معماری پایگاه داده‌های علم سنجی

نوشته:

دکتر علی منصوری

مهندس احمد یوسفان

دکتر فرامرز سهیلی

شیوا شیردوانی

تاریخچه

مفهوم پایگاه اطلاعاتی از سال ۱۹۶۰ میلادی تکامل یافته است. در دهه ۶۰ میلادی مشکلات زیادی در طراحی، ساخت و نگهداری سیستمهای اطلاعاتی وجود داشته است، به خصوص اینکه این سیستمها برا کارایی پیچیده قابلیت نداشته یا توانایی ارائه خدمات هم زمان چند کاربر را نداشتند. پایگاه های اطلاعاتی اولیه طراحی مبتدیانه‌ای داشتند و تنها یک فرد در آن واحد می توانست از آنها استفاده نماید. تا اینکه سیر تکاملی آنها روز به روز رو به تکمیل شدن گذاشت و سیستم مدیریت پایگاه استفاده مؤثر از داده ها را فراهم نمود. در فرهنگ انگلیسی آکسفورد به نقل از یک گزارش فنی در سال ۱۹۶۲ برای اولین بار از اصطلاح «داده پایه» استفاده شده است. استفاده از این اصطلاح بیشتر در زمینه پردازنده، حافظه رایانه و ذخیره سازی رایانه است. این مفهوم به همراه مفهوم سیستمهای مدیریت پایگاه اطلاعاتی که ویرایش و تغییرات مهم را در پایگاه داده ها ممکن کرده است، رواج یافته است (نورمحمدی، ۱۳۹۳ به نقل از میداو، ۱۹۸۷).

اولین کاربرد اصطلاح پایگاه اطلاعاتی به ژوئن ۱۹۶۳ بازمی گردد،یعنی زمانی که شرکتی با نام System Development Coمسئولیت اجرایی یک طرح به نام «توسعه و مدیریت محاسباتی یک پایگاه داده‌ای مرکزی» را برعهده گرفت. اصطلاح «پایگاه داده» به عنوان یک واژه واحد در اوایل دهه ۷۰ در اروپا و در اواخر ۷۰ در خبرنامه های معتبر امریکایی به کار رفت. البته واژه «پایگاه داده» تقریباً اوایل ۱۹۶۶ در روزنامه واشنگتن پست نیز به کار رفت (ونگ و زانیولو، ۲۰۰۵).

اولین سیستم مدیریت پایگاه اطلاعاتی در دهه ۶۰ گسترش یافت. از پیشگامان این شاخه چالز باخمن است (نورمحمدی، ۱۳۹۳ به نقل از باخمن، ۱۹۷۳). مقالات باخمن بر این موضوع تأکید داشت و نشان داد که فرضیات او کاربرد بسیار مؤثرتری برای دسترسی به تجهیزات و وسایل ذخیره سازی را مهیا می کند. در آن زمان پردازش داده بر پایه کارت های منگنه و نوارهای مغناطیسی بود. این امر موجب می شد تا پردازش اطلاعات به صورت دسته ای فراهم شود. در آن ایام دو نوع مدل داده ای وجود داشت؛ یکی به صورت مدل توصیه شده کنفرانس زبانهای سیستم داده‌ای که موجب توسعه مدل شبکه ای شد و ریشه در نضریات باخمن داشت و دیگری مدل سلسله مراتبی که توسط مؤسسه راکول ایجاد شد و بعدها شرکت آی بی ام با استفاده از آن محصول آی ام اس را تولید نمود.

در سال ۱۹۷۰ مدل دیگری به نام «مدل رابطه ای» توسط شخصی به نام ادگار اف کد ارائه شد. او در این مدل سایر مدل ها را مورد انتقاد قرار داد. این مدل تا مدت های نسبتاً طولانی مورد تأیید مجامع علمی بود. د رسال ۱۹۸۰ کار بر روی پایگاه های مدل توزیع شده و ماشین های پایگاهی متمرکز شد،اما تأثیر زیادی در عمل بوجود نیاورد. در دهه ۹۰ میلادی بیشتر توجهات به مدل شئ گرا گرایش پیدا کرد. این مدل بر روی برخی پایگاه داده های خاص و داده های چند رسانه ای به خوبی کار میکرد. در این اثنا اتفاق خاصی که تأثیر زیادی بر پایگاه اطلاعاتی داشته باشد نیافتاد تا اینکه در سال ۲۰۰۰ نوآوری دیگری در زمینه پایگاه بوجود آمد و آن ایجاد پایگاه بر اساس زبان XML بود که هدف اصلی این مدل از بین بردن تفاوت بین مستندات و داده ها بود و کمک می کرد تا منابع اطلاعاتی چه به صورت ساختارمند و چه بدون ساختار در کنار هم مورد استفاده قرار گیرد (ون و زانیولو، ۲۰۰۵).

تکنولوژی پایگاه داده‌ها از اواسط دهه شصت میلادی و به منظور توسعه سیستم های فایلینگ ایجاد شدند. طول دهه هفتاد مدلهای پایگاه های داده سلسله مراتبی و شبکه ای توسعه یافته و مورد استفاده زیادی قرار گرفتند. در اوایل دهه هشتاد، شاخه ای از آن تکنولوژی به نام سیستم مدیریت پایگاه داده‌های رابطه ای مورد توجه بیشتری قرار گرفت و به عنوان تکنولوژی برتر شناخته شد و هم اکنون هم بسیار مورد استفاده قرار می گیرد.

از اواسط دهه هشتاد تا کنون، سیستم های دیگری نیز ایجاد و عرضه شد از جمله:

- سیستم فایلینگ معمولی داده‌ها
- سیستم مدیریت داده‌ها و جستجوها
- سیستم مدیریت پایگاه ها (پایگاه داده)

تعریف پایگاه اطلاعاتی

پایگاه های داده ها معمولاً در قالبی که برای دستگاه ها و رایانه ها قابل خواندن و دسترسی باشد ذخیره می شوند. البته این شیوه ذخیره سازی اطلاعات تنها روش موجود نیست و شیوه های دیگری مانند ذخیره سازی

ساده در پرونده ها نیز استفاده می گردد. آنچه ذخیره سازی داده ها در پایگاه های داده ها را مؤثر میسازد وجود یک ساختار مفهومی برای ذخیره سازی و روابط بین داده هاست (والدمن، ۲۰۰۳؛ لیو، ۲۰۰۹).

پایگاه داده در اصل مجموعه ای سازمان یافته از اطلاعات است. این واژه از دانش رایانه سرچشمه می گیرد، اما کاربرد وسع و عمومی نیز دارد. این وسعت به اندازه ایست که مرکز اروپایی پایگاه اطلاعاتی (که تعارف خردمندانه ای برای پایگاه اطلاعاتی ایجاد می کند) شامل تعاریف غیر الکترونیکی برای پایگاه اطلاعاتی می باشد(والدمن، ۲۰۰۳؛ لیو، ۲۰۰۹).

