TRUE, or Boolean_expression evaluates to TRUE. result expression is any valid expression.

ELSE else_result_expression

Is the expression returned if no comparison operation evaluates to TRUE. If this argument is omitted and no comparison operation evaluates to TRUE, CASE returns NULL. else_result_expression is any valid expression. The data types of else_result_expression and any result_expression must be the same or must be an implicit conversion.

WHEN Boolean_expression

Is the Boolean expression evaluated when using the searched CASE format. Boolean_expression is any valid Boolean expression.

۱. متن بالا را مطالعه کرده و در مورد آن توضیح دهید.

۲. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. کتاب‌هایی که قیمت آن‌ها کم‌تر از ۵۰۰۰۰ ریال است، ارزان و بقیه گران محسوب می‌شوند. پرس‌وجویی بنویسید که نشـــان دهد پس از اعمال ۱۰% تخفیف روی قیمت کتاب‌ها، چند عنوان جزو کتاب‌های ارزان قرار می‌گیرند.

۲. پرس‌وجویی بنویسید که قیمت کتاب‌ها را پس از اعمال ۲۳% درصد تخفیف به صورت زیر گرد کند :

مقدار سه رقم سمت راست	روش گرد کردن
۵۰۰ و بیش‌تر	به سمت بالا
کمتر از ۵۰۰	به سمت پایین

۳. در پرسش‌نامه‌ای که حاوی ده سؤال بله-خیر است، پاسخ‌ها به صورت اعداد دودویی نگه‌داری می‌شوند. برای نمونه عدد ۱۱۰۱۱۱۱۱۱۱ نشان می‌دهد پاسخ همه پرسش‌ها به‌جز سؤال سوم، بله است. اگر بخواهیم با نوشـــتن یک پرس‌وجو، تعداد پاسخ‌های مثبت به سؤال پنجم را استخراج کنیم، شما چه روشی پیشنهاد می‌کنید؟

۴. بدون استفاده از توابع SQL، نام کتابی را قیمت آن از بقیه کتاب‌ها بیش‌تر است استخراج کنید.

۵. پرس‌وجویی بنویسید که نشان دهد در هر فاکتور، چند نسخه کتاب خریداری شده است.

۶. در پایگاه داده‌ای که تاریخ صدور فاکتور به صورت میلادی ذخیره می‌شود، میانگین فروش کتاب را در روزهای یکشنبه سال ۲۰۱۰ استخراج نمایید.

۷. پرس‌وجویی بنویسید که میزان فروش کتاب‌ها در نیمه اول سال ۸۹ را رتبه‌بندی نموده و نام کتاب و رتبه آن را نشان دهد.

۸. پرس‌وجویی بنویسید که شماره کتاب را همراه با یک ستون جدید با عنوان «میزان فروش» نشان داده و ستون جدید را به صورت زیر مقداردهی کند :

تعداد نسخه‌های فروخته شده	محتوای فیلد «میزان فروش»
۱۰۰۰ و بیش‌تر	پرفروش
بین ۵۰۰ و ۱۰۰۰	متوسط
۵۰۰ و کمتر	کم‌فروش

فصل نهم

فصل نهم : کار با Join

در مباحث مربوط به طراحی پایگاه داده به این نتیجه رسیدیم که برای ایجاد یک پایگاه داده کارآمد باید داده‌ها را درون چند جدول توزیع نماییم تا قواعد نرمال‌سازی به شکل مطلوبی رعایت شود. این کار باعث می‌شود هنگام بازیابی داده‌ها مجبور شویم داده‌های دو یا چند جدول را با هم ترکیب کنیم تا نتیجه موردنظر به دست آید.

این کار با استفاده از دستور JOIN یا الحاق انجام می‌شود و این فصل تماماً به بررسی این قابلیت پرکاربرد SQL Server اختصاص دارد.

۹-۱ الحاق متقاطع (CROSS JOIN)

الحاق کردن دو یا چند جدول در SQL Server که با استفاده از دستور JOIN انجام می‌شود از یک عملیات ریاضی به نام «ضرب دکارتی» تبعیت می‌کند. برای آشنایی با این مفهوم ریاضی فرض کنید دو مجموعه زیر را داریم :

$A=\{a_1 , a_2 , a_3\} , B=\{b_1 , b_2\}$

حاصل‌ضرب دکارتی A و B را به شکل تعریف می‌کنیم و بنابراین :

$A \times B = \{ (a_1,b_1) , (a_1,b_2) , (a_2,b_1) , (a_2,b_2) , (a_3,b_1) , (a_3,b_2)\}$

یعنی با استفاده از این عمل همه ترکیب‌های دوتایی از اعضای دو مجموعه ایجاد می‌شود به گونه‌ای که در زوج مرتب‌های تولید شده، عضو اول از مجموعه A و عضو دوم از مجموعه B است.

حال تصور کنید جدول A حاوی اطلاعات سه فاکتور و جدول B حاوی اطلاعات دو مشتری باشد.

A		
FactorID	CustomerID	Date
۱	۱۲	۱۳۸۸/۱۲/۱۲
۲	۱۴	۱۳۸۹/۰۵/۰۲
۳	۱۲	۱۳۸۸/۰۵/۱۶

B	
CustomerID	Name
۱۲	طلوع
۱۴	عابد

با اســـتفاده از دســـتور CROSS JOIN این دو جدول (رابطه) در هم ضرب دکارتی می‌شوند؛ یعنی هر رکورد از جدول A کنار هر رکورد از جدول B قرار می‌گیرد و مجموعاً شش رکورد جدید تشکیل می‌شود.

C				
A			B	
FactorID	CustomerID	Date	CustomerID	Name
۱	۱۲	۱۳۸۸/۱۲/۱۲	۱۲	طلوع
۱	۱۲	۱۳۸۸/۱۲/۱۲	۱۴	عابد
۲	۱۴	۱۳۸۹/۰۵/۰۲	۱۲	طلوع
۲	۱۴	۱۳۸۹/۰۵/۰۲	۱۴	عابد
۳	۱۲	۱۳۸۸/۰۵/۱۶	۱۲	طلوع
۳	۱۲	۱۳۸۸/۰۵/۱۶	۱۴	عابد

شـــاید این سؤال برای شـــما پیش بیاید که انجام این کار چه فایده‌ای دارد؟ اگر ادامه مطلب را با دقت بخوانید پاسخ پرسش خود را دریافت خواهید کرد.

در جدول C که از الحاق متقاطع دو جدول A و B به دست آمده، رکوردهایی را که در آن‌ها CustomerID جدول A و CustomerID جدول B برابر هستند جدا کنید.

C				
A			B	
FactorID	CustomerID	Date	CustomerID	Name
۱	۱۲	۱۳۸۸/۱۲/۱۲	۱۲	طلوع
۲	۱۴	۱۳۸۹/۰۵/۰۲	۱۴	عابد
۳	۱۲	۱۳۸۸/۰۵/۱۶	۱۲	طلوع

اکنون می‌توانید ببینید که اســـتفاده از الحاق چه مزیتی دارد. ما در جدول A فقط «شماره مشتری» را داشتیم اما با ترکیب آن با جدول B و ایجاد جدول C به «نام مشتری» هم دسترسی پیدا کرده‌ایم.

هنگام نرمال‌ســـازی جداول، داده‌هایی را که زیاد تکرار می‌شـــوند درون جدول مجزایی قرار می‌دهیم و در جدول اصلی فقط شناســـه آن‌ها را قید می‌کنیم. مثلاً در جدول مشـــتریان، استان محل سکونت مشتری با عدد ۱۰ مشـــخص می‌شود و در جدول اســـتان‌ها تعیین می‌کنیم که مثلاً منظور از عدد ۱۰ استان تهران است. بدون استفاده از عمل‌گرهای JOIN نمی‌توانیم گزارشی تهیه کنیم که نام شهر محل سکونت مشتری را بنویسد نه شناسه آن را.

حالت کلی نگارش یک الحاق متقاطع به صورت زیر است :

```
SELECT table_1.column_1,
   …
   table_1.column_n,
   table_2.column_1,
  …
   table_2.column_m,
FROM table 1 CROSS JOIN table_2
```

مثال ۱ : میان جداول فاکتورها و مشتریان یک الحاق متقاطع ایجاد کنید.

	شماره فاکتور	شماره مشتری-جدول فاکتورها	تاریخ فاکتور	شماره مشتری-جدول مشتریان	نام مشتری
18	3	1	1388/05/15	7	اداره ارشاد سپیدان
19	3	1	1388/05/15	8	فروشگاه محمدی
20	3	1	1388/05/15	9	کتابفروشی سازمان تبلیغات
21	3	1	1388/05/15	10	کتابسرای افق
22	3	1	1388/05/15	11	رضا
23	4	5	1388/06/11	1	انتشارات طلوع
24	4	5	1388/06/11	2	کتابسرای محمدی

مثال ۲ : پرس‌وجوی مثال قبل را با استفاده از نام‌های مستعار برای جداول خلاصه کنید.

وقتی عبارتی را پس از نام یک جدول می‌نویسیم، این عبارت تبدیل به نام مستعار جدول می‌شود و می‌توانیم در بخش‌هایی از پرس‌وجو، به جای نام واقعی جدول از این نام مستعار استفاده کنیم که روش خوبی برای خلاصه‌کردن کد و افزایش خوانایی آن است. در این مثال به جدول Tbl_Factors، نام مستعار f نسبت داده شده است. همچنین نام مستعار c برای جدول Tbl_Customers در نظر گرفته شده است.

```
SELECT f.ID AS 'شماره فاکتور',
    f.CustomerID AS 'شماره مشتری – جدول فاکتورها',
    f.[Date] AS 'تاریخ فاکتور',
    c.ID AS 'شماره مشتری – جدول مشتریان',
    c.Name AS 'نام مشتری'
FROM Tbl_Factors f CROSS JOIN Tbl_Customers c
ORDER BY f.ID
```

مثال ۳ : پرس‌وجوی مثال قبل را به گونه‌ای تغییر دهید که شماره فاکتور، نام مشتری و تاریخ فاکتور را نشان دهد.

	شماره فاکتور	تاریخ فاکتور	نام مشتری
1	2	1388/02/25	انتشارات طلوع
2	3	1388/05/15	انتشارات طلوع
3	4	1388/06/11	کتابفروشی امینی
4	5	1387/01/30	انتشارات امیرکبیر
5	6	1388/03/12	فروشگاه کتاب آزادی

در این کد، با استفاده از عبارت WHERE توانستیم عمل جداسازی رکوردهای موردنظر را انجام دهیم.

۹-۲ الحاق درونی (INNER)

رایج‌ترین نوع الحاق، INNER JOIN است و عمل‌کرد آن شباهت زیادی به الحاق متقاطعِ حاوی عبارت WHERE دارد. الحاق درونی فقط ردیف‌هایی را برمی‌گرداند که در شرط قید شده درون الحاق صدق کنند.

شکل کلی این الحاق به صورت زیر است :

```
SELECT columns_name FROM table_1 INNER JOIN table_2
ON conditions
```

- بعد از کلیدواژه ON می‌توانید یک یا چند شرط را برای جداسازی رکوردهای موردنظر قرار دهید.
- امکان ایجاد اتصال درونی بین بیش از دو جدول هم وجود دارد.

مثال ۱ : پرس‌وجویی بنویسید که با برقراری الحاق درونی میان جداول فاکتورها و مشتریان، شماره فاکتور، تاریخ فاکتور و نام مشتری را بازیابی کند.

```
SELECT f.ID AS 'شماره فاکتور',
    f.[Date] AS 'تاریخ فاکتور',
  c.Name AS 'نام مشتری'
FROM   Tbl_Factors f INNER JOIN Tbl_Customers c
ON f.CustomerID = c.ID
ORDER BY f.ID
```

مثال ۲ : پرس‌وجویی بنویسید که با برقراری الحاق درونی میان جداول جزییات فاکتورها و کتاب‌ها، نشان دهد که در هر ردیف فاکتور شماره ۵ چه کتاب‌هایی خریداری شده است.

```
SELECT b.Title
FROM   Tbl_Books b INNER JOIN Tbl_FactorDetails f
ON b.ID = f.BookID
WHERE f.FactorID = 5
```

بررسی کد :

در جدول کتاب‌ها، هر کتاب دارای شماره (ID) و عنوان (Title) است.

در جدول جزییات فروش، در هر رکورد شـــماره فاکتور (FactorID) ، شماره کتاب (BookID) و ... درج شده است.

الحاق درونی میان دو جدول با شرط برابر بودن شماره کتاب در جدول کتاب‌ها با شماره کتاب در جدول جزییات فروش اجرا می‌شود.

نهایتاً با استفاده از عبارت WHERE، رکوردهایی جدا می‌شوند که شماره فاکتور آن‌ها برابر ۵ است.

مثال ۳ : پرس‌وجویی بنویسید که نشان دهد هر کتاب توسط چه مشتریانی خریداری شده است.

مطابق تصویر زیر، پرس‌وجوی موردنظر باید روی چهار جدول پایگاه داده اجرا شـــود تا نتیجه نهایی به دست آید زیرا :

- شماره کتاب‌های فروخته شده در جدول جزییات فاکتور قرار دارد.
- نام کتاب‌های فروخته شده در جدول کتاب‌ها نگه‌داری می‌شود.
- شماره مشتری از طریق جدول فاکتورها قابل بازیابی است.
- نام مشتری در جدول مشتریان ذخیره شده است.

به همین دلیل باید سه بار عمل الحاق درونی صورت گیرد.

```
SELECT  b.Title, c.Name
FROM    Tbl_FactorDetails fd INNER JOIN
        Tbl_Books b ON fd.BookID =b.ID INNER JOIN
        Tbl_Factors f ON fd.FactorID = f.ID INNER JOIN
        Tbl_Customers c ON f.CustomerID = c.ID
ORDER BY b.Title
```

مثال ۴ : پرس‌وجویی مثال قبل را طوری تغییر دهید که این اطلاعات فقط برای مشتریان ساکن استان تهران نشان داده شود.

```
SELECT  b.Title, c.Name
FROM    Tbl_FactorDetails fd INNER JOIN
        Tbl_Books b ON fd.BookID =b.ID INNER JOIN
        Tbl_Factors f ON fd.FactorID = f.ID INNER JOIN
        Tbl_Customers c ON f.CustomerID = c.ID AND c.Province='تهران'
ORDER BY b.Title
```

مثال ۵ : شماره کتاب‌هایی را پیدا کنید که در بهار ۸۸ فروش داشته‌اند.

```
SELECT DISTINCT fd.BookID
FROM   dbo.Tbl_FactorDetails fd INNER JOIN
    dbo.Tbl_Factors f ON fd.FactorID = f.ID
WHERE  f.Date BETWEEN '1388/01/01' AND '1388/03/31'
```

بررسی کد :

تاریخ فاکتورها در جدول Tbl_Factors و جزییات فروش کتاب‌ها در جدول Tbl_FactorDetails نگه‌داری می‌شود بنابراین باید این دو جدول را با شرط برابر بودن شماره فاکتورها الحاق درونی بزنیم.

با استفاده از عبارت WHERE باید فروش‌هایی را که در فصل بهار بوده از آن‌ها جدا کنیم.

عبارت DISTINIC هم از تکرار شماره‌های کتاب جلوگیری می‌کند.

مثال ۶ : نام کتاب‌هایی را پیدا کنید که در بهار ۸۸ فروش نداشته‌اند.

در مثال قبل شماره کتاب‌هایی را که در این بازه فروش داشته‌اند پیدا کردیم، حال می‌توانیم پرس‌وجویی را روی پرس‌وجوی قبلی بنویسیم تا به نتیجه دلخواه برسیم.

```
SELECT DISTINCT Title FROM Tbl_Books
WHERE ID NOT IN
(
SELECT DISTINCT fd.BookID
FROM   dbo.Tbl_FactorDetails fd INNER JOIN
    dbo.Tbl_Factors f ON fd.FactorID = f.ID
WHERE  f.Date BETWEEN '1388/01/01' AND '1388/03/31'
)
```

بررسی کد :

دستور انتخاب که درون پرانتز قرار گرفته همان مثال قبل است و شماره کتاب‌هایی را برمی‌گرداند که در بهار ۸۸ فروش داشته‌اند.

یک عبارت انتخاب جدید روی این نتایج می‌نویسیم تا عنوان کتاب‌هایی را استخراج کند که شماره آن‌ها در نتایج پرس‌وجوی دوم وجود ندارد. عبارت NOT IN وجود نداشتن یک مقدار را در مجموعه‌ای از مقادیر بررسی می‌کند.