پایگاه اطلاعاتی مجموعه ای از رکوردهای ذخیره شده در رایانه با یک روش سیستماتیک (اصولی) مثل یک برنامه رایانه ای است که می تواند به سؤالات کاربر پاسخ دهد. برای ذخیره بازیابی بهتر، هر رکورد معمولاً به صورت مجموعه ای از اجزای داده ای یا رویدادها سازماندهی می گردد. بخش های بازیابی شده در هر پرسش به اطلاعاتی تبدیل می شود که برای اتخاذ یک تصمیم کاربرد دارد. برنامه ای رایانه ای که برای مدیریت و پرسش و پاسخ بین پایگاه های داده ای استفاده می شود، مدیریت سیستم پایگاه داده ای نامیده می شود که از خصوصیات اصلی برای طراحی سیستم های پایگاه داده ای در علم اطلاعات می باشد(والدمن، ۲۰۰۳؛ لیو، ۲۰۰۹).

پایگاه اطلاعاتی مجموعه ای از رکوردها یا تکه هایی از یک شناخت است. نوعاً در یک پایگاه اطلاعاتی توصیف ساختاریافته ای برای موجودیت های نگهداری شده در پایگاه اطلاعاتی وجود دارد: این توصیف با یک الگو یا مدل شناخته می شود. مدل توصیفی، اشیاء پایگاه های اطلاعاتی و ارتباط بین آن ها را نشان می دهد. روش های متفاوتی برای سازماندهی این مدل ها وجود دارد که به آن ها مدل های پایگاه اطلاعاتی می گوییم. امروزه مدل رابطه ای پرکاربردترین مدل است که تعریف عموم یآن عبارت است از: نمایش تمام ازطلاعات به صورت جداول مرتبط که از سطرها و ستون ها تشکیل شده است. در این مدل وابستگی ها به کمک مقادیر مشترک در بیش از یک جدول نشان داده می شود. مدل های دیگری مثل مدل سلسه مراتب و مدل شبکه ای به طور صریح تری ارتباطات را نشان می دهند(والدمن، ۲۰۰۳؛ لیو، ۲۰۰۹).

در اصطلاحنامه کتابداری در تعریف پایگاه اطلاعاتی آمده است:«واحدی از سوابق و بایگانی های قابل خواندن با ماشین که برای یک کاربرد واحد تعبیه نشده، بلکه به منزله یک مجموعه متجانس برای مقاصد مختلف به کار می رود» (سلطانی و راستین، ۱۳۶۵).

پایگاه اطلاعاتی مجموعه ای است از داده های ذخیره شده در مورد انواع موجودی هایی که محیط عملیاتی و ارتباط های آن ها به صورت مجتمع و مبتنی بر یک ساختار، تعریف شده است، که به صورت صوری با حداقل افزونگی، تحت کنترل متمرکز می باشد و مورد استفاده یک یا چند کاربر به طور اشتراکی و همزمان قرار می گیرد (روحانی رانکوهی، ۱۳۷۲).

سیستم های پایگاه داده

پایگاه داده چیست؟ اساساً پایگاه داده چیزی بیش از مجموعه ای از اطلاعات موجود طی یک بازه زمانی طولانی، معمولا سالهای زیادی نیست. به عبارت عام تر اصطلاح پایگاه داده اشاره دارد به مجموعه ای از داده‌ها که توسط سیستم مدیریت پایگاه داده مدیریت می شوند.

پایگاه داده را می توان یکی از انواع «سیستم های ذخیره و بازیابی اطلاعات» محسوب کرد.در این نوع سیستم ها به کاربر اجازه داده می شود داده‌های مورد نظر خود را ذخیره نمود و ضمن انجاک 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 استفاده می شد. در این سیستم، اطلاعات وارد شده در برنامه، درون یک یا چند فایل نوشته می شد و برنامه نویس مجبور بود الگوریستم های مورد نیاز برای مرتب سازی یا جستجوی داده‌ها را درون نرم افزار پیاده سازی کند.

با وجود اینکه هنوز هم برای تولید برنامه های کوچک از سیستم فایلینگ استفاده می شود اما تولید نرم افزارهای 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. بهره گیری از یک «سیستم مدیریت پایگاه داده» باعث ایجاد تمرکز در ذخیره سازی اطلاعات می شود و در صورت طراحی صحیح، افزونگی و ناسازگاری اطلاعات از بین می برد. به عنوان مثال، اطلاعات یک مقاله تنها یکبار و با رعایت استانداردی مشخص ذخیره خواهد شد. علاوه بر این، ذخیره سازی و بازیابی اطلاعات برعهده این سیستم گذاشته می شود و برنامه نویس مجبور نست مانند روش فایلینگ، برای ایجاد یا تغییر فایل ها، اضافه کردن رکورد و جستجو و مرتب سازی، کدهای اضافی بنویسد.

با توجه به مطالب ذکر شده می توان پایگاه داده را به این صورت تعریف کرد: «مجموعه ای سازمان یافته و بدون افزونگی از داده‌های مرتبط با هم که در چارچوب یک مدل داده ای و تحت کنترل یک سیستم متمرکز قرار دارند.» (دهکردی، ۱۳۹۴)

سیستم مدیریت پایگاه داده

اولین سیستم مدیریت پایگاه داده به صورت تجاری در اواخر دهه ۱۹۶۰ میلادی ظاهر شدند. این سیستم ها تکامل یافته سیستم های فایلینگ بودند.

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 می نامیم؛ یعنی دروازه ای که هرگونه دسترسی به داده‌ها و تغییر اطلاعات باید از طریق آن صورت گیرد. در واقع، پایگاه داده و فایل های آن تحت کنترل نرم افزار DBMS هستند و حتی برنامه نویس، درخواست های خود را از طریق کد به آن ارسال نموده و تنها این سیستم است که می داند چگونه داده‌ها را درون فایل های پایگاه داده ذخیره و بازیابی کند و در صورت نیاز تغییر دهد.

در حال حاضر Access، MySQL، SQL Server، Oracle از سیستم های مدیریت پایگاه داده پرکاربرد در دنیا محسوب می شوند و افراد و سازمان ها بسته به میزان کارایی و هزینه ای که می خواهند پرداخت کنند، یکی از آن ها را به عنوان منبع ذخیره سازی و مدیریت داده‌های خود انتخاب می نمایند.

یک سیستم مدیریت پایگاه داده باید:

۱. به کاربران اجازه دهد پایگاه داده جدید ایجاد کند و ساختار منطقی داده خود را مشخص کنند و زبان تعریف داده مخصوص به خود را بکار ببر.
۲. برای کاربران توانایی جستجو و تغییر داده، با استفاده از زبانی مناسب که معمولاً زبان کوئری نامیده می شود را فراهم کند.
۳. ذخیره جم زیادی از داده‌ها را پشتیبانی نماید.

۴. دسترسی به داده ها را برای تعداد زیادی از کاربران به صورت همزمان کنترل کند. (Chapter 1)

مفاهیم تشکیل دهنده پایگاه داده

داده

اصطلاح داده در مفهوم کلی عبارت است از نمایش ذخیره شده کلیه وجودیت‌ها، واقعیات و رخدادها که در تصمیم‌گیری به‌کار می‌آیند.

داده از جمله مفاهیمی است که تعاریف متعددی برای آن در اذهان وجود دارد و معمولاً هر کسی یا هر متخصصی به زعم خود از آن تعریفی دارد. ولی به طور کلی، می توان همه دانسته ها، آگاهی ها، داشته ها، آمارها، شناسه ها، پیشینه ها و پنداشته ها را داده نامید. مثلاً در مقاله «داده چیست یا داده ها چی هستند؟» از ردمن (۱۳۸۱) بحث اصلی طرح چشم اندازهایی برای راهنمایی اطلاع رسانان در بهبود کیفیت داده است. در این مقاله بیشتر به ماهیت پویای داده توجه شده و سه فعالیت عمده برای آن در نظر گرفته شده است: تولید، ذخیره و کاربرد داده (ردمن، ۱۳۸۱).

در هر صورت برای ثبت ودرک مشترک هر واقعیت و پدیده باید از نشانه های ویژه ای استفاده کرد و از آنها بهره گرفت.

«داده» به اعداد، حروف و علائمی که موجب درک انسان ها از رایانه می شود اطلاق می گردد. این درک مشترک از طرف انسان ها به صورت حروف، اعداد، و علائم مشخص می شود و در رایانه ها به صورت علائمی است که در قالب نمادهایی همانند علائم مثبت و منفی یا رمزهای صفر و یک به صورت قراردادی ارائه می شود. معمولاً «داده» یک عبارت نسبی است؛ به این معنا که اگر به صورت کامل قابل درک و فهم باشد به عنوان اطلاعات به کار گرفته می شود و در واقع به آن داده پردازش شده نیز میگویند. حال چنانچه موجب درک و فهم کامل نگردد به عنوان داده یا اطلاعات خام به شمار می آید، زیرا هدف نهایی داده این است که توسط انسان ها مورد استفاده قرار گیرد. زمانی داده قابلیت کاربرد دارد که پردازش لازم در آن صورت گرفته باشد. به عبارت دیگر، توسط انسان دستکاری شود و در رایانه مورد تجزیه و تحلیل قرار گیرد و به صورت خروجی ارئه گردد.

در اینجا منظور از دستکاری یا پردازش داده ها، انجام عملیات لازم بر روی داده است. در صورت لزوم باید عملیات ریاضی روی داده ها صورت گیرد یا با سایر موراد موجود مقایسه شود. در واقع به صورتی باشد که حقایق را برای کاربر روشن نماید و قابلیت کاربرد پیدا کند. یعنی با حواس مختلف قابل درک باشد و برای رایانه نیز از طریق ورودی و خروجی رایانه قابل انتقال و بازیابی باشد. زمانی که صحبت از پردازش داده ها می شود، یعنی داده ها باید به نوعی تغییر کند. در پردازش داده ها (داده پردازی) در رایانه ابتدا داده ها به رایانه وارد می شود و پس از ذخیره در رایانه بر روی آن ها عملیاتی صورت می گیرد. پس از اینکه این عملیات (پردازش) صورت گرفت معمولاً داده ها به صورت اطلاعات دوباره به انسان ها منتقل می شود. بنابراین در اغلب گزارش ها و یادداشت های سازمانی و سایر موارد داده ها به چشم می خورند. برای نمونه، تاریخ و تعداد منابع کتابخانه ای که باید به کتابخانه برگشت داده شوند نمونه هایی از داده ها هستند (نورمحمدی، ۱۳۹۳).

اطلاع

هر نوع داده پردازش شده (ساخت یافته) را اطلاع می‌نامند. این تعریف یک تعریف بسیار ساده است که بیانگر تفاوت بین دو اصطلاح داده و اطلاع است. به طور کلی می‌توان گفت اطلاع مجموعه داده هایی است که در تصمیم گیری بکار می‌روند. (آیت و فراهی، پایگاه داده)

اطلاع معنایی است که انسان از طریق قراردادهای شناخته شده ای که در نمایش داده به کار می روند به داده منتسب می کند. جلال مساوات (۱۳۶۲) در خصوص اطلاع چنین می گوید:

۱. متخصصان و صاحب نظران از دیدگاه های مختلف به موضوع نگریسته اند. بعضی از دانشمندان و کارشناسان نیز به اقتضای رشته تخصصی خود، برخی از ابعاد مسأله را برجسته کرده و جنبه های دیگر را کمتر مورد توجه قرار داده اند و بدین گونه هر یک از متخصصان یا هر گروه از آنان متناسب با خواست ها و علایق خود برای اطلاع مفهوم سازی کرده اند.
۲. مفاهیمی که تا کنون در خصوص اطلاع ارائه شده اند بسیار کلی یا بسیار محدود هستند. آنهایی که بسیار کلی هستند غالباً غیر دقیق می باشند و آنهایی که محدود هستند تنها به یک یا چند قلمرو علمی قابل اطلاق اند.

۳. در دهه های اخیر دانش اطلاع رسانی و دکومانتاسون گسترش سریع و اهمیت فوق الاده ای یافته است. سازمان های دکومانتاسیون و اطلاع رسانی تقریباً در تمام کشورهای جهان بوجود آمده اند. اهمیت تحقیق در مبانی دانش اطلاع رسانی و نقش سازمان ها و مراکز اطلاع رسانی و دکومانتاسیون در پیشرفت های علمی، فنی، اقتصادی، اجتماعی و فرهنگی همه جا مورد تأیید و تأکیید قرار گرفته است.

اشمال در همین مورد می گوید: «در جهان امروز مباحث و مسائل مربوط به اطلاع رسانی و دکومانتاسیون برای جامعه بشری، درست به اندازه مسائلی نظیر مسأله مواد خام یا مسأله انرژی حائز اهمیت است». همچنین می گوید:«تکنیک جدید بدون وجود سازمان های اطلاع رسانی و دکومانتاسیون هرگز پیشرفت های امروزی را نداشته است».

درکل می توان اطلاع را به این صورت تعریف کرد: هر منبعی که تصمیم گیری را آسان نماید و بر اطلاعات گذشته ما بیفزاید. همچنین هر گونه پردازش بر روی داده اولیه را اطلاع گویند (نورمحمدی، ۱۳۹۳).

پایگاه داده ای رابطه ای

مفاهیم پایه‌ی پایگاه داده‌ی رابطه‌ای

انسان‌ها از دیرباز می‌پنداشتند با گردآوری،نگهدای و پالایش اشیاء و اطلاعاتی از پیرامون خود زندگی بهتری خواهند داشت. کمک گرفتن از حافظه‌ی انسان ساده‌ترین نوع نگهداری اطلاعات بوده است ولی فراموش و از یادبردن برخی جزییات باعث شد تا از نوشتن کمک گرفته شود. برای نمونه نوشتن فهرست عضوهای کنونی کتابخانه می‌تواند به صورت زیر باشد.