۳-۹ الحاق بیرونی (OUTER)

گاهی اوقات لازم است پرس‌وجو را اجرا کنیم که صرف‌نظر از انطباق رکوردها در جداول موجود در الحاق، تمام رکوردهای یک یا چند جدول را برگرداند.

دو جدول مشتریان و فاکتورها را درنظر بگیرید. چنان‌چه پرس‌وجویی از نوع الحاق درونی را روی آن‌ها اجرا کنیم، تنها نام آن دسته از مشتریان جزو نتایج قرار می‌گیرد که حداقل یک فاکتور برای آن‌ها صادر شده باشد.

```
SQLQuery2.sql ...istrator (53))*
SELECT c.ID, c.Name,f.ID, f.CustomerID, f.[Date]
FROM Tbl_Customers c INNER JOIN Tbl_Factors f
ON c.ID = f.CustomerID
```

Results | Messages

	ID	Name	ID	CustomerID	Date
1	1	انتشارات طلوع	2	1	1388/02/25
2	8	فروشگاه محمدي	3	8	1388/05/15
3	5	كتابفروشي اميني	4	5	1388/06/11
4	3	انتشارات اميركبير	5	3	1387/01/30
5	6	فروشگاه كتاب آزادي	6	6	1388/03/12
6	4	كتابفروشي انديشه	8	4	1388/03/15
7	3	انتشارات اميركبير	9	3	1388/02/11
8	4	كتابفروشي انديشه	11	4	1388/01/25

اگر بخواهیم در این پرس‌وجو نام مشـــتریانی هم که هیچ فاکتوری برای آن‌ها صادر نشـــده وجود داشته باشـــد، راه‌حل چیست؟ این کار با استفاده از الحاق بیرونی انجام می‌گیرد. سه نوع الحاق بیرونی وجود دارد که عبارتند از :

الف) الحاق بیرونی چپ[1]

ب) الحاق بیرونی راست[2]

ج) الحاق بیرونی کامل[3]

در الحاق بیرونی چپ که با عبارت LEFT OUTER JOIN یا به شـــکل مختصرتر LEFT JOIN نوشته می‌شود، همه سطرهای جدول سمت چپ الحاق در نتایج ظاهر می‌شوند و فیلدهایی از جدول سمت راست الحاق که مقداری برای آن‌ها وجود ندارد با NULL مقداردهی می‌شوند.

برای روشن‌تر شدن مسأله، در پرس‌وجویی که ابتدای این بخش توضیح دادیم، الحاق درونی را به الحاق بیرونی چپ تبدیل و آن را اجرا کنید تا نتیجه صفحه بعد حاصل شود.

---

1 . LEFT OUTER JOIN
2 . RIGHT OUTER JOIN
3 . FULL OUTER JOIN

	ID	Name	ID	CustomerID	Date
1	1	انتشارات طلوع	2	1	1388/02/25
2	2	کتابسرای محمدی	NULL	NULL	NULL
3	3	انتشارات امیرکبیر	5	3	1387/01/30
4	3	انتشارات امیرکبیر	9	3	1388/02/11
5	4	کتابفروشی اندیشه	8	4	1388/03/15
6	4	کتابفروشی اندیشه	11	4	1388/01/25
7	5	کتابفروشی امیدی	4	5	1388/06/11
8	6	فروشگاه کتاب آزادی	6	6	1388/03/12
9	7	اداره ارشاد سپیدان	NULL	NULL	NULL
10	8	فروشگاه محمدی	3	8	1388/05/15
11	9	کتابفروشی سازمان تبلیغات	NULL	NULL	NULL
12	10	کتابسرای افق	NULL	NULL	NULL
13	11	رسا	NULL	NULL	NULL

همان‌طور که می‌بینید تمامی رکوردهای موجود در جدول سمت چپ الحاق (یعنی Tbl_Customers) در نتیجه نهایی وجود دارند. از آن‌جا که برای تعدادی از مشتری‌ها هیچ فاکتوری صادر نشده و رکورد متناظری در جدول سمت راست الحاق (یعنی Tbl_Factors) وجود ندارد، فیلدهای رکورد فاکتور با NULL مقداردهی شده است.

اگر در پرس‌وجوی قبل، محل قرارگیری دو جدول را نسبت به عبارت الحاق تغییر دهیم و در عوض از الحاق خارجی راست استفاده کنیم، چه تغییری در نتیجه پرس‌وجو ایجاد می‌شود؟

```
SELECT c.ID, c.Name, f.ID, f.CustomerID, f.[Date]
FROM  Tbl_Factors f RIGHT OUTER JOIN Tbl_Customers c
ON c.ID = f.CustomerID
```

هیچ تغییری ایجاد نمی‌شود زیرا در الحاق بیرونی راست، همه رکوردهای موجود در جدول سمت راست الحاق (یعنی Tbl_Customers) برگردانده می‌شود و به جای فیلدهایی از جدول سمت چپ الحاق که مقدار متناظری برای آن‌ها وجود ندارد، NULL قرار داده می‌شود.

در پایگاه داده‌ای که از ابتدای کتاب با آن کار کرده‌ایم، بین جدول فاکتورها و مشتریان یک رابطه برقرار نمودیم. درهنگام ایجاد این رابطه، گزینه Enforce Foreign Key Constraint را روی Yes تنظیم کردیم تا اجازه ورود فاکتورهایی را ندهد که شماره مشتری آن‌ها نامعتبر است.

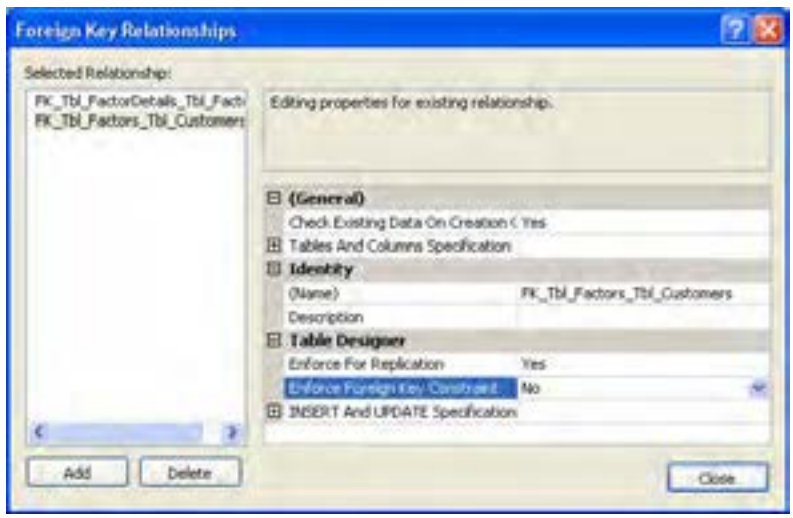
برای مشـــاهده اثر الحاق خارجی کامل، ابتدا به روش زیر محدودیت اعمال شـــده را بردارید و شـــماره مشتری برخی فاکتورها را به شماره‌هایی تغییر دهید که در جدول مشتریان وجود ندارند.

۱. پوشه Tables را باز نموده و به سراغ جدول Tbl_Factors بروید.

۲. پوشه Keys را باز نموده و روی کلید خارجی FK_Tbl_Factors_Tbl_Customers راست‌کلیک کنید.

۳. گزینه Modify را انتخاب نمایید.

۴. در پنجره‌ای که ظاهر می‌شود گزینه Enforce Foreign Key Constraint را روی No تنظیم کنید.



۵. روی دکمه Close کلیک و سپس تغییرات را ذخیره کنید.

۶. جدول فاکتورها را در حالت ویرایش باز نمایید.

۷. در تعدادی از رکوردها، شماره مشتریان را به شماره‌ای تغییر دهید که در جدول مشتریان وجود ندارد.

ID	CustomerID	Date
2	1	1388/02/25
3	8	1388/05/15
4	5	1388/06/11
5	13	1387/01/30
6	10	1388/03/12
8	4	1388/03/15
9	3	1388/02/11
11	14	1388/01/25
12	10	1388/09/12
NULL	NULL	NULL

ID	Name
1	انتشارات طلوع
2	کتابسرای محمدی
3	انتشارات امیرکبیر
4	کتابفروشی اندیشه
5	کتابفروشی امینی
6	فروشگاه کتاب آزادی
7	اداره ارشاد سپیدان
8	فروشگاه محمدی
9	کتابفروشی سازمان...
10	کتابسرای افق
11	رسا
NULL	NULL

۸. پرس‌وجوی زیر را در محیط Management Studio نوشته و اجرا کنید.

```
SQLQuery2.sql ...istrator (53))*
SELECT c.ID, c.Name, f.ID, f.CustomerID, f.[Date]
FROM  Tbl_Customers c FULL OUTER JOIN Tbl_Factors f
ON c.ID = f.CustomerID
```

Results | Messages

ID	Name	ID	CustomerID	Date
1	انتشارات طلوع	2	1	1388/02/25
2	کتابسرای محمدی	NULL	NULL	NULL
3	انتشارات امیرکبیر	9	3	1388/02/11
4	کتابفروشی اندیشه	8	4	1388/03/15
5	کتابفروشی امینی	4	5	1388/06/11
6	فروشگاه کتاب آزادی	NULL	NULL	NULL
7	اداره ارشاد سپیدان	NULL	NULL	NULL
8	فروشگاه محمدی	3	8	1388/05/15
9	کتابفروشی سازمان تبلیغات	NULL	NULL	NULL
10	کتابسرای افق	6	10	1388/03/12
10	کتابسرای افق	12	10	1388/09/12
11	رسا	NULL	NULL	NULL
NULL	NULL	5	13	1387/01/30

همان‌طور که مشـــاهده می‌کنید، در الحاق بیرونی کامل، همه رکوردها از جداول ســـمت چپ و راست الحاق در نتیجه ظاهر می‌شـــوند و هر جا رکورد متناظری از جدول دیگری وجود نداشـــته باشـــد، NULL گذاشته می‌شود.

برای نمونه در پرس‌وجوی قبل، برای دو فاکتور آخر، مشـــتری با شـــماره ۱۳ یا ۱۴ وجود ندارد بنابراین مقدار NULL به جای مشـــخصات مشتری قرار می‌گیرد. به همین ترتیب برای مشتریانی که هیچ فاکتوری برای آن‌ها صادر نشده، در فیلدهای جدول فاکتورها مقدار NULL نمایش داده می‌شود.

مثال : پرس‌وجوی زیر چه نتایجی را برمی‌گرداند؟

```
SELECT c.ID, c.Name,f.ID, f.CustomerID, f.[Date]
FROM  Tbl_Customers c RIGHT JOIN Tbl_Factors f
ON c.ID = f.CustomerID
```

از آن‌جا که الحاق بیرونی راســـت اســـتفاده شـــده و در سمت راســـت این الحاق، جدول Tbl_Factors قرار دارد همه فاکتورهای موجود برگردانده می‌شـــوند و چنان‌چه شـــماره مشتریِ متناظر در جدول Tbl_Customers وجود نداشته باشد، فیلدها با NULL مقداردهی می‌شوند.

Results | Messages

	ID	Name	ID	CustomerID	Date
1	1	انتشارات طلوع	2	1	1388/02/25
2	8	فروشگاه محمدی	3	8	1388/05/15
3	5	کتابفروشی امینی	4	5	1388/06/11
4	NULL	NULL	5	13	1387/01/30
5	10	کتابسرای افق	6	10	1388/03/12
6	4	کتابفروشی اندیشه	8	4	1388/03/15
7	3	انتشارات امیرکبیر	9	3	1388/02/11
8	NULL	NULL	11	14	1388/01/25
9	10	کتابسرای افق	12	10	1388/09/12

CROSS JOIN

A cross join that does not have a WHERE clause produces the Cartesian product of the tables involved in the join. The size of a Cartesian product result set is the number of rows in the first table multiplied by the number of rows in the second table.

INNER JOIN

An inner join is a join in which the values in the columns being joined are compared using a comparison operator.

In the ISO standard, inner joins can be specified in either the FROM or WHERE clause. This is the only type of join that ISO supports in the WHERE clause. Inner joins specified in the WHERE clause are known as old-style inner joins.

OUTER JOIN

Inner joins return rows only when there is at least one row from both tables that matches the join condition. Inner joins eliminate the rows that do not match with a row from the other table. Outer joins, however, return all rows from at least one of the tables or views mentioned in the FROM clause, as long as those rows meet any WHERE or HAVING search conditions.

All rows are retrieved from the left table referenced with a left outer join, and all rows from the right table referenced in a right outer join. All rows from both tables are returned in a full outer join.

SQL Server uses the following ISO keywords for outer joins specified in a FROM clause:

- LEFT OUTER JOIN or LEFT JOIN
- RIGHT OUTER JOIN or RIGHT JOIN
- FULL OUTER JOIN or FULL JOIN

۱. متن فوق را مطالعه کرده و در مورد انواع الحاق توضیح دهید.
۲. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. انواع الحاق‌ها و تفاوت میان آن‌ها را توضیح دهید.

۲. پرس‌وجویی بنویســـید که عنوان و نام نویسندگان کتاب‌هایی را که در پاییز ۸۸ بیش‌تر از ۲۰۰ نسخه از آن‌ها فروخته شده برگرداند.

۳. نام مشتریانی را به دست آورید که فاکتورهایی با مجموع خرید بیش‌تر از ۱۰۰۰۰۰۰ ریال دارند.

۴. پرس‌وجوی زیر چه مقادیری را برمی‌گرداند؟

```
SELECT DISTINCT(b.ID) FROM Tbl_Books b
RIGHT OUTER JOIN Tbl_FactorDetails fd
ON b.ID=fd.BookID
```

اگر از الحاق بیرونی چپ استفاده شود، چه تغییری در نتایج خروجی ایجاد خواهد شد؟

فصل دهم

فصل دهم : کار با زیرپرس‌وجوها[1]

زیرپرس‌وجو در واقع یک پرس‌وجو اســت که درون یک عبــارت SELECT، INSERT، UPDATE یا DELETE و یا درون یک زیرپرس‌وجوی دیگر نوشته می‌شود و توانایی پرس‌وجو را برای انجام کار موردنظر، به نحو چشم‌گیری افزایش می‌دهد.

در مثال‌های فصول قبل بدون آن‌که نامی از زیرپرس‌وجوها برده شود، با کارکرد آن‌ها آشنا شدید اما در این فصل قصد داریم انواع زیرپرس‌وجوها را به صورت دسته‌بندی شده و با جزییات بیش‌تری بررسی کنیم.

۱- ۱۰ زیرپرس‌وجوهای تک‌مقداری[2]

زیرپرس‌وجوهای تک‌مقداری یا اســکالر یکی از انواع زیرپرس‌وجوها به شمار می‌روند که فقط یک مقدار را برمی‌گردانند و می‌توانند در عبارت‌هایی مانند SELECT ، WHERE و CASE مورداســتفاده قرار گیرند. مقــدار حاصل از اجرای این زیرپرس‌وجوها که معمولاً توســط یک تابع تجمعی (مانند جمع، میانگین و ...) برگردانده می‌شود حاوی یک ردیف و یک ستون است. به این ترتیب می‌توانید با ایجاد پرس‌وجوهای تودرتو[3] از همه قابلیت‌های SQL Server جهت ایجاد کدهای پیچیده و بازیابی نتایج دلخواه بهره ببرید.

در ادامه نمونه‌هایی از کاربرد زیرپرس‌وجوهای تک مقداری را بررســی خواهیم کرد. توجه داشته باشید که زیرپرس‌وجوها باید درون پرانتز قرار داده شوند تا از پرس‌وجوی اصلی متمایز گردند.

---

1 . Subqueries

2 . Scalar or Single Value

3 . Nested

الف) عبارت‌های SELECT: با اســـتفاده از یک زیرپرس‌وجـــوی Scalar درون یک عبارت SELECT می‌توانید فیلدهای موردنظر از یک یا چند جدول را اســـتخراج نموده و حالت الحاق جداول را پیاده‌ســـازی کنید.