- »کوروش» کاظمی» عضو شماره‌ی «۲» کتابخانه است و به موضوع «هنر» علاقه‌مند است.
- »زیبا» «سروش» عضو شماره‌ی «۳» کتابخانه است و به موضوع «ریاضی» علاقه‌مند است.
- »علی» «انتشاری» عضو شماره‌ی «۵» کتابخانه است و به موضوع «شعر» علاقه‌مند است.
- »کامران» «خداپرستی» عضو شماره‌ی «۶» کتابخانه است و به موضوع «داستان» علاقه‌مند است.
- »کیانوش» «محمدی» عضو شماره‌ی «۷» کتابخانه است و به موضوع «شیمی» علاقه‌مند است.
- »آیتا» «کمالی» عضو شماره‌ی «۹» کتابخانه است و به موضوع «رایانه» علاقه‌مند است.

اغلب این نام‌ها در برگه‌های متفاوت و پراکنده و بدون ارتباط با همدیگر نگاشته می‌شدند و نگهداری آن دشوار بود. از سوی دیگر این نام‌ها و شماره‌ها برای کتابدار اطلاعات بیشتری به همراه دارد که در اثر برخورد با این افراد و رفتارهایشان به دست آمده است. یافتن موضوع مورد علاقه‌ی یک فرد در این جمله‌ها یکسان نیست. دقت کنید که در جمله‌های بالا کوشش شده است ساختار نسبتا یکسانی همراه با تعداد داده‌های یکسانی نوشته شود و فقط دربردارنده‌ی نام و شماره‌ی عضویت و موضوع مورد علاقه‌ی هر عضو باشد. نخستین گام برای بهبود این وضعیت ساخت یک دفتر ویژه‌ی نگهداری نام عضوها یا یک جدول (۱) از نام عضوها است که می‌تواند به صورت زیر باشد و به سادگی نیز برای همه قابل فهم است و جستجو در آن ساده‌تر است. این ساخت یافتگی و یکنواخت شدن داده‌ها نقطه‌ی آغازین مدل رابطه‌ای و پایگاه داده‌های رابطه‌ای است.

جدول ۱: نام و نشانی اعضاء

موضوع مورد علاقه	شماره‌ی عضویت	نام خانوادگی	نام
هنر	۲	کاظمی	کوروش
ریاضی	۳	سروش	زیبا
شعر	۵	انتشاری	علی
داستان	۶	خداپرستی	کامران
شیمی	۷	محمدی	کیانوش
رایانه	۹	کمالی	آیتا

همچنین دقت کنید که نام جدول نیز معنایی ضمنی دربردارد یعنی این که در این جدول فقط نام کسانی را نگهداری می شود که اکنون عضو کتابخانه هستند و نام کسانی که عضو نیستند در این جدول نیست حتی نام کسانی که پیش از این عضو کتابخانه بوده‌اند ولی اکنون عضو نیستند نیز در این جدول نیست و احتمالا از آن حذف شده است. روشن است که نگهداری، ویرایش و دیگر کارها در این جدول ساده‌تر از جمله‌ها است. این جدول نمایش کوچکتری از داده‌های موجود در جمله‌ها را در خود دارد. همچنین ستون‌های دیگری می‌تواند به سادگی به این جدول افزوده شود و اطلاعات بیشتری از یک عضو در آنها نگهداری شود. برای نمونه جدول می‌تواند

ستون‌هایی برای شغل، تاریخ تولد، شهر، زمان ثبت نام و همانند آن را دربرداشته باشد. به این ترتیب دفترهای ویژه‌ای برای کارهای گوناگون آماده شد که امروز هنوز کاربرد فراوانی دارند. اطلاعات یک جدول فراتر از داده‌های درونِ آن است؛ برای نمونه از جدولِ عضوها می‌توان موضوعی را یافت که که علاقه‌مندان بیشتری دارد یا تعدادِ علاقه‌مندان به هر موضوع را یافت. اگر ستونِ شغل به این جدول افزوده شود شاید بتوان ارتباط‌هایی میان شغل و موضوعِ موردِ علاقه یافت. بنابراین به تعریف دقیق‌تری از داده و اطلاعات نیازمندیم.

همانطور که عنوان شد داده یا داده‌ی خام نمایش قراردادی و کمّی (اندازه‌پذیر) از چیزی در گیتی است که اغلب به همراهِ خود و در کنار دیگر داده‌ها معنایی را دربردارد. برداشت و تفسیرِ مجموعه‌ای از داده‌ها، اغلب در کنار یکدیگر، اطلاعات را به وجود می‌آورد. مجموعه‌ای از اطلاعات به هم پیوسته و قرار گرفته در یک گروه یک علم را به وجود می‌آورد. در جدول بالا تعدادی داده وارد شده است که اگر در کنار هم قرار گیرند اطلاعاتی را به مخاطب ارائه می کنند. به عنوان مثال مجموعه داده های "رایانه"، "۹"، "کمالی"، "آیتا" به طور مستقل شاید نتوانند اطلاعاتی قابل تفسیر رابه مخاطب ارائه نمایند، اما اگر این داده ها به صورت جمعی کنار هم تفسیر شوند، دارای اطلاعاتی مفید خواهند بود. بنابراین" دانشجویی به نام آیتا کمالی با شماره عضویت ۹ علاقه مند به حوزه رایانه است" می تواند اطلاعاتی را به کتابدار در راستای تصمیم گیری ارائه نماید.

جدول و پایگاه داده

یک جدول از تعدادی سطر(رکورد، چندتایی) و ستون(فیلد، ویژگی، حوزه) تشکیل شده است. اغلب درسر تیتر جدول نام ستون‌ها در سطری ذکر می‌شود. تعداد سطرها عدد اصلی (cardinality) و تعداد ستون‌ها درجه‌ی جدول نامیده می‌شود. به عبارت دیگر کاردینالتی در هر ستون مقدارهایی از یک نوع قرار می‌گیرد به عبارت دیگر مقدارهای یک ستون متعلق به یک دامنه‌ی خاص از انواع هستند برای نمونه نوع عددی، رشته‌ای یا تاریخ از جمله دامنه‌های مهم هستند. هر دامنه نامی مانند integer دارد و در عنوان ستون نام دامنه افزون بر نام ستون نیز گذاشته می‌شود.

برای جدول‌های پایگاه داده می‌توان شرط‌های زیر را نیز در نظر گرفت:

- سطرها ترتیب ندارند.

- ستون‌ها ترتیب ندارند.
- سطر تکراری در جدول وجود ندارد.

دامنه و نوع

مقدارهای درون جدول همراه با نام ستون (ویژگی) و نوع (دامنه) ستون هستند و به این صورت این مقدارها معنا پیدا می‌کنند.

برای نمونه عدد ۲ بدون توجه به این که در کجا به کار برده شود معناهای گوناگونی می‌یابد مانند امانت ۲ کتاب، ۲ کتابدار کتابخانه! بنابراین دانستن فقط مقدار یا واحد آن کافی نیست بلکه متعلق بودن آن به یک موضوع خاص به آن معنا می‌بخشد. ۲۰ تومان در جایی به معنای ۲۰ تومان بهای مکالمه‌ی درون شهر به ازای هر دقیقه باشد. در جایی دیگر ۲۰ تومان می‌تواند جریمه‌ی دیرکرد برگرداندن کتاب به ازای هر هفته دیرکرد باشد و به همین ترتیب ۲۰ تومان می‌تواند در جاهای گوناگون معناهای متفاوتی داشته باشد. روشن است که مقایسه یا تفریق یا جمع دو عدد از دو دامنه‌ی متفاوت معنادار نیست هر چند هر دو واحد یکسانی داشته باشند.