مثال ۱: پرس‌وجویی بنویســـید که عنوان، نام نویســـنده و قیمت هر کتاب را اســـتخراج نموده و میانگین قیمت کتاب‌هایی با این عنوان را محاسبه کند.

```
SQLQuery1.sql ...istrator (53))*
SELECT Title, Author, Price ,
 (SELECT AVG(Price)FROM Tbl_Books b WHERE Title=b.Title) AS 'میانگین قیمت کتاب'
FROM Tbl_Books
ORDER BY Title
```

Results | Messages

	Title	Author	Price	میانگین قیمت کتاب
1	Access 2007	رضا اکبری	65000.00	71000.00
2	Access 2007	مهدی رضایی	83000.00	71000.00
3	Access 2007	رضا باقری	65000.00	71000.00
4	CorelDraw	سعید برزگر	65000.00	65000.00
5	Dreamweaver	معصومه نظری	120000.00	120000.00
6	Excel 2007	علی اکبری	200000.00	200000.00

مثال ۲: پرس‌وجویی بنویســـید که شـــماره فاکتور و تاریخ صدور را از جدول فاکتورها استخراج و همراه با نام مشتری نمایش دهد. توجه داشته باشید که نام مشتری در جدول مشتریان نگه‌داری می‌شود و در جدول فاکتورها صرفاً شماره مشتری موجود است.

```
SQLQuery1.sql ...istrator (53))*
SELECT f.ID, f.[DATE],
 (SELECT name FROM Tbl_Customers c WHERE c.ID=f.CustomerID) AS 'نام مشتری'
FROM Tbl_Factors f
```

Results | Messages

	ID	DATE	نام مشتری
1	2	1388/02/25	انتشارات طلوع
2	3	1388/05/15	فروشگاه محمدی
3	4	1388/06/11	کتابفروشی امینی

ب) عبارت‌های WHERE: گاهی اوقات لازم می‌شـــود نتیجه زیرپرس‌وجو در قسمت شرط یک عبارت دیگر مورد استفاده قرار گیرد.

مثال ۳: برای استخراج مشخصات کتاب‌هایی که قیمت آن‌ها از میانگین قیمت کتاب‌های موجود در جدول پایین‌تر است یک پرس‌وجو طراحی کنید.

```
SELECT * FROM Tbl_Books
WHERE Price < (SELECT AVG(Price)FROM Tbl_Books)
```

ج) عبارت‌های CASE: در درس‌های قبل با عملکرد عبارت CASE آشنا شدید. این عبارت یک مقدار را مورد ارزیابی قرار می‌دهد و بســـته به نتیجه ارزیابی، مقدار مشخص شده را برمی‌گرداند. با استفاده از این عبارت به عنوان یک زیرپرس‌وجوی Scalar می‌توانید با بررسی یک شرط، مقدار دلخواه را در ستون جدیدی درج نمایید.

مثال ۴: پرس‌وجویی بنویسد که نام کتاب‌ها و قیمت آن‌ها را استخراج نموده و اگر حاصل تقسیم قیمت بر تعداد صفحات بیش‌تر از ۲۳۰ ریال بود، کتاب را «گران» و در غیراین‌صورت «ارزان» اعلام کند.

۲- ۱۰ زیرپرس‌وجوهای چندمقداری[1]

زیرپرس‌وجوهای Multi-Valued برخلاف زیرپرس‌وجوهای Scalar می‌توانند چندین مقدار را برگردانند. غالباً این مقادیر باید توسط یک عمل‌گر مورد بررسی قرار گیرند تا نتیجه نهایی حاصل شود. زیرپرس‌وجوهای چند مقداری می‌توانند با عمل‌گرهای IN، ALL/ANY و یا EXISTS به کار روند. در ادامه مروری مختصر بر ترکیب این نوع پرس‌وجوها با عمل‌گرهای فوق خواهیم کرد. توجه داشته باشید که بسیاری از پرس‌وجوهای حاوی Multi-Value Subquery را می‌توان به‌وسیله الحاق درونی هم پیاده‌سازی نمود.

الف) استفاده از IN : با اســـتفاده از عمل‌گر IN و یک زیرپرس‌وجوی چندمقداری می‌توانید وجود یک مقدار را در مجموعه‌ای از مقادیر که توســـط زیرپرس‌وجو برگردانده می‌شـــوند بررسی کنید. شکل کلی این عبارت به صورت زیر است :

```
SELECT … FROM … WHERE expression IN (SELECT … FROM …)
```

مثال ۱ : نام کتاب‌هایی را استخراج کنید که حداقل یک‌بار با تخفیف بیش‌تر از ۳۰ درصد فروخته شده‌اند.

```
SELECT b.Title FROM Tbl_Books b
WHERE b.ID IN
(SELECT BookID FROM Tbl_FactorDetails fd WHERE Discount>30)
```

بررسی کد :

زیرپرس‌جویی که درون پرانتز قرار گرفته، شماره کتاب‌هایی را از جدول جزییات فاکتور استخراج می‌کند که حداقل یک بار با تخفیف بیش‌تر از ۳۰ درصد فروخته شده‌اند.

پرس‌وجوی اصلی، عنوان کتاب‌هایی را به دســـت می‌آورد که شماره آن‌ها در این مجموعه موجود است.

مثال ۲ : مثال قبـــل را به گونه‌ای تغییر دهید تا نام کتاب‌هایی را برگرداند که هیچ‌گاه با تخفیف بیش‌تر از ۳۰ درصد فروخته نشده‌اند.

```
SELECT b.Title FROM Tbl_Books b
WHERE b.ID NOT IN
(SELECT BookID FROM Tbl_FactorDetails fd WHERE Discount>30)
```

---

1 . Multi-Value

مثال ۱: با استفاده از عمل‌گر IN، نام مشتریانی را به دست آورید که حداقل در یکی از خریدهای خود تخفیف بیش از ۳۰ درصد دریافت کرده‌اند.

```
SELECT Name FROM Tbl_Customers c WHERE c.ID IN
(SELECT CustomerID FROM Tbl_Factors f
WHERE f.ID IN
(SELECT fd.FactorID FROM Tbl_FactorDetails fd WHERE Discount>30))
```

بررسی کد:

زیرپرس‌وجوی دوم، شماره فاکتورهایی را استخراج می‌کند که در آن‌ها خریدی با تخفیف بیش‌تر از ۳۰ درصد انجام شده باشد.

زیرپرس‌وجوی اول، شماره مشتریانی را به دست می‌آورد که فاکتوری با یکی از شماره‌های حاصل از اجرای زیرپرس‌وجوی اول در جدول فاکتورها دارند.

پرس‌وجوی اصلی، نام مشتریانی را برمی‌گرداند که شماره آن‌ها در نتیجه حاصل از اجرای زیرپرس‌وجوی اول وجود دارد.

ب) استفاده از عمل‌گرهای ALL/ANY: این دو عمل‌گر یک مقدار را با مجموعه‌ای از مقادیر که در قالب یک ستون توسط زیرپرس‌وجو برگردانده شده‌اند مقایسه می‌کند. برای مقایسه مقادیر می‌توانید از همه عمل‌گرهای مقایسه‌ای مانند > ، < ، >= ، <= و ... استفاده کنید. برای مثال عبارت‌های زیر را در نظر بگیرید:

```
n > ANY (1,5,14)
n > ALL (1,5,14)
```

مقدار عبارت اول صحیح است اگر n حداقل از یکی از مقادیر درون پرانتز بزرگ‌تر باشد بنابراین اگر به جای N عدد ۴ قرار گیرد، مقدار عبارت صحیح خواهد بود. دقت داشته باشید که به جای ANY می‌توانید از SOME هم استفاده کنید.

مقدار عبارت دوم در صورتی صحیح است که n از همه مقادیر درون پرانتز بزرگ‌تر باشد.

مثال ۱: درستی یا نادرستی عبارت‌های زیر را بررسی کنید.

```
4 = ANY (1,2,3)
4 <> ALL (1,2,3)
```

عبارت اول نادرســـت اســـت چون عدد ۴ با هیچ‌یک از مقادیر درون پرانتز برابر نیست. اما عبارت دوم به همین دلیل که ذکر شد صحیح است.


مثال ۲ : پرس‌وجوی زیر چه کاری انجام می‌دهد؟ آن را با استفاده از عمل‌گر IN بازنویسی کنید.

```
SELECT b.Title FROM Tbl_Books b
WHERE b.ID = ANY
(SELECT BookID FROM Tbl_FactorDetails fd WHERE fd.Discount>25)
```

این پرس‌وجو عنوان کتاب‌هایی را برمی‌گرداند که با تخفیف حداقل یک‌بار با تخفیف بیش‌تر از ۲۵ درصد فروخته شده‌اند. استفاده از عمل‌گر IN پرس‌وجو را به صورت زیر درمی‌آورد.

```
SELECT b.Title FROM Tbl_Books b
WHERE b.ID IN
(SELECT BookID FROM Tbl_FactorDetails fd WHERE fd.Discount>25)
```


مثال ۳ : پرس‌وجوی زیر چه نتایجی برمی‌گرداند؟ آن را با استفاده از عمل‌گر IN بازنویسی کنید.

```
SELECT b.Title FROM Tbl_Books b
WHERE b.ID <> ALL
(SELECT BookID FROM Tbl_FactorDetails fd WHERE fd.Discount>25)
```

این پرس‌وجو عنوان کتاب‌هایی را اســـتخراج می‌کند که تاکنون با تخفیف بیش‌تر از ۲۵ درصد فروخته نشده‌اند. این پرس‌وجو را می‌توان به صورت زیر بازنویسی نمود.

```
SELECT b.Title FROM Tbl_Books b
WHERE b.ID NOT IN
(SELECT BookID FROM Tbl_FactorDetails fd WHERE fd.Discount>25)
```


مثـــال ۴ : فرض کنید در جدول جزییات فاکتورها، کتاب‌های شـــماره ۱۲۱۰، ۱۲۱۵ و ۱۲۰۸ با تخفیف بیش‌تر از ۴۵ درصد فروخته شده‌اند. با این فرض، پرس‌وجوی زیر چه مقادیری را برمی‌گرداند؟

```
SELECT b.Title FROM Tbl_Books b
WHERE b.ID > ANY
(SELECT DISTINCT(BookID) FROM Tbl_FactorDetails WHERE Discount>=45)
```

زیرپرس‌وجـــو، جدول زیر را برمی‌گردانـــد، بنابراین پرس‌وجوی اصلی، عناوین کتاب‌هایی را اســـتخراج می‌کند که شـــماره آن‌ها از حداقل یکی از شماره‌های زیر بیش‌تر باشد. بنابراین عنوان کتاب‌هایی با شماره بیش‌تر از ۱۲۰۸ برگردانده می‌شود.

1208
1210
1215

ج) استفاده از EXISTS: این عمل‌گر بررسی می‌کند که آیا یک زیرپرس‌وجو ردیفی را برگردانده است یا خیر و شکل کلی آن به صورت زیر است.

```
WHERE [NOT] EXISTS (Subquery)
```

مثال ۱: شـــماره و نام مشتریانی را اســـتخراج کنید که حداقل یک فاکتور برای آن‌ها صادر شده است.

```
SELECT c.ID,c.Name FROM Tbl_Customers c
WHERE EXISTS
(SELECT * FROM Tbl_Factors f WHERE f.CustomerID=c.ID)
```

مثال ۲: پرس‌وجویی بنویسید تا مشتریانی را که هیچ فاکتوری برای آن‌ها صادر نشده حذف کند.

```
DELETE FROM Tbl_Customers
WHERE NOT EXISTS
(SELECT * FROM Tbl_Factors WHERE
Tbl_Factors.CustomerID=Tbl_Customers.ID)
```

مثال ۳: پرس‌وجوی زیر چه نتایجی را برمی‌گرداند؟ آن را با اســـتفاده از عمل‌گر IN بازنویســـی کنید.

```
SELECT DISTINCT(b.Title) FROM Tbl_Books b
WHERE EXISTS
(SELECT * FROM Tbl_FactorDetails fd
 WHERE fd.BookID=b.ID AND Discount>40)
```

عنـــوان کتاب‌هایی را بازیابی می‌کند که حداقل یک‌بار با تخفیف بیش‌تر از ۴۰ درصد فروخته شـــده‌اند. روش پیاده‌سازی این پرس‌وجو با استفاده از عمل‌گر IN به صورت زیر است.

```
SELECT DISTINCT(b.Title) FROM Tbl_Books b
WHERE b.ID IN
(SELECT fd.BookID FROM Tbl_FactorDetails fd WHERE Discount>40)
```

شـــاید با مشاهده مثال‌های فوق، این پرسش برای شـــما پیش بیاید که وقتی پیاده‌سازی یک پرس‌وجو با اســـتفاده از چند عمل‌گر (مثلاً IN، ANY/ALL و EXISTS) یا دستورات JOIN امکان‌پذیر است، کاربرد کدام‌یک اولویت دارد؟ این ســـؤال پاسخ مفصلی دارد اما اگر بخواهیم پاسخی در حد و اندازه این کتاب ارایه کنیـــم، باید بگوییم که در پایـــگاه داده‌هایی که حجم اطلاعات بالایی دارند، ســـرعت اجرای یک پرس‌وجو فوق‌العاده مهم است و تفاوت این دستورات و عمل‌گرها در سرعت اجرای پرس‌وجو نهفته است. غالباً JOIN نسبت به زیرپرس‌وجو سریع‌تر عمل می‌کند و ضمناً کارایی EXISTS از IN و ANY/ALL بیش‌تر است.

۳- ۱۰ زیرپرس‌وجوهای مستقل و وابسته[1]

در دو بخش قبل زیرپرس‌وجوها را بســـته به تعداد ردیف‌هایی که برمی‌گردانند به دو دسته تک‌مقداری و چندمقداری تقســـیم‌بندی کردیم. حال می‌خواهیم دسته‌بندی جدید را معرفی کنیم که به وابستگی یک زیرپرس‌وجو به پرس‌وجوی اصلی برمی‌گردد. از این منظر زیرپرس‌وجوها به دو دســـته مســـتقل و وابسته[2] تقسیم‌بندی می‌شوند.

در مثال‌های قبلی مشـــاهده کردید که یـــک زیرپرس‌وجو، مقادیری را بـــرای پرس‌وجوی اصلی فراهم می‌آورد. اگر زیرپرس‌وجو برای بازیابی مقادیر، وابسته به پرس‌وجوی اصلی نباشد «مستقل» نامیده می‌شود اما چنان‌چه به ردیفی از پرس‌وجوی اصلی رجوع داشته باشد یک زیرپرس‌وجوی وابسته است.

به دو پرس‌وجوی زیر توجه کنید.

نمونه اول :

```
SELECT fd.FactorID FROM Tbl_FactorDetails fd
WHERE fd.Discount=
(SELECT MAX(Discount) FROM Tbl_FactorDetails fd)
```

1 . Self-Contained
2 . Correlated

```
----------
```

نمونه دوم :

```
SELECT * FROM Tbl_FactorDetails fd1
WHERE fd1.Discount=
(SELECT MAX(Discount) FROM Tbl_FactorDetails fd2
WHERE fd1.FactorID=fd2.FactorID)
```

پرس‌وجوی اول شـــماره فاکتورهایی را برمی‌گرداند که میزان تخفیف در آن‌ها، بیش‌ترین تخفیفی است که در جدول جزییات فروش وجود دارد. همان‌طور که مشـــاهده می‌کنید، زیرپرس‌وجوی درون پرانتز فقط یک‌بار اجرا می‌شـــود و بیش‌ترین مقدار تخفیف را در اختیار پرس‌وجـــوی اصلی قرار می‌دهد؛ بنابراین یک زیرپرس‌وجوی Self-Contained است.

اما در مورد نمونه دوم وضع کمی متفاوت است؛ این پرس‌وجو بررسی می‌کند که در هر فاکتور بیش‌ترین مقدار تخفیف چه عددی اســـت. در این حالت زیرپرس‌وجو باید یک شماره فاکتور را دریافت نموده و هربار بیش‌ترین مقدار تخفیف را در میان رکوردهای آن فاکتور پیدا کند. بنابراین زیرپرس‌وجو نمی‌تواند مســـتقل از پرس‌وجوی اصلی اجرا گردد و لذا یک زیرپرس‌وجوی Correlated یا وابسته است.

زیرپرس‌وجوی وابســـته ممکن اســـت برای هر ردیفی که توسط پرس‌وجوی اصلی فرستاده می‌شود یک بار اجرا شود و به همین دلیل، گاهی آن‌ها را زیرپرس‌وجوهای تکرارشونده[1] هم می‌نامند. یک عبارت حاوی زیرپرس‌وجوی وابسته به صورت زیر اجرا می‌شود:

۱. پرس‌وجوی اصلی یک رکورد را بازیابی نموده و آن را به زیرپرس‌وجو می‌فرستد.

۲. زیرپرس‌وجو بر اساس مقادیر دریافت شده اجرا می‌گردد.

۳. زیرپرس‌وجو نتیجه حاصل شـــده را برای پرس‌وجوی اصلی می‌فرســـتد تـــا از آن برای اتمام پردازش استفاده کند.

این روند آن‌قدر تکرار می‌شود تا همه نتایج مورد نیاز از پایگاه داده استخراج گردد.

---

1 . Repeating Subquery

Subquery

A subquery is a query that is nested inside a SELECT, INSERT, UPDATE, or DELETE statement, or inside another subquery. A subquery can be used anywhere an expression is allowed.

A subquery is also called an inner query or inner select, while the statement containing a subquery is also called an outer query or outer select.

Many Transact-SQL statements that include subqueries can be alternatively formulated as joins. Other questions can be posed only with subqueries. In Transact-SQL, there is usually no performance difference between a statement that includes a subquery and a semantically equivalent version that does not. However, in some cases where existence must be checked, a join yields better performance. Otherwise, the nested query must be processed for each result of the outer query to ensure elimination of duplicates. In such cases, a join approach would yield better results

Correlated Subquery

Many queries can be evaluated by executing the subquery once and substituting the resulting value or values into the WHERE clause of the outer query. In queries that include a correlated subquery (also known as a repeating subquery), the subquery depends on the outer query for its values. This means that the subquery is executed repeatedly, once for each row that might be selected by the outer query.

EXISTS Keyword

When a subquery is introduced with the keyword EXISTS, the subquery functions as an existence test. The WHERE clause of the outer query tests whether the rows that are returned by the subquery exist. The subquery does not actually produce any data; it returns a value of TRUE or FALSE.

۱. متن بالا را مطالعه کرده و در مورد زیرپرس‌وجوها توضیح دهید.
۲. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. تفاوت زیرپرس‌وجوهای تک‌مقداری و چند مقداری را با ذکر مثال توضیح دهید.

۲. پرس‌وجویی بنویسید که شماره و نام هر کتاب را استخراج نموده و در ستونی با عنوان «مجموع فروش بهار ۸۹»، درآمد حاصل از فروش آن کتاب را در بهار ۸۹ نشان دهد.

۳. پرس‌وجویی طراحی کنید که شماره هر فاکتور و مجموع بهای اقلام موجود در آن را محاسبه نموده و در ستونی جدید، مقدار به دست آمده را به صورت زیر ارزیابی کند :

بهای فاکتور	دسته
۱۰۰۰۰۰ ریال و کم‌تر	دسته ۳
بین ۱۰۰۰۰۰ ریال و ۲۰۰۰۰۰ ریال	دسته ۲
۲۰۰۰۰۰ ریال و بیش‌تر	دسته ۱

۴. نام نویسندگانی را برگردانید که مجموع فروش حداقل یکی از کتاب‌های آن‌ها بیش‌تر از ۱۰۰۰ نسخه باشد.

۵. تفاوت زیرپرس‌وجوهای مستقل و وابسته را با ذکر مثال شرح دهید.

فصل یازدهم

فصل یازدهم : کار با جداول

داده‌های موجود درون یک پایگاه داده در جداول ذخیره‌سازی می‌شوند؛ از این رو آشنایی با دستورات و قابلیت‌هـــای SQL Server برای بازیابی اطلاعات از جداول و به‌کارگیری آن‌ها در هنگام نگارش پرس‌وجوها کاملاً ضروری است.

گاهی اوقات لازم اســـت برای نگارش پرس‌وجوهای پیچیـــده، جداول موقتی را درون پرس‌وجو ایجاد یا داده‌های حاصل از الحاق جداول را درون نما[1] نگه‌داری کنیم؛ نما حاوی تعدادی از ستون‌ها و سطرهای یک یا چند جدول اســـت که نمایش خاصی از داده‌های درون پایگاه داده را در معرض دید کاربران قرار می‌دهد. در این فصل با مجموعه‌ای از مفاهیم مرتبط با جداول آشـــنا خواهید شد که توانایی شما را در بازیابی نتایج موردنظر از پایگاه داده به نحو چشم‌گیری افزایش می‌دهد.

۱- ۱۱ جدول مشتق‌شده[2]

جداول مشتق‌شده مجموعه نتایجی هستند که در پرس‌وجو مانند یک جدول مورد استفاده قرار می‌گیرند. گاهـــی اوقات برای نگارش پرس‌وجوهای پیچیده باید مجموعـــه‌ای از نتایج را درون یک جدول قرار داده و ســـپس عملیات موردنظر را روی آن جدول اجرا کنید. یک راه‌حل ابتدایی برای انجام این کار، تولید جداول موقت[3] است؛ اما تعدد مراحل به‌کارگیری این روش (شامل ایجاد جدول موقت، وارد کردن داده‌ها درون آن، بازیابی نتایج موردنظر و نهایتاً حذف جدول موقت) باعث محبوبیت Derived Table ها گردیده است چراکه

1 .View
2 . Derived Table
3 . Temporary Tables

جدول مشتق‌شده به سادگی و در یک مرحله مورد استفاده قرار می‌گیرد و ضمناً سرعت اجرای بالاتری دارد.

می‌خواهیم مجموع کتاب‌های خریداری شده توسط مشتریان را بررسی و دسته‌بندی زیر را ایجاد نماییم و در ادامه تعیین کنیم که در هر دسته چند مشتری وجود دارد.

دسته	مجموع کتاب‌های خریداری شده
A	بیش‌تر از ۱۰۰ جلد
B	بین ۵۰ تا ۱۰۰ جلد
C	کم‌تر از ۵۰ جلد

برای ایجاد این پرس‌وجوی نسبتاً پیچیده ابتدا باید پرس‌وجویی بنویسیم که مشخص کند هر مشتری چند جلد خریداری کرده است.

```
SELECT f.CustomerID,
SUM(fd.Quantity) AS TotalSum
FROM Tbl_FactorDetails fd INNER JOIN Tbl_Factors f
ON fd.FactorID = f.ID
GROUP BY f.CustomerID
```

	CustomerID	TotalSum
1	1	125
2	3	85
3	4	137
4	5	60
5	8	66
6	10	35

حال با اضافه کردن یک عبارت CASE، دسته‌ای را که هر مشتری به آن تعلق دارد مشخص می‌کنیم.

```
SELECT f.CustomerID,
SUM(fd.Quantity) AS TotalSum ,
CASE
WHEN SUM(fd.Quantity)>100 THEN 'A'
WHEN SUM(fd.Quantity) BETWEEN 50 AND 100 THEN 'B'
WHEN SUM(fd.Quantity)<100 THEN 'C'
```

```
END AS Grade
FROM Tbl_FactorDetails fd INNER JOIN Tbl_Factors f
ON fd.FactorID = f.ID
GROUP BY f.CustomerID
```

	CustomerID	TotalSum	Grade
1	1	125	A
2	3	85	B
3	4	137	A
4	5	60	B
5	8	66	B
6	10	35	C

اکنون می‌خواهیم روی این جدول عملیاتی انجام دهیم تا تعداد مشـــتری‌های هر دســـته به دست آید. برای انجام این کار پرس‌وجوی فوق را درون پرانتز قرار داده و به این ترتیب آن را به یک جدول مشتق‌شده تبدیل می‌کنیم. سپس با افزودن تابع تجمعی COUNT، نتیجه موردنظر را استخراج می‌کنیم.

```
SELECT DerivedTable.Grade,COUNT(DerivedTable.Grade) AS GradeCount
FROM
(SELECT f.CustomerID,
SUM(fd.Quantity) AS TotalSum ,
CASE
WHEN SUM(fd.Quantity)>100 THEN 'A'
WHEN SUM(fd.Quantity) BETWEEN 50 AND 100 THEN 'B'
WHEN SUM(fd.Quantity)<100 THEN 'C'
END AS Grade
FROM Tbl_FactorDetails fd INNER JOIN Tbl_Factors f
ON fd.FactorID = f.ID
GROUP BY f.CustomerID)AS DerivedTable
GROUP BY DerivedTable.Grade
```

	Grade	GradeCount
1	A	2
2	B	3
3	C	1

۲- ۱۱ کار با Common Table Expression [1]

CTE ، مجموعـــه‌ای از نتایج را به صورت موقت تولید می‌کند تا با اجرای یک پرس‌وجوی دیگر بر روی آن بتوانید نتایج دلخواه را از پایگاه داده اســـتخراج نموده یا تغییرات موردنظر را بر روی داده‌ها اعمال کنید. CTE از این منظر که به صورت یک شـــیء در پایگاه داده ذخیره نمی‌شـــود و طول عمر آن محدود به زمان اجرای پرس‌وجو اســـت مانند جدول مشتق‌شده است اما برخلاف جدول مشتق‌شده می‌تواند به خود ارجاع داشته باشد.

CTE می‌تواند در موارد زیر مورد استفاده واقع شود:

- ساخت پرس‌وجوهای بازگشتی [2]
- جایگزینی با نما
- گروه‌بندی داده‌ها بر اساس ستونی که از یک زیرپرس‌وجو مشتق شده است.

اســـتفاده از CTE مزیت‌هایی دارد که از جمله آن‌ها افزایش خوانایی و سادگی نگه‌داشت پرس‌وجوهای پیچیده اســـت. در این حالت پرس‌وجو به چند قطعه مجزا و ســـاده تبدیل می‌شـــود و ضمناً می‌توان از این قطعات برای ساخت پرس‌وجوهای پیچیده‌تر استفاده نمود.

شکل کلی نگارش CTE به صورت زیر است.

```
WITH expression_name [ ( column_name [,...n] ) ]
AS
( CTE_query_definition )
```

CTE پس از تعریف می‌تواند به شکل زیر مانند یک جدول یا نما مورد مراجعه قرار گیرد.

```
SELECT <column_list>
FROM expression_name;
```

مثال ۱:پرس‌وجوی دسته‌بندی مشتریان را که در بخش قبل با کمک جدول مشتق‌شده پیاده‌سازی کردیم، با استفاده از CTE بازنویسی کنید.

---

1 . CTE

2 . Recursive Queries

```
WITH Grade_CTE
AS
(SELECT f.CustomerID,
SUM(fd.Quantity) AS TotalSum ,
CASE
WHEN SUM(fd.Quantity)>100 THEN 'A'
WHEN SUM(fd.Quantity) BETWEEN 50 AND 100 THEN 'B'
WHEN SUM(fd.Quantity)<100 THEN 'C'
END AS Grade
FROM Tbl_FactorDetails fd INNER JOIN Tbl_Factors f
ON fd.FactorID = f.ID
GROUP BY f.CustomerID)

SELECT Grade_CTE.Grade,COUNT(Grade_CTE.Grade) AS GradeCount
FROM Grade_CTE
GROUP BY Grade_CTE.Grade
```

مثال ۲: با استفاده از CTE پرس‌وجویی بنویسید که درآمد حاصل از فروش کتاب شماره ۱۲۱۰ را در بهار ۸۹ نشان دهد.

```
WITH TotalSale_CTE
AS
(SELECT b.ID, SUM((100-fd.Discount) * .01 * b.Price * fd.Quantity) AS TotalPrice
FROM Tbl_Books b INNER JOIN
Tbl_FactorDetails fd ON b.ID = fd.BookID INNER JOIN
Tbl_Factors f ON fd.FactorID = f.ID
WHERE f.[Date] BETWEEN '1389/01/01' AND '1389/03/31'
GROUP BY b.ID)

SELECT TotalPrice
FROM TotalSale_CTE
WHERE ID=1210
```

بررسی کد :

- درون TotalSale_CTE با ایجاد ســه الحاق میان جداول زیر مجموع فروش همه کتاب‌ها در بازه زمانی موردنظر محاسبه و با عبرات TotalPrice نام‌گذاری شده است.

جدول کتاب‌ها برای استخراج قیمت کتاب

جدول فاکتورها برای بررسی تاریخ فروش

جدول جزییات فروش برای محاسبه میزان تخفیف و تعداد نسخه‌های فروخته شده

- پرس‌وجوی SELECT با مراجعه به این جدول موقت، فروش کتاب شماره ۱۲۱۰ را برمی‌گرداند.

۳- ۱۱ کار با نماها[1]

۱-۳-۱۱ معرفی نما

View که «نما» یا «دیدگاه» هم گفته می‌شود، یک جدول مجازی است که در قالب یک پرس‌وجو درون پایگاه داده ذخیره می‌شود و داده‌های موجود در یک یا چند جدول را به شکلی قابل‌فهم در معرض دید کاربر پایگاه داده قرار می‌دهد. توجه داشته باشید که درون نما هیچ داده‌ای وجود ندارد؛ فقط پرس‌وجوی سازنده نما در پایگاه داده ذخیره شده و هنگام بازشدن نما، داده‌ها را از جداول استخراج می‌کند و نمایش می‌دهد. ایجاد نما در یک پایگاه داده غالباً به دو دلیل زیر انجام می‌پذیرید :

الف) ارتقاء امنیت : گاهی اوقات مشاهده اطلاعات همه سطرها یا ستون‌های یک جدول توسط کاربران مجاز نیســـت. در این حالت نمایی از جدول ساخته می‌شـــود که فاقد ستون‌ها یا سطرهای حاوی اطلاعات حســـاس اســـت و به این ترتیب کاربر فقط به بخشی از اطلاعات دسترســـی خواهد داشت. برای مثال شما می‌توانید از جدول مشـــتریان نمایی بســـازید که فاقد فیلد «شماره تلفن» باشـــد و ضمناً مشتریان ساکن «تهران» را نشان ندهد.

جدول مشتریان

ID	Name	Province	City	Tel
۱	انتشارات طلوع	تهران	رودهن	۲۲۳۳۲۲۱۱
۲	کتابفروشی افق	فارس	شیراز	۷۷۶۶۵۵۲
۳	انتشارات عابد	گلستان	گرگان	۶۶۳۳۳۳۱۱

1 . Views

نمای ساخته شده

City	Province	Name	ID
شیراز	فارس	کتابفروشی افق	۲
گرگان	گلستان	انتشارات عابد	۳

ب) افزایش خوانایی داده‌ها : در بخش‌های مربوط به طراحی پایگاه داده مشاهده نمودید که داده‌ها باید در چند جدول توزیع شوند تا بتوان ذخیره و بازیابی اطلاعات را با سرعت و صحت مناسبی انجام داد. از این رو در تعدادی از جداول، مقادیر برخی ستون‌ها به صورت ارجاعی نگه‌داری می‌شوند. اگر بخواهید داده‌ها را در معرض دید یک کاربر معمولی قرار دهید یا آن‌ها را در گزارشی چاپ کنید باید این مقادیر ارجاعی را به مقادیر قابل فهم تبدیل نمایید. نما، کارامدترین ابزار موجود در SQL Server برای انجام این کار است.

برای نمونه با ترکیب جدول جزییات فاکتورها با جدول کتاب‌ها و ایجاد یک نما می‌توانیم بفهمیم که کتابی که فروخته شده و فقط شماره آن در جدول جزییات فاکتور درج شده، چه عنوانی دارد و قیمت آن چه‌قدر است.

جدول جزییات فاکتورها

Quantity	Discount	BookID	FactorID
۲۰	۱۰	۱۲۰۵	۱۰۰
۲۵	۱۵	۱۰۲۶	۱۰۰
۳۰	۲۵	۱۲۰۵	۱۰۲

جدول کتاب‌ها

Price	Author	Title	ID
۵۰۰۰۰	محمد امیری	اینترنت	۱۲۰۵
۶۵۰۰۰	امیر محمدی	فتوشاپ	۱۲۰۶

نمای جزییات فاکتورها

Price	Title	Quantity	Discount	BookID	FactorID
۵۰۰۰۰	اینترنت	۲۰	۱۰	۱۲۰۵	۱۰۰
۶۵۰۰۰	فتوشاپ	۲۵	۱۵	۱۰۲۶	۱۰۰
۵۰۰۰۰	اینترنت	۳۰	۲۵	۱۲۰۵	۱۰۲

حتی می‌توان یک فیلد محاسباتی نیز به نما اضافه کرد تا مثلاً مجموعه مبلغ هر ردیف از فاکتور را نشان دهد. ضمناً با توجه به این که نام کتاب در نما وجود دارد می‌توانیم شماره کتاب را از مجموعه ستون‌ها حذف کنیم.