بنابراین می‌توان این گونه پنداشت که مقدار هر خانه از جدول همراه با دامنه‌ی آن مقدار و نام آن مقدار است. پس در کارکردن با مقدارهای درون جدول‌ها باید دقت نمود و هر جا نیاز شد باید تبدیل نوع صریح انجام شود.

دقت شود که در زبان SQL دامنه Domain مفهوم متفاوتی نسبت به نوع type است ولی اکنون برای توضیح مفهوم‌های پایه نیازی به تفاوت گذاشتن میان آنها نیست.

درجه یک رابطه: تعداد ویژگی ها یا صفات یک جدول را درجه آن جدول یا رابطه می نامند. در جدول مقاله درجه این جدول برابر است با: ۸

کاردینالیتی یک رابطه: تعداد تاپلهای یک جدول در هر لحظه را کاردینالیتی آن جدول می نامند مثلا در جدول مقاله اگر ۴ مقاله ثبت شده باشد کاردینالیتی این جدول ۴ است و هر لحظه ممکن است تغییر کند.

پایگاه داده

پس از ساخته شدنِ رایانه‌ها یکی از کارهایی که به دوشِ آنها گذاشته شد نگهداری داده‌ها بود. این داده‌ها بر روی رسانه‌های گوناگونِ نگهداری داده‌ها مانند کارتِ منگنه، نوار، دیسک‌های مغناطیسی، نوری و الکتریکی ذخیره شده و می‌شوند. یکی از پایه‌ای ترین ویژگی‌هایی که یک رسانه‌ی نگهداری داده باید داشته باشد پایداری آن است. بسیاری از دفترها و پرونده‌هایی (file) که پیش از این به طور کاغذی نگهداری می‌شد اکنون درونِ رایانه‌ها گذاشته شده‌اند. با پیدایش نخستین سیستم‌های عامل(operating system)، سامانه‌ی پرونده (file system)به عنوان بخشی از سیستم عامل برای ساده‌تر شدنِ کارهای نگهداری، دسته‌بندی، بازیابی، حذف، جابجایی و ویرایش پرونده‌ها گسترش یافت.

برای پاسخ به گرایش و نیاز (گاهی ساختگی و گاه واقعی) به رایانه برای نگهداری و پردازش داده‌ها، برنامه‌های کاربردی و نرم افزارهای گوناگون ساخته شد. یکی از مهم‌ترین بخش‌های این نرم‌افزارها مجموعه‌ی داده‌های ذخیره شده‌ی آن است. به مجموعه داده‌های پایدار که در برنامه یا برنامه‌های کاربردی گوناگون به کار گرفته می‌شوند؛ پایگاه داده گفته می‌شود.

منظور از پایداری در پایگاه داده بدین معنا است که نوع داده‌های درون پایگاه داده با داده‌های ناپایداری همچون داده‌های ورودی و خروجی، دستورهای کنترلی، صف‌ها، پشته‌ها، بخش‌های کنترل نرم‌افزار، داده‌های میانی و به طور کلی داده‌های گذرا متفاوت است و با روشن یا خاموش شدن رایانه داده‌های درونِ پایگاه داده از میان نمی‌روند. هنگامی که داده‌ای درون پایگاه داده گذاشته شد فقط در صورتی از آن حذف می‌شود که درخواستی برای حذفِ آن داده شود. با گسترش روزافزون داده‌ها و افزایشِ ارزشِ داده‌ها، ارزشِ پایگاه‌های داده نیز افزایش یافت و برای کار با داده‌ها روال‌ها و تابع‌های کتابخانه‌ای گوناگونی فراهم شد.

فرایند ایجاد پایگاه داده

شناخت محیط عملیاتی

برای ثبت اطلاعات و داده های مربوط به یک موسسه، مکان، بخش در قالب رایانه و به طور خاص در پایگاه داده لازم است که ابتدا آن موسسه، مکان یا بخش از جهات هدف، رسالت ها و نیازهای اطلاعاتی شناسایی شود. به عبارت دیگر برای ایجاد پایگاه اطلاعاتی نیاز است که ابتدا محیط عملیاتی شناسایی شود. در همین راستا بایستی نیازها و خواسته ها به خوبی روش شود و تمامی افراد و متغیرهایی که در انجام دادن نیازها و برآورده ساختن خواسته ها نقشی خواهند داشت، به روشنی تعیین شود.

به عبارتی برای جایگزین کردن روندِ دستی نگهداری داده‌ها درون دفترها با یک نرم افزار رایانه‌ای و به کارگیری پایگاه داده‌ها برای نگهداری داده‌ها، باید در آغاز نیازها و خواسته‌ها به خوبی روشن شوند. باید کاری که بناست به کمکِ رایانه انجام شود؛ به خوبی شناخته شود. برای روشن شدنِ این روند، مسأله‌ی بسیار ساده‌ای گفته می‌شود و فشرده‌ی روندِ دستی انجامِ آن گفته می‌شود. سپس به چگونگی تبدیلِ آن به پایگاه داده‌ها پرداخته می‌شود. در این کتاب پایگاه داده‌ی بسیار ساده شده‌ای از یک کتابخانه گفته می‌شود تا با سادگی بیشتر بتوان مفهوم‌های وابسته به پایگاه داده را بر پایه‌ی آن آموخت. بیشتر نمونه‌های این کتاب بر پایه‌ی این پایگاه داده‌ی ساده است و در جای خود و بر پایه‌ی نیاز، این پایگاه داده گسترش می‌یابد و یا اینکه پایگاه داده‌های دیگری گفته شده و به کار گرفته می‌شود.

به تعدادی جدول که با هم داده‌های یک کاربرد ویژه را نگهدای می‌کنند پایگاه داده گفته می‌شود. در واقع این تعریف کامل نیست زیرا امروزه در پایگاه داده اطلاعات بیشتری همچون دیدها، تابع‌ها و همانند آن نیز نگهداری می‌شود ولی اکنون برای آشنا شدن با مفهوم‌های پایه ناچار هستیم تا بسیاری از جزییات را نادیده بگیریم.

چند تعریف در محیط اطلاعاتی

در اینجا چند تعریفِ پایه نوشته شده است و همزمان برخی از دیگر دشواری‌های کارِ مستقیم با پرونده‌ها نوشته شده است.

فیلد

یک حرف (یا نویسه) یا تعدادی حرف است که می‌توان معنای ویژه‌ای از آن برداشت کرد. در حالتِ دفتری به هر کدام از درایه‌های (خانه‌های) درون جدول فیلد گفته می‌شود. همچنین گاهی نیز به نام یک ستون (عنوان ستون) فیلد گفته می‌شود. برای نمونه گفته می‌شود «فیلدِ شماره‌ی کتاب» ولی اغلب «فیلدِ شماره‌ی کتاب از یک کتابِ با شماره‌ی ویژه» (از یک سطرِ ویژه) در نظر است.