نمای مجموع ردیف‌های فاکتور

TotalSum	Price	Title	Quantity	Discount	FactorID
۹۰۰۰۰۰	۵۰۰۰۰	اینترنت	۲۰	۱۰	۱۰۰
۱۳۸۱۲۵۰	۶۵۰۰۰	فتوشاپ	۲۵	۱۵	۱۰۰
۱۱۲۵۰۰۰	۵۰۰۰۰	اینترنت	۳۰	۲۵	۱۰۲

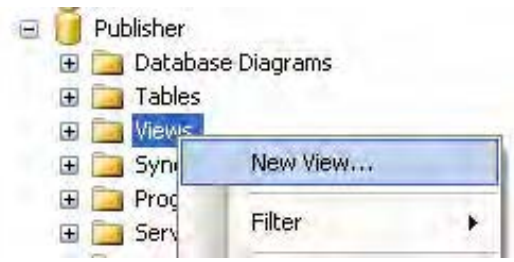
نما را می‌توان با استفاده از واسط گرافیکی SQL Server یا کدنویسی ایجاد کرد. در ادامه با هر دوی این روش‌ها آشنا خواهید شد.

۲-۳-۱۱ ایجاد نما در Management Studio

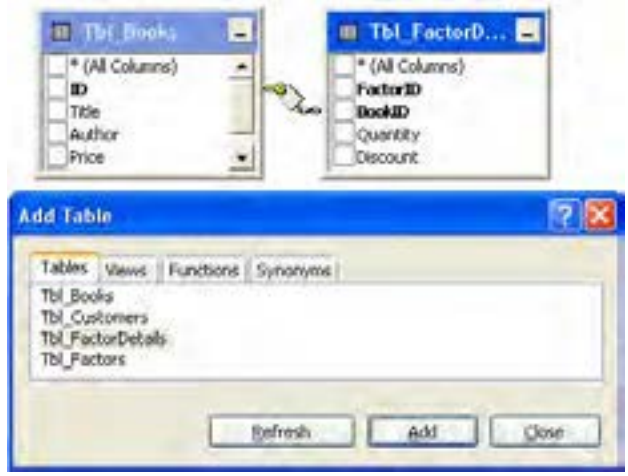
برای ایجاد نمای «مجموع ردیف‌های فاکتور» که در بخش قبل با ساختار آن آشنا شدید به روش زیر عمل نمایید:

۱. در قاب مرورگر اشیاء، پوشه پایگاه داده را باز کنید.

۲. روی پوشه Views راست‌کلیک نموده و گزینه New View را انتخاب نمایید.



۳. در پنجره Add Table، جدول Tbl_Books را انتخاب و روی دکمه Add کلیک کنید. همین کار را برای جدول Tbl_FactorDetails هم انجام دهید تا وارد محیط طراحی شود.



۴. در قاب نمودار[1]، فیلدهای FactorID، Quantity و Discount را از جدول جزییات فاکتورها و فیلدهای Title و Price را از جدول کتاب‌ها تیک بزنید.

۵. در آخرین ردیف قاب معیارها[2] و ستون Column، فرمول زیر را برای محاسبه مجموع هر سطر وارد کنید.

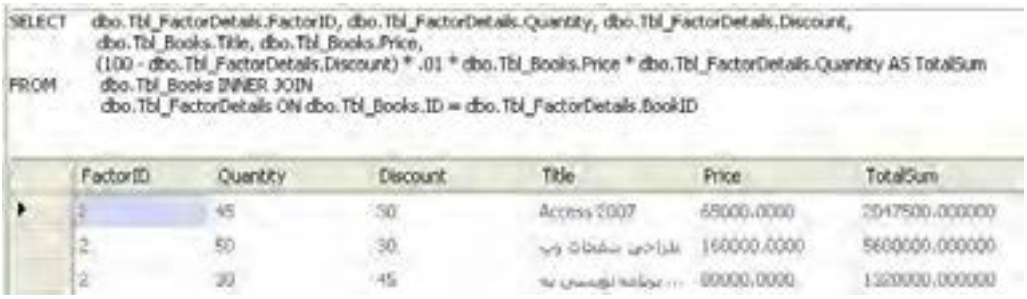
(100-Discount)*.01*Price*Quantity

۶. در ستون Alias، عبارت TotalSum را به عنوان نام مستعار ستون درج کنید.

1 . Diagram Pane
2 . Criteria Pane

۷. در نـــوار منوی Management Studio روی دکمه Execute کلیک کنید تا پرس‌وجوی تولید شـــده اجرا شود.

۸. در قاب SQL، پرس‌وجوی ایجاد شده را می‌بینید و قاب نتایج[1]، محتویات نما را نشان می‌دهد.



۹. اگر پرس‌وجو، داده‌های موردنظر را برمی‌گرداند، در نوار منو روی آیکن دیســـکت کلیک و نامی را برای نما وارد کنید. با کلیک روی دکمه OK، نما در پایگاه داده ذخیره می‌شود.

۱۰. پوشه View از پایگاه داده Publisher را باز کنید.

۱۱. روی نمای ســـاخته شده راست‌کلیک کنید. منویی شـــبیه به آن‌چه با راست‌کلیک روی جداول ظاهر می‌شـــود را می‌بینید؛ معنی گزینه‌های ظاهر شـــده این است که شـــما با یک جدول جدید مواجه هستید؛ جدولی مجازی که در قالب یک پرس‌وجو درون پایگاه داده ذخیره شده است.

۳-۳-۱۱ ساخت نما با کدنویسی

همانند سایر اشیاء SQL Server، برای ساخت نماها هم می‌توانید از کدهای SQL استفاده کنید. نگارش عمومی برای ایجاد یک نما به صورت زیر است :

```
CREATE VIEW view_name AS
SELECT column_name(s)
FROM table_name or view_name
WHERE condition
```

مثال ۱ :نمایی بسازید که مشخصات فاکتورهای صادر شده در سال ۱۳۸۸ را ذخیره کند.

---

1 . Result Pane

کد زیر را در محیط Management Studio وارد و آن را اجرا می‌کنیم.

```
CREATE VIEW View_Factors_Year_88
AS
SELECT * FROM Tbl_Factors
WHERE [Date] BETWEEN '1388/01/01' AND '1388/12/29'
```

ظاهر شدن پیغام Command(s) completed successfully در قاب نتایج نشان می‌دهد که پرس‌وجو با موفقیت اجرا و نما ساخته شده است.

مثال ۲ : با استفاده از نمای View_TotalSum که در بخش قبل ساختیم، مبلغ کل هر فاکتور را استخراج و در یک نمای دیگر ذخیره کنید.

```
CREATE VIEW View_FactorsSum
AS
SELECT FactorID , SUM(TotalSum) AS FinalSum FROM View_TotalSum
GROUP BY FactorID
```

با راست‌کلیک روی پوشه Views و انتخاب گزینه Refresh، نمای ساخته شده ظاهر می‌گردد. روی آن راست‌کلیک نموده و گزینه Select Top 1000 Rows را انتخاب کنید. محتویات نما ظاهر می‌شود.

Results | Messages

	FactorID	FinalSum
1	2	8967500.000000
2	3	5585750.000000
3	4	1398750.000000

۴-۱۱ توابع Inline تعریف شده توسط کاربر

توابع تعریف شـــده توسط کاربر[1] توابعی هستند که تعدادی پارامتر را دریافت نموده و پس از انجام یک عملیات (مانند محاســـبه یک مقدار یا جداسازی بخشی از رکوردها) نتیجه را برمی‌گردانند. چنان‌چه نتیجه برگشت داده شده یک جدول باشد، تابع از نوع Inline (درون‌برنامه‌ای) است. توابع Inline از لحاظ ساختاری مانند نماها هستند با این تفاوت که می‌توانند پارامترهایی را به عنوان ورودی بپذیرند.

فرض کنید می‌خواهیم نمایی بســـازیم که همه فاکتورهای صادر شده برای مشتری شماره ۴ را در خود

1 . User-defined Functions

جای دهد. این نما به شکل زیر تعریف می‌شود :

```
CREATE VIEW View_Factors_Customer4
AS
SELECT * FROM Tbl_Factors
WHERE CustomerID=4
```

در این حالت هرگاه بخواهید فاکتورهای صادر شـــده برای مشتری دیگری را در قالب یک نما برگردانید باید تعریف نما را تغییر داده و شماره مشتری موردنظر را جایگزین عدد ۴ کنید. خوشبختانه SQL Server به شما امکان می‌دهد با ایجاد یک تابع Inline، شماره مشتری را در قالب یک پارامتر برای تابع تعریف نمایید.

شکل کلی تعریف تایع Inline به صورت زیر است :

```
CREATE FUNCTION function_name
(parameters list)
RETURNS TABLE
AS
RETURN
(expression)
```

برای تعریف نمای توضیح داده شده در ابتدای بخش:

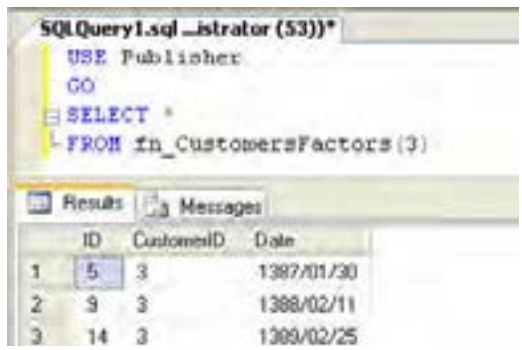
۱. دستور زیر را در محیط Management Studio وارد و آن را اجرا کنید:

```
CREATE FUNCTION fn_CustomersFactors
(@cid int)
RETURNS TABLE
AS
RETURN
(SELECT * FROM Tbl_Factors
WHERE CustomerID=@cid)
```

۲. ظاهر شدن عبارت Command(s) completed successfully در قاب Massages نشان می‌دهد دستور با موفقیت اجرا شده است.

۳. در قاب مرورگر اشیاء، پوشـــه‌های Programmbility، Functions و Table-valued Functions را باز نمایید. همان‌طور که مشاهده می‌کنید، تابع ایجاد شده است.

۴. پرس‌وجوی زیر را نوشته و آن را اجرا کنید.



می‌خواهیم تابع را به گونه‌ای تغییر دهیم که بتواند فاکتورهای صادر شده بعد از یک تاریخ خاص را هم برگرداند. برای انجام این کار باید یک پارامتر دیگر به تابع اضافه کنیم. برای تعریف پارامترها در SQL Server یک قاعده کلی به صورت زیر وجود دارد.

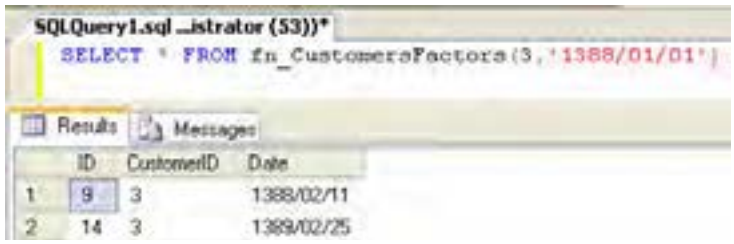
@ParameterName DataType

۵. کد زیر را وارد و آن را اجرا کنید. در فصل ۷ یادگرفتید که برای ایجاد تغییر در اشیاء پایگاه داده باید از کلیدواژه ALTER استفاده نمایید.

```
ALTER FUNCTION fn_CustomersFactors
(
@cid int,
@startdate char(10)
)
RETURNS TABLE
```

```
AS
RETURN
(SELECT * FROM Tbl_Factors
WHERE CustomerID=@cid AND [Date]>@startdate
)
```

۶. این بار برای اجرای تابع باید از دو پارامتر «شماره مشتری» و «تاریخ شروع فاکتورها» استفاده کنید. پرس‌وجوی زیر را اجرا کنید تا نتایجی شبیه به آن‌چه در تصویر وجود دارد را ببینید.



توابع Inline تعریف شده توسط کاربر تنها یک نوع از توابع موجود در SQL Server هستند. برای مثال تابع استخراج تاریخ جاری (GETDATE( )) یا محاسبه میانگین (AVG( )) انواع دیگری از توابع هستند که چون همراه با SQL Server عرضه می‌شـــوند، توابع سیستمی[1] نام دارند. شما می‌توانید برای انجام عملیات موردنظر روی داده‌های درون جداول، توابع موردنظر خود را ایجاد کنید. این توابع که «توابع تعریف شـــده توسط کاربر[2]» نام دارند برای مثال می‌توانند شماره مشتری و یک بازه زمانی را به صورت پارامتری دریافت نموده و مجموع خرید مشـــتری را به عنوان خروجی برگردانند. نکته مهم در مورد همه توابع این اســـت که اگرچه می‌توانند چندین ورودی داشـــته باشند اما فقط امکان برگرداندن یک خروجی را دارند. این خروجی می‌تواند یک عدد، یک رشته یا یک جدول باشد اما در صورت فقط یکی است.

استفاده از توابع مزایای زیر را دارد:

- امکان برنامه‌نویسی پیمانه‌ای[3] را فراهم می‌آورند؛ به این معنی که یک کد پیچیده را به چندین تابع کوچک‌تر تقســـیم نموده و از کنار هم قرار دادن این توابع، به نتیجه نهایی می‌رســـید. در این روش عیب‌یابی برنامه ساده‌تر است و از توابع ساخته شده می‌توان در سایر برنامه‌ها هم استفاده کرد.
- اجرای برنامه ســـریع‌تر می‌شـــود؛ چون تابع یک بار کامپایل و در پایگاه داده ذخیره می‌شـــود و در اجراهای بعدی نیازی به کامپایل مجدد کد تابع نیست.

---

1 . System Functions
2 . User-defined Functions
3 . Modular

- ترافیک شبکه کاهش می‌یابد؛ اگر برنامه تحت شبکه اجرا شود و از توابع استفاده نکنید، هربار باید مجموعه‌ای از رکوردها را از سرویس‌دهنده[1] خوانده و روی رایانه سرویس‌گیرنده[2] پردازش کنید تا به نتیجه موردنظر برسید. اما تابع عمل پردازش را روی سرویس‌دهنده انجام داده و فقط نتیجه را برای سرویس‌گیرنده ارسال می‌کند.

۱۱-۵ کار با APPLY

عمل‌گر APPLY به شما امکان می‌دهد یک تابع Inline را برای هر ردیف برگردانده شده توسط جدول موجود در پرس‌وجو فراخوانی کنید. برای به‌کار گیری این عمل‌گر، پرس‌وجویی ایجاد می‌شود که در سمت راست APLLY یک تابع Inline قرار دارد و در سمت چپ آن یک جدول قرار گرفته است؛ به ازای هر ردیفی که از جدول استخراج می‌شود، مقدار تابع ارزیابی شده و برای تولید خروجی نهایی مورد استفاده قرار می‌گیرد. مجموعه ستون‌هایی که در خروجی نهایی دیده می‌شوند، شامل ستون‌های جدول و نیز ستون‌های برگردانده شده توسط تابع Inline هستند.

عمل‌گر APPLY به دو صورت زیر مورد استفاده قرار می‌گیرد :

- **CROSS APPLY :** این عمل‌گر فقط ردیف‌هایی را از جدول برمی‌گرداند که مجموعه نتایجی را برای تابع Inline تولید می‌کنند.
- **OUTER APPLY :** این عمل‌گر همه ردیف‌های جدول را برمی‌گرداند، چه آن‌هایی که منجر به برگشت نتیجه از تابع می‌شوند و چه آن‌هایی که برای تابع، خروجی تولید نمی‌کنند. در حالت دوم، به جای خروجی تابع مقدار NULL در فیلدهای قرار داده می‌شود.

برای درک بهتر نحوه کار این عمل‌گر ابتدا تابعی ایجاد می‌کنیم تا با گرفتن شماره مشتری، مشخصات وی را برگرداند.

```
CREATE FUNCTION fn_GetCustomerFactors
(@cid int)
RETURNS TABLE
AS
RETURN
(
--Outer Query
```

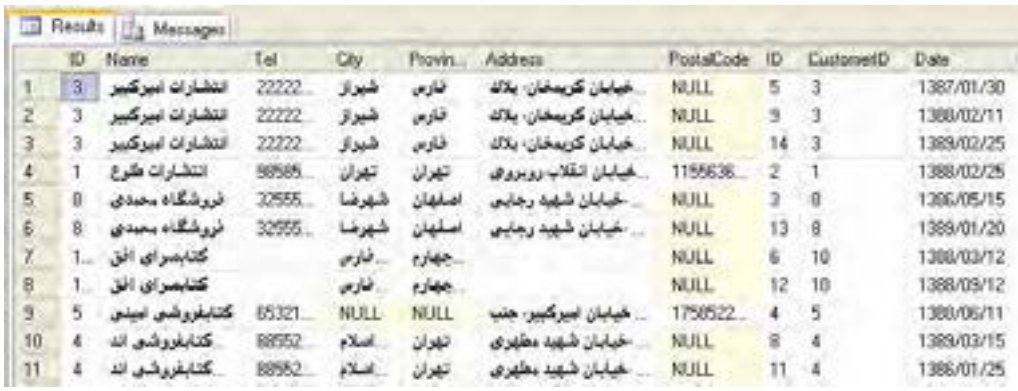
1 . Server
2 . Client

```
SELECT *
FROM Tbl_Factors f
WHERE f.CustomerID=@cid
)
```

حال با اســـتفاده از CROSS APPLY مشخصات مشتریان را استخراج و جزییات فاکتورهای صادر شده برای آن‌ها را هم نمایش می‌دهیم.

```
SELECT * FROM Tbl_Customers c
CROSS APPLY
fn_GetCustomerFactors(c.ID)
ORDER BY c.Name
```

در واقع به ازای هر مشـــتری، تابع تعریف شده، جزییات فاکتورهای مربوط به وی را استخراج می‌کند و به این ترتیب نام هر مشتری به تعداد فاکتورهای صادر شده برای وی تکرار می‌شود.



	ID	Name	Tel	City	Provin...	Address	PostalCode	ID	CustomerID	Date
1	3	انتشارات امیرکبیر	22222...	شیراز	فارس	خیابان کریمخان، بلاک	NULL	5	3	1387/01/30
2	3	انتشارات امیرکبیر	22222...	شیراز	فارس	خیابان کریمخان، بلاک	NULL	9	3	1388/02/11
3	3	انتشارات امیرکبیر	22222...	شیراز	فارس	خیابان کریمخان، بلاک	NULL	14	3	1389/02/25
4	1	انتشارات فکرو	98585...	تهران	تهران	خیابان انقلاب، روبروی	1155636...	2	1	1388/02/25
5	8	فروشگاه محمدی	32555...	شهرضا	اصفهان	خیابان شهید رجایی	NULL	3	8	1386/05/15
6	8	فروشگاه محمدی	32555...	شهرضا	اصفهان	خیابان شهید رجایی	NULL	13	8	1389/01/20
7	1...	کتابسرای افق		فارس	چهارم		NULL	6	10	1388/03/12
8	1...	کتابسرای افق		فارس	چهارم		NULL	12	10	1388/09/12
9	5	کتابفروشی امینی	65321...	NULL	NULL	خیابان امیرکبیر، جنب	1758522...	4	5	1388/06/11
10	4	کتابفروشی اند	88552...	اسلام	تهران	خیابان شهید مطهری	NULL	8	4	1389/03/15
11	4	کتابفروشی اند	88552...	اسلام	تهران	خیابان شهید مطهری	NULL	11	4	1386/01/25

اکنون پرس‌وجوی قبل را با استفاده از عمل‌گر OUTER APPLY بازنویسی و اجرا کنید.

```
SELECT * FROM Tbl_Customers c
OUTER APPLY
fn_GetCustomerFactors(c.ID)
ORDER BY c.Name
```

در این حالت علاوه بر مشـــتریانی که برای آن‌ها فاکتور صادر شده، مشخصات مشتریان فاقد فاکتور هم در نتیجه نهایی ظاهر می‌شـــود. بدیهی اســـت که در فیلدهای فاکتور این مشتریان مقدار NULL قرار داده می‌شود.

Results | Messages

	ID	Name	Tel	City	Provin...	Address	PostalCode	ID	CustomerID	Date
1	7	اداره ارشاد سپیدان	4585...	سپیدان	فارس	خیابان شهید بهشتی ...	NULL	NULL	NULL	NULL
2	3	انتشارات امیرکبیر	2222...	شیراز	فارس	خیابان کریمخان، بلاک ...	NULL	5	3	1387/01/30
3	3	انتشارات امیرکبیر	2222...	شیراز	فارس	خیابان کریمخان، بلاک ...	NULL	9	3	1388/02/11
4	3	انتشارات امیرکبیر	2222...	شیراز	فارس	خیابان کریمخان، بلاک ...	NULL	14	3	1389/02/25
5	1	انتشارات طلوع	9858...	تهران	تهران	خیابان انقلاب-روبروی ...	1155636251	2	1	1388/02/25
6	11	رسا	88889...	تهران	تهران		NULL	NULL	NULL	NULL
7	6	فروشگاه کتاب آ...	2586...	اصفهان	اصفهان	روبروی هتل عباسی	132520220	NULL	NULL	NULL
8	8	فروشگاه محمدی	3255...	شهرضا	اصفهان	خیابان شهید رجایی ...	NULL	3	8	1386/05/15
9	8	فروشگاه محمدی	3255...	شهرضا	اصفهان	خیابان شهید رجایی ...	NULL	13	8	1389/01/20
10	10	کتابسرای اقل		فارسان	چهارم		NULL	6	10	1388/03/12
11	10	کتابسرای اقل		فارسان	چهارم		NULL	12	10	1388/09/12
12	2	کتابسرای محمدی	3626...	نجف	NULL	خیابان رابعصر نبش ك...	1395255142	NULL	NULL	NULL
13	5	کتابفروشی ادیبی	6532...	NULL	NULL	خیابان امیرکبیر، جنب ...	175852200	4	5	1388/06/11
14	4	کتابفروشی اند	8855...	اسلام	تهران	خیابان شهید مطهری ...	NULL	8	4	1389/03/15
15	4	کتابفروشی اند	8855...	اسلام	تهران	خیابان شهید مطهری ...	NULL	11	4	1386/01/25
16	9	کتابفروشی ستار	3334...	شهر...	چهارم	خیابان ملت، چهارراه ...	8815752225	NULL	NULL	NULL

Views

A view can be created only in the current database. A view can have a maximum of 1,024 columns.

When querying through a view, the Database Engine checks to make sure that all the database objects referenced anywhere in the statement exist and that they are valid in the context of the statement, and that data modification statements do not violate any data integrity rules. A check that fails returns an error message. A successful check translates the action into an action against the underlying table or tables.

If a view depends on a table or view that was dropped, the Database Engine produces an error message when anyone tries to use the view. If a new table or view is created and the table structure does not change from the previous base table to replace the one dropped, the view again becomes usable. If the new table or view structure changes, the view must be dropped and re-created.

CTE

A common table expression (CTE) provides the significant advantage of being able to reference itself, thereby creating a recursive CTE. A recursive CTE is one in which an initial CTE is repeatedly executed to return subsets of data until the complete result set is obtained.

۱. متن بالا را مطالعه کرده و در مورد نماها توضیح دهید.

۲. مزیت CTEها نسبت به نماها چیست؟

۳. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. جدول مشتق‌شده را با ذکر مثال توضیح دهید.

۲. با اســـتفاده از جدول مشتق شده پرس‌وجویی بنویسید که فاکتورها را به صورت زیر دسته‌بندی نموده و نهایتاً نشان دهد در هر دسته چند فاکتور قرار دارد؟

دسته	تعداد نسخه‌های خریداری شده در فاکتور
C	۳۰ و کمتر
B	بین ۳۰ و ۱۰۰
A	بیش‌تر از ۱۰۰

۴. یک CTE بسازید که پرس‌وجوی مثال قبل را پیاده‌سازی کند. سپس یک عبارت SELECT بنویسید که نشان دهد فاکتور شماره ۱۰ در کدام دسته قرار دارد.

۵. با اســـتفاده از کدهای SQL، نمایی بسازید که مشخصات فاکتور، نام مشتری و مجموع فاکتور را نشان دهد. سپس پرس‌وجویی طراحی کنید که مشخصات فاکتورهای سال ۸۹ را از این نما برگرداند.

۶. یک تابع Inline ایجاد کنید تا نام نویسنده را دریافت نموده و مشخصات کتاب‌هایی از این نویسنده را برگرداند که بیش‌تر از ۱۰۰۰ نسخه از آن‌ها فروش رفته است.

۷. تفاوت Cross Apply و Outer Apply را توضیح دهید.

فصل دوازدهم

فصل دوازدهم : کار با عمل‌گرهای مجموعه‌ای

در فصول ابتدایی این کتاب آموختید که پایگاه داده‌های رابطه‌ای از قوانین جبر رابطه‌ای پیروی می‌کنند بنابراین می‌توان روابطی مانند اجتماع، تفاضل و اشتراک را برای جداول (که مجموعه‌ای از داده‌ها محسوب می‌شوند) تعریف نمود. در این فصل با سه عمل‌گر UNION، INTERSECT و EXCEPT که روابط مذکور را پیاده‌سازی می‌کنند آشنا خواهید شد.

۱-۱۲ کار با UNION

عمل‌گر اجتماع یا UNIION به شما امکان می‌دهد مجموعه نتایج حاصل از اجرای دو یا چند پرس‌وجو را با هم ترکیب نموده و یک مجموعه واحد ایجاد کنید؛ مشـــروط به این‌که نوع داده‌ای و تعداد ســـتون‌ها با هم تطابق داشته باشد.

فرض کنید دو جدول زیر درون یک پایگاه داده تعریف شده باشد:

Table_1		
Column_A	Column_B	Column_C
int	nvarchar(15)	float
۱۲۰۵	محمد اکبری	۱۲,۵
۱۲۰۸	سعید رضایی	۱۴,۷۵

Table_2		
Column_D	Column_E	Column_F
int	nvarchar(15)	float
۱۲۱۰	رضا صالحی	۱۵,۲۵
۱۲۱۲	محسن بیانی	۱۸,۲۵
۱۲۰۵	محمد اکبری	۱۲,۵

حاصل اجرای دستور UNION، مجموعه‌ای از نتایج به صورت زیر است :

```
SELECT * FROM Table 1
UNION
SELECT * FROM Table 2
```

Column_A	Column_B	Column_C
۱۲۰۵	محمد اکبری	۱۲,۵
۱۲۰۸	سعید رضایی	۱۴,۷۵
۱۲۱۰	رضا صالحی	۱۵,۲۵
۱۲۱۲	محسن بیانی	۱۸,۲۵

در واقع عمل‌گر UNION نتایج پرس‌وجویی را که بعد از کلیدواژه UNION قرار گرفته به نتایج حاصل از پرس‌وجوی اول اضافه می‌کند اما موارد تکراری را حذف خواهد نمود. اگر می‌خواهید رکوردهای تکراری در مجموعه نتایج وجود داشته باشد باید از UNION ALL استفاده کنید. دقت داشته باشید که نام سرستون‌های نتایج، از نام ستون‌های پرس‌وجوی اول گرفته می‌شود.

اگر بخواهیم کار عمل‌گر UNION را با JOIN مقایسه کنیم به این نتیجه می‌رسیم که JOIN، جداول را به صورت افقی با هم ترکیب می‌کند یعنی معمولاً تعداد ستون‌ها افزایش می‌یابد اما UNION باعث ترکیب عمودی نتایج شده و تعداد ردیف را اضافه خواهد کرد.

مثال ۱: فرض کنید پایگاه داده دیگری با نام Publisher2 روی SQL Server وجود دارد که ساختار جداول موجود در آن شبیه به پایگاه داده Publiser است اما اطلاعات یک مؤسسه انتشاراتی دیگر در آن درج شده است. نتیجه اجرای پرس‌وجوی زیر چیست؟

```
SELECT ID, Title, Author, Price FROM Publisher.dbo.Tbl_Books
UNION
SELECT ID, Title, Author, Price FROM Publisher2.dbo.Tbl_Books
```

شماره، عنوان و قیمت کتاب‌های موجود در پایگاه داده Publisher2 در ادامه کتاب‌های پایگاه داده Publisher نشان داده می‌شود.

Results | Messages

	ID	Title	Author	Price	
19	1220	Access 2007	مهدی رضایی	83000.00	رکوردهای جدول کتاب ها از پایگاه داده Publisher
20	1221	CorelDraw	سعید برزگر	65000.00	
21	1222	FreeHand	سعید برزگر	52000.00	
22	1223	برنامه نویسی پایگاه داده با دلفی	محمد سعیدی	35200.00	
23	1224	Access 2007	رضا باقری	65000.00	
24	1225	ویندوز ویستا	حامد قنبری	45000.00	
25	1301	Access 2007	رضا اکبری	65000.00	رکوردهای جدول کتاب ها از پایگاه داده Publisher2
26	1302	ویندوز ویستا	محمود کبیری	140000.00	
27	1303	برنامه نویسی به زبان دلفی	منصور کبیری	80000.00	
28	1305	برنامه نویسی به زبان ویژوال بیسیک ۶	حامد قنبری	25000.00	
29	1306	برنامه نویسی به زبان C#	محمد محمدی	80000.00	
30	1307	برنامه نویسی به زبان JAVA	حمید مومنی	120000.00	

مثال ۲: فیلد Author را از دستور انتخاب اول حذف و پرس‌وجو را اجرا کنید. نتیجه چیست؟

```
SELECT ID, Title, Price FROM Publisher.dbo.Tbl_Books
UNION
```

```
SELECT ID, Title, Author, Price FROM Publisher2.dbo.Tbl_Books
```

با اجرای پرس‌وجو، خطای زیر ظاهر می‌شـــود؛ به این معنی که تعداد ستون‌های دو پرس‌وجویی که در اجتماع مشارکت دارند، برابر نیست.

All queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists.

مثال ۳ : پرس‌وجوی زیر چه نتایجی را برمی‌گرداند؟

```
SELECT ID, Title FROM Tbl_Books
UNION
SELECT BookID, Quantity FROM Tbl_FactorDetails
```

اجرای این پرس‌وجو با وقوع خطا مواجه می‌شـــود چون فیلد Title از نوع رشته و فیلد Quantity از نوع عددی است. در تعریف عمل‌گر اجتماع تأکید کردیم که نوع داده‌ای ستون‌ها باید با هم مطابقت داشته باشد.

۲-۱۲ کار با INTERSECT

عمل‌گر اشتراک، تنها رکوردهایی را برمی‌گرداند که در هر دو پرس‌وجو وجود دارند و ضمناً موارد تکراری را حذف خواهد کرد. برای جلوگیری از حذف موارد تکراری می‌توانید از INTERSET ALL استفاده نمایید.

مثال ۱ : خروجی اجرای پرس‌وجوی زیر چیست؟

```
SELECT * FROM Tbl_Factors
WHERE [Date] <= '1388/09/30'
INTERSECT
SELECT * FROM Tbl_Factors
WHERE [Date] >= '1388/04/01'
```

پرس‌وجوی اول، فاکتورهای صادر شده قبل از زمستان ۸۸ را برمی‌گرداند و پرس‌وجوی دوم فاکتورهای بعد از بهار همان سال را استخراج خواهد کرد. نتیجه اشتراک این دو نتیجه، فاکتورهایی است که در تابستان و پاییز ۸۸ صادر شده است.

مثال ۲ : پایگاه داده Publisher2 برای شـــماره‌گذاری کتاب‌های خود از شـــماره‌هایی متفاوت از پایگاه داده Publisher استفاده کرده است؛ با این‌حال کتاب‌های مشترکی در این دو پایگاه داده وجود دارد. ابا این فرض نتیجه اجرای پرس‌وجوی زیر چیست؟

```
SELECT * FROM Publisher.dbo.Tbl_Books
INTERSECT
SELECT * FROM Publisher2.dbo.Tbl_Books
```

هیچ رکوردی برگردانده نمی‌شـــود چون علی‌رغم وجود کتاب‌هایی با مشخصات یکسان، شماره کتاب‌ها متفاوت است و لذا هیچ رکورد مشترکی وجود ندارد.

مثال ۳ : پرس‌وجوی مثال قبل را به گونه‌ای تغییر دهید که کتاب‌های مشـــابه در دو پایگاه داده را برگرداند.

```
SELECT Title, Author, Price, Pages
FROM Publisher.dbo.Tbl_Books
INTERSECT
SELECT Title, Author, Price, Pages
FROM Publisher2.dbo.Tbl_Books
```

با حذف شماره کتاب از فیلدهای موجود در رکوردها، کتاب‌های مشترک استخراج می‌شود.