رکورد

تعدادی فیلد که روی هم می‌توان معنایی از آنها برداشت کرد. با کمی ساده انگاری در حالتِ دفتری می‌توان سطرها را همان رکوردها دانست که می‌توان جمله‌ای بر پایه‌ی آن ساخت و مفهوم کاملی را می‌رساند.

پرونده

تعدادی رکورد که ارتباط منطقی با هم دارند و کنار هم نگهداری می‌شوند. با چشم پوشی از برخی تفاوت‌ها پرونده‌ها را می‌توان همان دفتر در نظر گرفت که در رایانه به صورت پایدار ذخیره شده است. نیازی نیست پرونده‌ها همانند دفترها دارای سطر و ستون باشند به گونه‌ای که کاربر واقعا به صورت دفتر بتواند آنها را ببیند. برای نمونه نیازی نیست پرونده‌ها متنی باشند و در هر سطر پرونده‌ی متنی یک سطر از دفتر گذاشته شود بلکه اغلب در حالتِ دودویی پرونده‌ها نگهداری می‌شوند و دیدنِ داده‌های درونِ پرونده بدون به کارگیری برنامه‌ی وابسته به آن چندان ساده نیست و نمی‌توان داده‌های درونِ آن را به طورِ دلخواه مانند دفتر دید.

موجودیت:

برای انجام اهداف و انتظارات محیط عملیاتی متغیرهای زیادی از جمله عوامل انسانی و غیرانسانی دخیل هستند که به آنهاموجودیت گفته می شود. به عنوان مثال برای ایجاد یک پایگاه اطلاعاتی استنادی متغیرهایی از قبیل نویسنده، مقاله، مقالات استناد دهنده، مقالات استناد شونده نقش دارند. آنچه باید توجه داشت این است که

موجودیت به عنوان یک مفهوم یا واقعیت مد نظر است. در مثال فوق منظور از مقاله یک مفهوم و یک واقعیت به نام مقاله است و نه اینکه مقاله خاصی و با نویسنده مشخص مد نظر است. بنابراین در مثال مذکور مقاله به عنوان یک مفهوم وجودی موجودیت نام دارد.

پایگاه داده(database)

مجموعه‌ای از داده‌ها و فراداده‌های در ارتباط با هم که بر روی رسانه‌ای پایدار نگهداری شوند. برای نمونه به چند پرونده‌ای که برای نگهداری داده‌های کتاب‌ها، عضوها و امانت‌ها برای کتابخانه در کنار هم که اغلب درون یک پوشه گذاشته می‌شوند؛ پایگاه داده‌ی کتابخانه گفته می‌شود. این که چه تعداد پرونده برای نگهداری داده‌ها و فراداده‌ها به کار گرفته شود و اینکه کجا و چگونه این پرونده‌ها و داده‌های درونِ آنها ذخیره شوند؛ شکل‌های گوناگونی دارد و به عامل‌های گوناگونی بستگی دارد. در حالت برنامه نویسی مستقیم با پرونده‌ها به چگونگی برنامه نویسی بستگی دارد.

برنامه‌ی کاربردی(application)

برنامه‌ی رایانه‌ای که برای انجام کاری نوشته شده است و کاربرانِ برنامه‌های کاربردی آن را برای انجام کارهای خود به کار می‌گیرند. بسیاری از برنامه‌های کاربردی امروزی نیاز به ذخیره سازی پایدار برخی از داده‌ها بر روی رسانه‌ی ذخیره سازی مانندِ دیسک دارند. برای نمونه برنامه‌ی کتابخانه برای ساده کردن کارهای ثبت نام عضوها و افزودن و ویرایش داده‌های کتاب‌ها و همچنین امانت نوشته می‌شود و به کار گرفته می‌شود. این برنامه داده‌های خود را باید به صورت پایدار بر روی دیسک ذخیره کند تا بتواند همواره آنها را به کار بگیرد و پس از خاموش شدن رایانه یا بسته شدن برنامه بتواند باز آنها را به کار گیرد. داشتن رابط کاربر پسند (User Friendly Interface)برای برنامه‌ها یکی دیگر از نیازهای بسیار مهم برنامه‌های کاربردی است تا نمایش اطلاعات و گرفتن اطلاعات از کاربر به روشی ساده، شایسته و کاربر پسند انجام شود. بنابراین برنامه نویسان همواره می‌کوشند رابطِ کاربر پسندتری را برای برنامه آماده نمایند.

کاربران نهایی یا کاربران برنامه(end users)

کاربران نهایی در یک نرم افزار کسانی هستند که برای برآورده شدنِ نیازهای خود برنامه یا برنامه‌های کاربردی را به کار می‌گیرند. در کتابخانه عضوها و کتابداران کاربران نهایی برنامه‌ی کتابخانه هستند. علاوه بر اعضاء رسمی یک نرم افزار که اطلاعات آنها برای نرم افزار قابل شناسایی است، این امکان هم در برنامه‌ی کاربردی فراهم می شود که دیگران نیز بتوانند بدونِ عضو شدن یا کتابدار شدن با برخی از بخش‌های برنامه‌ی کتابخانه کار کند و به برخی از داده‌ها را دسترسی داشته باشند. برای نمونه بتوانند فهرست کتاب‌ها را ببینند یا کتابی را جستجو کنند. به این دسته نیز کاربرانِ نهایی یا کاربران برنامه گفته می‌شود.

ناسازگاری داده‌ها(data inconsistency)

اگر داده‌های درون یک پرونده یا دفتر به گونه‌ای باشد که بخشی از داده‌ها معنایی نقیضِ معنای بخشِ دیگری از داده‌ها داشته باشند آن‌گاه گفته می‌شود که ناسازگاری داده پیش آمده است. برای نمونه یک کتاب را دو عضو نمی‌توانند به صورت همزمان امانت گرفته باشند و در واقع فقط یکی از آنها کتاب را امانت گرفته است و دیگری کتاب را پیش از آن پس داده است ولی برای امانتِ او مقدارِ ستون) ret کتاب برگردانده شده است یا خیر) تغییر نکرده است و همان n باقی مانده است و به این ترتیب دو سطر از جدول امانت مربوط به این کتاب وجود دارد که در آنها ret مقدارِ n دارد. اگر برای کتاب‌های دیگر یا کاربران بیشتری نیز این وضعیت وجود داشته باشد باز ناسازگاری داده خواهیم داشت. حالت‌های گوناگونی پیش می‌آید که ناسازگاری داده‌ها را در پی دارد. در حالتی که برنامه‌ی کاربردی به کار گرفته می‌شود برنامه یا بخش کار با داده در برنامه باید مراقب این وضعیت‌ها باشد و از پیش آمدنِ آنها جلوگیری کند. کار با پرونده‌ها به کمک برنامه نویسی برای نگهداری داده‌ها بر دردسرهای ناسازگاری داده‌ها می‌افزاید زیرا دستِ کم بررسی کردنِ شرط‌های سازگاری داده‌ها و پیشگیری از ناسازگاری داده‌ها در این حالت بسیار سخت است. همچنین افزودن، ویرایش یا حذف شرط‌های سازگاری در این حالت بسیار دشوار است.