مثال ۴ : نام مشتریانی را به دست آورید که برای آن‌ها حداقل یک فاکتور صادر شده است.

```
SELECT ID FROM Tbl_Customers
INTERSECT
SELECT CustomerID FROM Tbl_Factors
```

۳-۱۲ کار با EXCEPT

با اســـتفاده از این عمل‌گر می‌توانید رکوردهای یکتایی را استخراج کنید که در پرس‌وجوی سمت چپ EXCEPT وجود دارند اما جزو نتایج پرس‌وجوی ســـمت راست آن نیستند. در این عمل‌گر، ضرورتی ندارد فیلدهای دو پرس‌وجو نوع داده‌ای یکسان باشند اما باید امکان مقایسه آن‌ها وجود داشته باشد.

مثال ۱: نتیجه اجرای پرس‌وجوی زیر چیست؟

```
SELECT * FROM Tbl_Factors
WHERE [Date] <= '1388/09/30'
EXCEPT
SELECT * FROM Tbl_Factors
WHERE [Date] >= '1388/07/01'
```

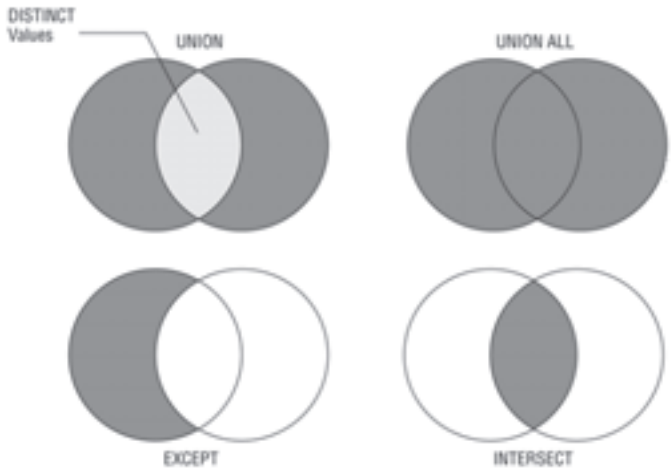
مشـــخصات فاکتورهایی برگردانده می‌شود که قبل از زمستان ۸۸ صادر شده‌اند اما در فاکتورهای پاییز ۸۸ و پس از آن وجود ندارد. بنابراین فاکتورهای قبل از پاییز ۸۸ استخراج خواهد شد.

مثال ۲: شماره مشتریانی را به دست آورید که برای آن‌ها هیچ فاکتوری صادر نشده است.

```
SELECT ID FROM Tbl_Customers
EXCEPT
SELECT CustomerID FROM Tbl_Factors
```

این پرس‌وجو شـــماره همه مشتریان را برمی‌گرداند به‌جز مشتریانی که شماره آن‌ها در جدول فاکتورها وجود دارد.

مثال ۳: عمل‌گرهای UNION، UNION ALL، INTERSECT و EXCEPT را روی نمودار ون[1] نشان دهید.



1 . Venn Diagram

EXCEPT and INTERSECT (Transact-SQL)

Returns distinct values by comparing the results of two queries.

EXCEPT returns any distinct values from the left query that are not also found on the right query.

INTERSECT returns any distinct values that are returned by both the query on the left and right sides of the INTERSECT operand.

The basic rules for combining the result sets of two queries that use EXCEPT or INTERSECT are the following:

- The number and the order of the columns must be the same in all queries.
- The data types must be compatible.

Syntax

```
{ <query_specification> | ( <query_expression> ) }

{ EXCEPT | INTERSECT }

{ <query_specification> | ( <query_expression> ) }
```

Arguments

<query_specification> | (<query_expression>)

Is a query specification or query expression that returns data to be compared with the data from another query specification or query expression. The definitions of the columns that are part of an EXCEPT or INTERSECT operation do not have to be the same, but they must be comparable through implicit conversion. When data types differ, the type that is used to perform the comparison and return results is determined based on the rules for data type precedence.

When the types are the same but differ in precision, scale, or length, the result is determined based on the same rules for combining expressions. For more information, see Precision, Scale, and Length (Transact-SQL).

The query specification or expression cannot return **xml**, **text**, **ntext**, **image**, or nonbinary CLR user-defined type columns because these data types are not comparable.

EXCEPT

Returns any distinct values from the query to the left of the EXCEPT operand that are not also returned from the right query.

INTERSECT

Returns any distinct values that are returned by both the query on the left and right sides of the INTERSECT operand.

۱. متن بالا را مطالعه کرده و در مورد دو عمل‌گر INTERSECT و EXCEPT توضیح دهید.

۲. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. پرس‌وجوی زیر چه نتایجی را برمی‌گرداند؟

```
SELECT * FROM Tbl_Books
UNION ALL
SELECT * FROM Tbl_Books
```

۲. با اســـتفاده از عملگر INTERSECT، پرس‌وجویی بنویسید تا مشـــخصات کتاب‌هایی را برگرداند که حداقل یک نسخه از آن‌ها فروخته شده است.

۳. با اســـتفاده از عملگر EXCEPT، پرس‌وجویی بنویسید تا مشخصات کتاب‌هایی را برگرداند که حداقل یک‌بار بدون هیچ تخفیفی به فروش رفته‌اند.

فصل سیزدهم

فصل سیزدهم : تغییر داده‌ها

طراحی و پیاده‌سازی یک پایگاه داده جنبه‌های متنوعی دارد که برخی از آن‌ها ساده و برخی دیگر پیچیده هستند. همه این جنبه‌ها در کنار یک‌دیگری باعث می‌شوند فرایند ذخیره‌سازی داده‌ها و بازیابی آن‌ها با سرعت، صحت و امنیت بالایی انجام گیرد.

در این میان، چهار عملیات اصلی شامل بازیابی[1]، بروزرسانی[2]، درج[3] و حذف[4] داده‌ها وجود دارد که پرکاربردترین دستورات در طول حیات یک پایگاه داده به شمار می‌روند. در فصول گذشته، عملیات بازیابی به صورت مفصل و با ذکر جزییات بررسی کردیم و اینک نوبت به سه عملیات بعدی رسیده است.

1 . Select or Retrieve
2 . Update
3 . Insert
4 . Delete

۱-۱۳ ورود داده‌ها

وارد کردن داده‌های جدید در جدول با استفاده از دستور درج یا INSERT انجام می‌شود که شکل کلی آن به صورت زیر است :

```
INSERT INTO table_name (field_1, filed_2 ,…, field_n)
VALUES (value_1 ,value_2 ,…, value_n)
```

چنان‌چه ترتیب و تعداد مقادیر موردنظر برای درج در جدول متناسب با فیلدهای جدول باشد می‌توانید از نگارش خلاصه‌تر دستور درج استفاده کنید:

```
INSERT INTO table_name VALUES (value_1 ,value_2 ,…, value_n)
```

مثال ۱ : پرس‌وجویی بنویسید که اطلاعات کامل یک کتاب جدید را در جدول کتاب‌ها درج کند. این پرس‌وجو را در محیط Management Studio نوشته و اجرا کنید.

```
INSERT INTO Tbl_Books VALUES(1240,'پایگاه داده','محمد رحیمی',55000,220 )
```

بررسی کد :

از آن‌جا که می‌خواهیم مقادیر همه فیلدها را وارد کنیم، استفاده از نگارش خلاصه جایز است و نیازی به ذکر نام فیلدها نیست.

مقادیر رشته‌ای باید درون یک جفت علامت نقل‌قول[1] قرار بگیرند.

در صورت اجرای موفقیت‌آمیز پرس‌وجو، عبارت 1 row(s) affected در قاب پیغام‌ها ظاهر می‌شود. به این معنی که این پرس‌وجو روی یک ردیف تأثیر گذاشته است.

مثال ۲ : پرس‌وجوی مثال قبل را مجدداً اجرا کنید. چه اتفاقی می‌افتد؟

پیغام خطای زیر ظاهر می‌شود و اعلام می‌کند که جدول دارای محدودیت کلید اصلی است و چون شماره کتاب یک‌بار در جدول درج شده و امکان درج تکراری کلید اصلی وجود ندارد، عملیات لغو شده است.

```
Violation of PRIMARY KEY constraint 'PK_Tbl_Books'. Cannot insert duplicate key in
object 'dbo.Tbl_Books'.
The statement has been terminated.
```

1 . Quotation

مثال ۳ : پرس‌وجویی بنویسید که شماره و نام یک کتاب را درج کند.

```
INSERT INTO Tbl_Books(ID,Title)
VALUES(1227,'اصول طراحی کامپایلر')
```

مثال ۴ : پرس‌وجویی بنویسید که اطلاعات یک کتاب را درج نموده و شماره کتاب را برابر با «شماره آخرین کتاب موجود در جدول به‌علاوه یک» قرار دهد.

```
INSERT INTO Tbl_Books
VALUES((SELECT MAX(ID)+1 FROM Tbl_Books),'رضا کریمی',
'Visual Basic.NET 2010',130000,520)
```

بررسی کد :

عبارت SELECT MAX(ID)+1 FROM Tbl_Books بزرگ‌ترین شماره موجود در جدول را استخراج و آن را به‌علاوه یک می‌کند. این عدد جایگزین شماره کتاب می‌شود.

اگر نام نویسنده را از پرس‌وجوی فوق حذف کنیم چه اتفاقی می‌افتد؟

عملیات درج انجام نمی‌گیرد چون وقتی از فیلدها نام نمی‌بریم باید تعداد و ترتیب فیلدها متناسب با تعریف فیلدهای جدول باشد.

اگر پرس‌وجوی مثال ۳ را مجدداً اجرا کنیم، آیا عملیات درج با موفقیت به انجام می‌رسد؟

بله، چون عبارت MAX(ID)+1 هربار یک شماره کتاب جدید تولید می‌کند و محدودیت کلید اصلی مانع از درج رکورد نخواهد شد.

مثال ۴ : اطلاعات یک رکورد جدید را در جدول فاکتورها ثبت کنید.

```
INSERT INTO Tbl_Factors
VALUES(12,'1389/02/31')
```

بررسی کد:

در جدول فاکتورها، شماره فاکتور به صورت خودکار تولید می‌شود و بنابراین نیازی نیست مقداری برای آن درنظر گرفته شود.

دقت داشـــته باشـــید که جدول فاکتورها از طریق فیلد شماره مشتری به جدول مشتریان مرتبط اســـت و ویژگی Enforce Foreign key ارتباط[1]، روی Yes تنظیم شده است. بنابراین نمی‌توانید هنگام درج فاکتور، شماره‌ای را به عنوان شماره مشتری تعیین کنید که در جدول مشتریان وجود ندارد.

مثال ۵ : فاکتوری با شماره مشتری نامعتبر در جدول فاکتورها وارد و پیغام ظاهر شده را بررسی کنید.

```
INSERT INTO Tbl_Factors
VALUES((SELECT MAX(ID)+1 FROM Tbl_Customers),'1389/02/31')
```

شـــماره‌ای که قصد درج آن را در جدول فاکتورها داریم، از بزرگ‌ترین شـــماره‌ای که برای مشتریان در جدول Tbl_Customers وارد شـــده، یک واحد بزرگ‌تر است و بنابراین نامعتبر محسوب می‌شود. با اجرای پرس‌وجوی فوق، پیغام خطای زیر ظاهر می‌شـــود. این پیغام اعلام می‌کند به دلیل تعارض رکورد وارد شده با محدودیت ناشی از کلید خارجی، درج رکورد امکان‌پذیر نیست.

```
The INSERT statement conflicted with the FOREIGN KEY constraint “FK_Tbl_Factors_
Tbl_Customers”. The conflict occurred in database “Publisher”, table “dbo.Tbl_Customers”,
column ‘ID’.
The statement has been terminated.
```

مثال ۶ : مشخصات سه کتاب را در قالب یک پرس‌وجو وارد پایگاه داده کنید.

```
INSERT INTO Tbl_Books
VALUES
```

1 . Relationship

```
(1229,'Word 2010','علی رحیمی',115000,280),
(1230,'Excel 2010','رضا کریمی',125000,300),
(1231,'Access 2010','حمید ریاحی',100000,240)
```

مثال ۷: در نمایی که برای نگه‌داری فاکتورهای سال ۸۸ ایجاد کردیم، مشخصات یک فاکتور جدید را وارد کنید.

```
INSERT INTO View_Factors_Year_88
VALUES (4,'1388/02/15')
```

فاکتور وارد شده درون نما، در واقع درون جدول مرتبط درج شده است، چون اصولاً نماها فاقد داده هستند. به این نکته مهم دقت داشته باشید که چنان‌چه یک نما حاصل الحاق دو یا چند جدول باشد، امکان درج در آن وجود ندارد چون عملاً باید رکوردهای همه جداول تحت تأثیر قرار بگیرند.

مثال ۸: پرس‌وجویی بنویسید که رکوردهای موجود در جدول کتاب‌ها از پایگاه داده Publisher2 را خوانده و در جدول کتاب‌های پایگاه داده Publisher درج کند.

```
INSERT INTO Publisher.dbo.Tbl_Books
SELECT * FROM Publisher2.dbo.Tbl_Books
```

۲-۱۳ به روزرسانی داده‌ها

بروزرسانی داده‌ها یعنی تغییر یک یا چند فیلد از رکوردهایی که قبلاً وارد جدول شده‌اند. برای انجام این کار از دستور UPDATE استفاده می‌شود که شکل کلی آن به صورت زیر است:

```
UPDATE table_name SET field_1=new_value_1, field_2=new_value_2,…, field_n=new_value_n WHERE condition
```

مثال ۱: پرس‌وجویی بنویسید که نام نویسنده کتاب شماره ۱۲۰۵ را به «سعید غلامی» تغییر دهد.

```
UPDATE Tbl_Books SET Author='سعید غلامی'
WHERE ID=1205
```

ظاهر شدن پیغام (1 row(s) affected نشان می‌دهد بروزرسانی با موفقیت انجام شده است. اگر به جای

این عبارت، 0 row(s) affected ظاهر شود به این معنی است که کتابی با این شماره در جدول وجود ندارد.

مثـــال ۲ : قیمت همه کتاب‌های موجود در جدول کتاب‌هـــا را که حجم آن‌ها بالای ۳۵۰ صفحه است، ده درصد کاهش دهید.

```
UPDATE Tbl_Books SET Price=Price*0.9
WHERE Pages>=350
```

مثال ۳ : پرس‌وجویی بنویســـید که قیمت کتاب‌هـــای موجود در پایگاه داده را با توجه به جدول زیر تغییر دهد.

نسبت قیمت به صفحات	میزان تغییر
کم‌تر از ۱۸۰ ریال	پانزده درصد افزایش
بین ۱۸۰ ریال و ۲۱۰ ریال	پنج درصد افزایش
بیش‌تر از ۲۱۰ ریال	ده درصد کاهش

```
UPDATE Tbl_Books SET Price=
CASE
WHEN Price/Pages < 180 THEN Price*1.15
WHEN Price/Pages BETWEEN 180 AND 210 THEN Price*1.05
WHEN Price/Pages > 210 THEN Price*0.9
END
```

مثال ۴ : پرس‌وجویی بنویسید که یک ستون با نام TotalSale از نوع Smallint به جدول کتاب‌ها اضافه کند و مقدار آن را برابر با مجموع فروش هر کتاب قرار دهد. سپس پرس‌وجویی بنویسید که مشخصات ۵ کتاب پرفروش را استخراج نماید.

```
ALTER TABLE Tbl_Books ADD TotalSale smallint
GO
UPDATE Tbl_Books SET TotalSale=
(SELECT SUM(Quantity) AS Qty FROM Tbl_FactorDetails fd
```

```
WHERE Tbl_Books.ID= fd.BookID)
GO
SELECT TOP(5)* FROM Tbl_Books
ORDER BY TotalSale DESC
```

بررسی کد :

پرس‌وجوی اول، ساختار جدول را تغییر داده و یک ستون با نام TotalSale و نوع داده‌ای Smallint ایجاد می‌کند.

پرس‌وجوی دوم با استفاده از یک زیرپرس‌وجوی همبسته، جمع فروش را در این فیلد قرار می‌دهد.

سومین پرس‌وجو هم رکوردها را برحسب آمار فروش به صورت نزولی مرتب می‌کند و ۵ رکورد اول را برمی‌گرداند.