افزونگی داده

اگر داده‌های یکسان (یا با معناهای یکسان) به شکل‌های گوناگون (یا یکسان) در جاهای گوناگونی تکرار شود و به هیچ دلیلِ منطقی نیازی به این تکرار نباشد آن‌گاه به این حالت افزونگی داده گفته می‌شود. برای نمونه اگر به جدول امانت ستونِ دیگری به نام موضوعِ موردِ علاقه‌ی عضوِ امانت گیرنده افزوده شود آن‌گاه افزونگی داده

داریم زیرا موضوع در جدول عضو در نظر گرفته شده است. افزونگی داده مشکلاتی همچون افزایش بی دلیل اندازه‌ی داده‌ها و همچنین احتمال پیش آمدن ناسازگاری را در پی دارد. برای نمونه اگر عضو بخواهد موضوع موردِ علاقه‌اش را عوض کند و اگر فقط این تغییر در جدول عضو انجام شود آن‌گاه میان موضوع مورد علاقه‌ی کاربر درونِ جدولِ عضو و موضوع‌های نوشته شده برای عضو در جدول امانت ناسازگاری پیش می‌آید. بنابراین باید از افزونگی داده پرهیز شود. دقت شود که ستون‌های شماره‌ی کتاب و شماره‌ی عضو در جدول امانت به هیچ عنوان به معنای افزونگی داده‌ها نیست بلکه بخشِ مهمی از داده‌های درونِ جدولِ امانت است که همزمان ارتباط دهنده‌ی میان سه جدول است.

کار با پرونده‌ها به کمک برنامه نویسی برای نگهداری داده‌ها احتمال پیش آمدنِ افزونگی داده را افزایش می‌دهد. اگر کتابخانه بخواهد بخشی نیز برای حسابداری داشته باشد و بدهکاری‌ها را حساب نماید یا حقوق کارمندان را پرداخت نماید آن‌گاه باید برنامه نویس این بخش حسابداری را نیز به برنامه بیفزاید یا اینکه برنامه‌ی جداگانه‌ای برای حسابداری بنویسد که پرونده‌های برنامه‌ی کتابخانه را نیز به کار گیرد. اگر بنا باشد که برنامه نویس ستون‌های دیگری را نیز به جدول‌های کنونی درونِ برنامه‌ی کتابخانه‌ی کنونی بیفزاید تا داده‌های حسابداری به خوبی نگهداری شوند آن‌گاه باید بسیاری از بخش‌های برنامه‌ی کتابخانه که وابسته به جدول‌های ویرایش شده هستند نیز بازنویسی شوند. این کار اگر نشدنی نباشد بسیار دشوار و هزینه‌بر است. بنابراین راه حل دیگر این است که برخی از داده‌های جدول‌های کنونی درونِ جدول‌های دیگری گذاشته شود تا نیاز نباشد جدول‌های اصلی ویرایش شوند. با این کار افزونگی داده پیش می‌آید. اگر داده‌ای در جدول‌های اصلی ویرایش شود باید داده‌های وابسته به آن در جدول‌های فرعی برای حسابداری نیز ویرایش شوند وگرنه ناسازگاری داده پیش می‌آید.

وابستگی داده‌ای (data dependency)و وابستگی ساختاری(structure dependency)

اگر بنا شود جریمه‌ی دیرکرد به ازای روز برای هر کتاب (fpd) به صورت اعشاری نگهداری شود آن‌گاه افزون بر ویرایش نوعِ fpd در ساختارِ کتاب (ص۸) بخش‌های گوناگونی از برنامه‌ی کاربردی باید بازنویسی شود. به کارگیری عدد اعشاری به جای عدد صحیح پس از مدتی کار کردن برنامه می‌تواند به دلیل نیازهای تازه‌ی کتابخانه

Comment [A1]:

باشد. بنابراین میان نوعِ داده‌های درونِ پرونده‌ها و برنامه‌ی کاربردی وابستگی شدیدی برقرار است به این وابستگی، وابستگی داده‌ای گفته می‌شود.

اگر بنا شود مترجم کتاب به ساختار کتاب افزوده شود آن‌گاه افزون بر افزودن یک فیلد دیگر به ساختار کتاب باید همه‌ی بخش‌هایی از برنامه که با این ویرایش دچار مشکل می‌شوند بازنویسی شوند. روشن است که بخش‌های زیادی از برنامه باید بازنویسی شوند. به این وابستگی شدید میان ساختارِ داده‌های درونِ پرونده‌ها و برنامه‌ی کاربردی وابستگی ساختاری گفته می‌شود. در برابرِ وابستگی داده‌ای و وابستگی ساختاری، استقلال داده‌ای[2] و استقلال ساختاری[3] تعریف می‌شوند که به ترتیب به معنای وابسته نبودن برنامه‌ی کاربردی به نوعِ داده و وابسته نبودن برنامه‌ی کاربردی به ساختارِ داده است. در اینجا و بیشتر بخش‌های این کتاب «داده» به معنای داده‌ی نگهداری شده در رسانه‌ی پایدار است و نه داده‌های موقتِ درونِ برنامه‌ها. اگر تغییر در نمایشِ فیزیکی (چگونگی ذخیره سازی در رسانه‌ی پایدار) و روش‌های دستیابی به داده‌ها بر روی برنامه‌های کاربردی اثر نگذارد، آن‌گاه استقلالِ داده‌ای فراهم شده است. اگر استقلال داده‌ای فراهم باشد نگهداری، پشتیبانی، ویرایش و گسترش برنامه‌های کاربردی بسیار ساده‌تر خواهد شد. استقلال داده ای بیشتر توضیح داده شود

ناهنجاری‌ها(anomaly)

اگر یک کتاب گم شود یا از میان برود یا به هر دلیلی بخواهند آن کتاب را کنار بگذارند و حذف کنند آن‌گاه امانت‌ها این کتاب نیز باید حذف شود و گرنه این امانت‌ها به کتابی اشاره می‌کنند که نیست. در این صورت یک ناهنجاری از نوعِ حذف پیش آمده است. بنابراین باید همه‌ی امانت‌های این کتاب حذف شود. همچنین اگر به هر دلیل بخواهند شماره‌ی یک کتاب را در جدول کتاب تغییر دهند آن‌گاه باید در تمامی امانت‌های این کتاب نیز شماره‌ی کتاب ویرایش شود و اگر این کار انجام نشود ناهنجاری از نوعِ ویرایش پیش آمده است. اگر عضوی کتابی به کتابخانه اهدا کرده باشد ولی هنوز در جدول کتاب اطلاعات کتاب مذکور وارد نشده باشد ولی شماره‌ی برای آن کتاب در جدولِ عضو گذاشته شده باشد آن‌گاه ناهنجاری از نوعِ افزودن پیش آمده است. ناهنجاری‌های حذف، افزودن(اضافه کردن)، ویرایش (اصلاح) یکی از مشکل‌هایی است که در کار با داده‌ها پیش می‌آید. اگر برای کار با داده‌ها برنامه نویسی مستقیمِ کار با پرونده انجام شود آن‌گاه انجام این دست تغییرها دشوار خواهد بود.