مثال ۵ : یک تابع Inline طراحی کنید که به کمک آن بتوان مشخصات تعدادی از کتاب‌های پرفروش موجود در پایگاه داده را استخراج کرد. سپس با استفاده از آن، قیمت پنج کتاب پرفروش را به میزان ده درصد کاهش دهید.

```
-- function definition
CREATE FUNCTION fn_popular_books
(@count smallint)
RETURNS TABLE
AS
RETURN
(SELECT TOP(@count) BookID,SUM(Quantity)AS Cnt
FROM Tbl_FactorDetails
GROUP BY(BookID)
ORDER BY Cnt DESC)

-- query execution
UPDATE Tbl_Books SET
Price=Price*0.9
WHERE ID IN
(SELECT BookID FROM fn_popular_books(5))
```

بررسی کد :

تابع تعریف شده، مقدار @count را دریافت نموده و به تعداد آن، کتاب‌هایی را که دارای بیش‌ترین فروش بوده‌اند در قالب یک جدول برمی‌گرداند.

پرس‌وجوی بروزرسانی، قیمت کتاب‌هایی را که شماره آن‌ها توسط تابع برگردانده شده به میزان ده درصد کاهش می‌دهد.

۱۳-۳ حذف داده‌ها

حذف رکوردها با استفاده از عبارت DELETE انجام می‌گیرد که ساختار ساده‌ای دارد اما اگر در به‌کارگیری آن دقت کافی نداشته باشید می‌تواند به داده‌های شما آسیب جدی وارد کند؛ چون برگرداندن داده‌های حذف شده تنها با استفاده از فایل پشتیبان امکان‌پذیر است و در هر صورت داده‌هایی که بعد از تهیه فایل پشتیبان وارد پایگاه داده نموده یا تغییر داده‌اید از دست خواهند رفت.

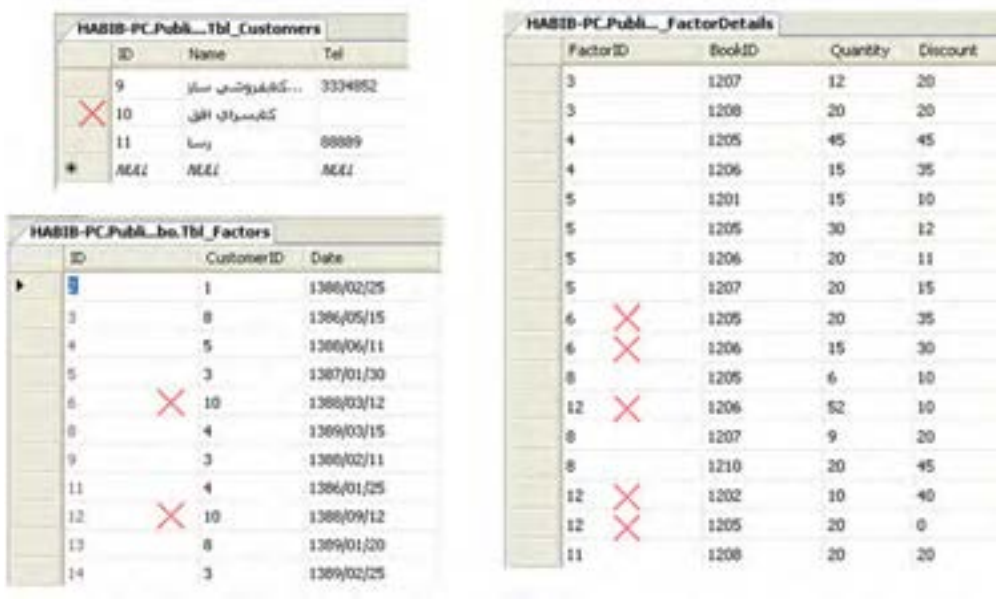
ساختار دستور DELETE به صورت زیر است :

```
DELETE FROM table_name WHERE condition
```

مثال ۱ : پرس‌وجویی برای حذف مشخصات مشتری شماره ۱۰ بنویسید.

```
DELETE FROM Tbl_Customers WHERE ID=10
```

دقت داشته باشید که هنگام برقراری روابط میان جداول، روش حذف رکوردهای مرتبط را روی حالت آبشاری یا CASCADE تنظیم کردیم و از آن‌جا که جدول فاکتورها از شماره مشتری به عنوان کلید خارجی استفاده می‌کند بنابراین با اجرای پرس‌وجوی فوق، فاکتورهای مرتبط با این مشتری هم از جدول فاکتورها حذف می‌شوند. علاوه بر این چون جدول جزییات فاکتورها از شماره فاکتور به عنوان کلید خارجی استفاده می‌کند رکوردهای فاکتورهای حذف شده هم مشمول دستور DELETE می‌شوند. بنابراین با یک دستور DELETE سه جدول مرتبط تحت تأثیر قرار می‌گیرند.



HABIB-PC.Publi...Tbl_Customers

ID	Name	Tel
9	...کتابفروشی سنا	3334852
10	کتابسرای افق	
11	رسا	88889
NULL	NULL	NULL

HABIB-PC.Publi...FactorDetails

FactorID	BookID	Quantity	Discount
3	1207	12	20
3	1208	20	20
4	1205	45	45
4	1206	15	35
5	1201	15	10
5	1205	30	12
5	1206	20	11
5	1207	20	15
6	1205	20	35
6	1206	15	30
8	1205	6	10
12	1206	52	10
8	1207	9	20
8	1210	20	45
12	1202	10	40
12	1205	20	0
11	1208	20	20

HABIB-PC.Publi...bo.Tbl_Factors

ID	CustomerID	Date
1	1	1386/02/25
3	8	1386/05/15
4	5	1386/06/11
5	3	1387/01/30
6	10	1388/03/12
8	4	1389/03/15
9	3	1386/02/11
11	4	1386/01/25
12	10	1388/09/12
13	8	1389/01/20
14	3	1389/02/25

مثال ۲: فرض کنید نحوه رفتار با دستور DELETE در جداول مرتبط روی حالت No Action تنظیم شده؛ یعنی حذف یک رکورد از جدول روی جداول مرتبط تأثیری ندارد. دستور DELETE را برای مشتری شماره ۱۰ به‌گونه‌ای بنویسید که عملاً حذف آبشاری اتفاق بیفتد.

```
BEGIN TRAN cascade_delete
DELETE FROM Tbl_FactorDetails
WHERE FactorID IN
(SELECT ID FROM Tbl_Factors
WHERE CustomerID=10)
GO
DELETE FROM Tbl_Factors
WHERE CustomerID=10
GO
DELETE FROM Tbl_Customers WHERE ID=10
COMMIT TRAN cascade_delete
```

بررسی کد:

کد را درون به یک تراکنش تبدیل می‌کنیم تا همه دستورات با هم اجرا شوند.

ابتدا رکوردها را از جدول جزییات فاکتورها، سپس از جدول فاکتورها و نهایتاً از جدول مشتریان حذف می‌کنیم.

۴-۱۳ کار با عبارت‌های OUTPUT

عبارت‌های OUTPUT، امکان دسترسی به رکوردهایی را که توسط دستورات INSERT، UPDATE و DELETE تحت تأثیر قرار گرفته‌اند فراهم می‌آورند. این عبارت می‌تواند با دو کلیدواژه زیر به کار رود :

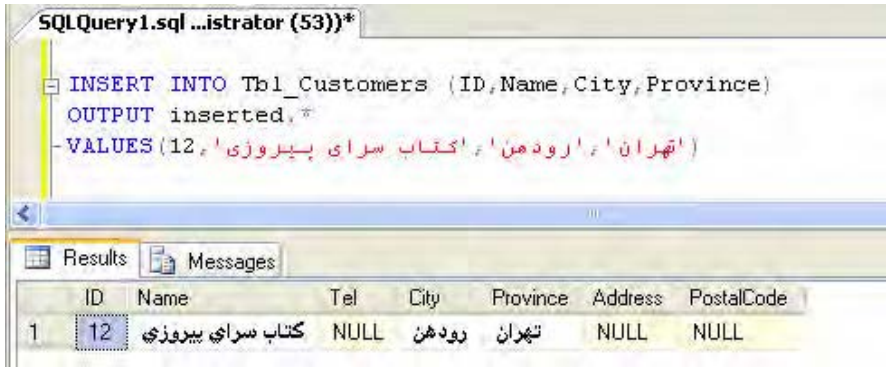
deleted : برای نشان دادن مقادیری که توسط دستور حذف یا بروزرسانی، حذف شده‌اند به کار می‌رود؛ با استفاده از این کلیدواژه مقادیر قبلی فیلدها نشان داده می‌شود.

inserted : برای نمایش مقادیری که توسط دستور درج یا بروزرسانی وارد جدول شده‌اند کاربرد دارد. استفاده از این کلیدواژه منجر به نمایش مقادیر فعلی فیلدها خواهد شد.

در واقع deleted و inserted دو جدول مجازی هستند که رکوردهای تحت تأثیر قرار گرفته پس از اجرای پرس‌وجو را نگه‌داری می‌کنند.

مثال ۱ : مشخصات یکی از مشتریان را به صورت تصادفی حذف نموده و نام مشتری حذف شده را نمایش دهید.

مثال ۲ : مشخصات یک مشتری را در پایگاه داده درج کنید و مقادیر درج شده را از جدول inserted برگردانید.



توجه داشته باشید که عبارت OUTPUT باید قبل کلیدواژه VALUES نوشته شود.

مثال ۳: پرس‌وجویی بنویسید که به ابتدای شماره تلفن مشتریان ساکن شهر تهران پیش شماره ۰۲۱ را اضافه نموده و رکوردها را قبل و بعد از بروز شدن نمایش دهد.

عبارت OUTPUT باید قبل از کلیدواژه WHERE قرار گیرد؛ در غیر این‌صورت پرس‌وجو با خطا مواجه خواهد شد.

در صورت استفاده از عبارت OUTPUT inserted.* چه نتایجی ظاهر می‌شود؟

در جدول اول هم شـــماره‌ها با پیش‌شـــماره ۰۲۱ نشـــان داده می‌شوند چون inserted مقادیر را پس از درج نمایش می‌دهد.

مثال ۴ : اگر قیمت کتاب شماره ۱۲۱۸ مقدار ۱۲۰۰۰۰ ریال باشد، پرس‌وجوی زیر چه مقادیری را برمی‌گرداند؟

```
UPDATE Tbl_Books SET Price=Price*1.5
OUTPUT deleted.Price AS LastPrice,
inserted.Price AS CurrentPrice
WHERE ID=1218
```

از جدول deleted مقدار ۱۲۰۰۰۰ ریال و از جدول inserted مقدار ۱۸۰۰۰۰ ریال برگردانده خواهد شد.

مثال ۵ : پرس‌وجویی بنویسید که مشخصات یک فاکتور را به صورت زیر وارد جدول کند :

فاکتور برای مشتری شماره ۵ به تاریخ ۱۳۸۹/۰۳/۱۵		
شماره کتاب	تعداد	تخفیف
۱۲۰۵	۲۰	۱۵
۱۲۰۶	۱۰	۵
۱۲۰۸	۱۵	۲۵

برای درج جزییات فاکتور در جدول Tbl_FactorDetails نیازمند درج شـــماره فاکتور هستیم و از آن‌جا که در جدول فاکتورها، شماره به صورت خودکار تولید می‌شود باید ابتدا شماره تخصیص داده شده به فاکتور را استخراج نموده و از آن در ثبت اقلام فاکتور استفاده کنیم.

```
CREATE TABLE TempTable (ID int)
GO

INSERT INTO Tbl_Factors

OUTPUT inserted.ID INTO TempTable
VALUES(5,'1388/03/15')
GO

INSERT INTO Tbl_FactorDetails
VALUES

((SELECT ID FROM TempTable),1205,20,15),

((SELECT ID FROM TempTable),1206,10,5),

((SELECT ID FROM TempTable),1208,15,25)
GO

DROP TABLE TempTable
```

بررسی کد:

- ابتدا یک جدول تک ستونی با نام TempTable ایجاد می‌کنیم که تنها حاوی ستون ID است تا از آن برای نگه‌داری شماره فاکتور استخراج شده استفاده نماییم.
- فاکتور را با مشخصات ذکر شده در جدول فاکتورها درج می‌کنیم و مقدار inserted.ID را که حاوی شماره تولید شده برای فاکتور است با دستور INTO به درون جدول TempTable می‌بریم.
- در جدول جزییات فاکتور، مقادیر موردنظر را درج و به‌جای شماره فاکتور، یک زیرپرس‌وجو تعریف می‌کنیم که شماره فاکتور را از جدول موقت بخواند و در جدول جزییات فاکتور درج کند.
- در پایان جدول ساخته شده را حذف می‌کنیم.

OUTPUT Clause

Returns information from, or expressions based on, each row affected by an INSERT, UPDATE, DELETE, or MERGE statement. These results can be returned to the processing application for use in such things as confirmation messages, archiving, and other such application requirements. The results can also be inserted into a table or table variable. Additionally, you can capture the results of an OUTPUT clause in a nested INSERT, UPDATE, DELETE, or MERGE statement, and insert those results into a target table or view.

An UPDATE, INSERT, or DELETE statement that has an OUTPUT clause will return rows to the client even if the statement encounters errors and is rolled back. The result should not be used if any error occurs when you run the statement.

DELETED

Is a column prefix that specifies the value deleted by the update or delete operation. Columns prefixed with DELETED reflect the value before the UPDATE, DELETE, or MERGE statement is completed.

DELETED cannot be used with the OUTPUT clause in the INSERT statement.

INSERTED

Is a column prefix that specifies the value added by the insert or update operation. Columns prefixed with INSERTED reflect the value after the UPDATE, INSERT, or MERGE statement is completed but before triggers are executed.

INSERTED cannot be used with the OUTPUT clause in the DELETE statement.

۱. متن بالا را مطالعه کرده و در مورد OUTPUT Clause توضیح دهید.

۲. ترجمه و تلفظ عباراتی را که زیر آن‌ها خط کشیده شده در فرهنگ لغات پیدا کنید.

۱. پرس‌وجویی بنویسید که یک پایگاه داده جدید با نام Publisher_New ساخته و جدول Tbl_Books_New را در آن ایجاد نماید. سپس همه رکوردهای موجود در جدول کتاب‌ها را از پایگاه داده Publisher درون جدول ساخته شده درج نماید.

۲. یک نما حاوی فاکتورهای سال ۸۹ ایجاد کنید. سپس پرس‌وجویی بنویسید تا فاکتور جدیدی به تاریخ ۱۳۸۸/۱۲/۱۱ در آن درج کند. آیا عملیات درج موفقیت‌آمیز است؟

۳. پرس‌وجویی بنویسید که یک ستون جدید با نام TotalSum به جدول فاکتورها اضافه نموده و در آن جمع فاکتور را ذخیره کند.

۴. یک تابع Inline تعریف کنید که عدد n و تاریخ d را دریافت نموده و مشخصات n فاکتوری را که دارای بیش‌ترین جمع مبلغ هستند و بعد از تاریخ d صادر شده‌اند از جدول تمرین قبل برگرداند.

۵. ستونی با نام explain به جدول کتاب‌ها اضافه کنید و مقدار آن را برابر با عبارت زیر قرار دهید :

۶. کتاب {عنوان کتاب} که توسط {نام نویسنده} تألیف گردیده با قیمت {قیمت به تومان} تومان به فروش می‌رسد.

۷. در مورد دستور TRANCATE TABLE و تفاوت آن با دستور DELETE مطالبی را از راهنمای Management Studio استخراج کنید.

۸. اگر قیمت کتاب شماره ۱۲۱۰، صدهزار ریال باشد نتیجه اجرای پرس‌وجوی زیر چیست؟

```
UPDATE Tbl_Books SET Price=Price*1.2
OUTPUT inserted.Price - deleted.Price
WHERE ID=1210
```

- Date C.J , An Introduction to Databse Systems 8th edition, Addison-Wesely, 2003
- Codd E.F , The Relational Model for Database Management, Addison-Wesely, 1990
- Dewson, Robin, SQL Server 2008 : From Novice to Professional, Apress 2008
مترجم : حبیب فروزنده دهکردی, انتشارات نقش سیمرغ, ۱۳۸۸
- Vieira, Rob, Professional Microsoft SQL Server 2008 Programming, Wiley, 2009
- Chulcher ,Clare ,Beginning SQL Queries : From Novice To Professional , Apress, 2008
- Microsoft Access 2007 Help
- Microsoft SQL Server 2008 Books online
- روحانی رانکوهی، محمدتقی، مفاهیم بنیادی پایگاه داده‌ها، انتشارات جلوه، ویراست سوم، ۱۳۸۶
- یمقانی، محمدرضا، بانک اطلاعاتی، وزارت آموزش و پرورش سازمان پژوهش و برنامه‌ریزی آموزشی