تحمل خرابی(fault tolerance)

اگر به هر دلیلی برنامه در هنگام ویرایش داده‌ها بسته شود و نتواند کار خود را به پایان برساند، آن‌گاه داده‌ها به طور کامل در رسانه ذخیره نمی‌شوند یا به طورِ کامل تغییر نمی‌یابند و فقط بخشی از داده ذخیره می‌شود یا تغییر می‌کند. در این حالت شاید ناسازگاری داده پیش آید. دلیل‌های بسته شدن برنامه می‌تواند؛ مشکل خود برنامه باشد یا سیستم عامل برنامه را در حالت اجرا به هر دلیلی می‌بندد یا برق قطع می‌شود یا به هر دلیل دیگری (برای نمونه سخت افزاری) رایانه خاموش می‌شود یا دوباره راه اندازی می‌شود. برای نمونه اگر در هنگام برگردانده شدن کتابی که دیرکرد داشته است برنامه بسته شود آن‌گاه ممکن است دیرکرد پیشین با دیرکرد کنونی جمع شده باشد و درونِ پرونده گذاشته شده باشد یا خیر. همچنین روشن نیست که عددی که پس از پیش آمدن خرابی درونِ این فیلد است آیا دیرکرد پیشین است یا جمع با دیرکرد کنونی یا هیچکدام (در هنگام نوشته نتیجه فقط بخشی از جمع دیرکرد نوشته شده باشد). به سختی می‌توان با روشِ برنامه نویسی با پرونده‌ها چنین چیزی را فهمید و برطرف کردن این مشکل با برنامه نویسی بسیار دشوار است. هنگامی که داده‌ها ارتباط‌ها پیچیده‌تری داشته باشند و یک تغییر نیازمند تعدادی تغییر در چندین جدول باشد آن‌گاه برطرف کردن این مشکل و تحمل خرابی دشوارتر می‌شود. برای نمونه اگر بخواهیم شماره‌ی عضوی را تغییر دهیم باید در همه‌ی امانت‌های آن عضو نیز شماره‌ی عضو را به روز کنیم و اگر در این میان برنامه بسته شود روشن نیست که چه امانت‌هایی وابسته به این عضو بوده است.

تراکنش(transaction)

در برابر مشکل‌های گفته شده در تحمل خرابی می‌خواهیم روشی داشته باشیم که به کمکِ آن یک تغییر به طور کامل انجام شود یا هیچ بخشی از آن انجام نشود در این حالت می‌توان دوباره تغییر خواسته شده را از نو انجام داد و انجام کار حالت نیمه کاره نخواهد داشت و کار به صورت تجزیه ناپذیر انجام نشود. مفهومِ تراکنش در سامانه‌های مدیریت پایگاه داده همین است. بودنِ توانایی تراکنش در این سامانه‌ها کارِ تحمل خرابی را به خوبی انجام می‌دهد.

قیدها

جامعیت داده‌ها(data integrity)

اگر به هر دلیلی تعدادِ روزِ دیرکرد برای یک امانت منفی گردد حساب کردن جریمه‌ی عضو به درستی انجام نخواهد شد. در واقع مقدار این فیلد با تعریفِ آن سازگار نیست. این یک مشکلِ جامعیتِ داده است. بنابراین باید شرطی وجود داشته باشد که همواره بررسی کند مقدارِ این فیلد نباید منفی شود و این یک قید جامعیت است. این قیدهای جامعیت اغلب در هنگامِ کار برنامه نیازشان احساس می‌شود؛ بنابراین بایدبه شکلی ساده‌تر بتوان این قیدها را بر روی داده‌ها گذاشت. این قیدها زمانی دشوارتر می‌شوند که چند فیلد از چند جدول را در بر بگیرند. برای نمونه «به عضوهایی که جریمه‌ی دیرکردِ آنها بیشتر از ۱۰۰۰ تومان است نباید کتابی امانت داده شود»، یک قید جامعیت است. برای برآورده شدنِ این قید جامعیت باید یک قانون جامعیت تعریف و به کار گرفته شود. دقت شود که قیدهایی که برای سازگاری داده‌ها و پیشگیری از ناسازگاری داده‌ها گذاشته می‌شوند نیز قیدهای جامعیت هستند. پیش آمدن هر گونه ناهنجاری نیز جامعیت داده‌ها را از میان می‌برد بنابراین قیدهای جامعیت می‌توانند برای پیشگیری از ناهنجاری‌ها نیز به کار برده شوند. تکراری نشدن شماره‌های کتاب‌ها در جدول کتاب نیز یک قید جامعیت است.

امنیت

تعریفِ کاربرانِ گوناگون با سطح دسترسی‌های گوناگون به داده‌ها یکی دیگر از نیازهای کار با داده‌ها است. روشن است که در برنامه نویسی مستقیم با پرونده‌ها افزودن کاربرانِ گوناگونِ داده‌ها و تعریفِ دسترسی‌های گوناگون برای آنان دشوارتر است. ویرایش سطح دسترسی کاربران در این حالت نیز دشوار است. به عنوان مثال در کتابخانه حداقل چهار دسته از کاربران وجوددارد؛ اول مدیر یا مدیران، دوم کارمندان کتابخانه، سوم عضوهای کتابخانه، چهارم بازدیدکنندگانِ کتابخانه که عضو نیستند. هر کدام از این چهار دسته باید توانایی‌ها و محدودیت‌های ویژه‌ی خود را داشته باشند.

اجرای همزمان

اغلب نیاز است که برنامه روی چندین رایانه نصب شود تا کاربران بتوانند به سادگی به کمکِ شبکه برنامه‌ی کتابخانه را اجرا نمایند ولی داده‌ها فقط در یکجا ذخیره شوند، زیرا تکرار داده‌ها روی چند رایانه دردسرهای گوناگونی را پیش می‌آورد؛ در این صورت درخواست‌های همزمانی به داده‌ها داده می‌شود بنابراین باید این حالت نیز پشتیبانی شود و بتوان چندین درخواستِ همزمان را به داده‌ها داد و دردسری نیز پیش نیاید. برای

نمونه اگر چند حذف یا ویرایش همزمان پیش بیاید باید به شکلِ درستی این درخواست‌ها انجام شود تا از روی هم افتادگی و ترکیبِ کارهایی که ترکیبِ آنها نتیجه‌ی نادرستی به بار می‌آورد؛ جلوگیری کند.

پشتیبان گیری(backup)

یکی دیگر از کارهایی که باید هر از چند گاهی انجام شود پشتیبان گیری از داده‌های ذخیره شده بر روی رسانه‌ای دیگر یا به گونه‌ای دیگر است. به این ترتیب اگر به هر دلیل رسانه‌ی ذخیره سازی یا داده‌ها دچار مشکل شد بتوان داده‌های پشتیبان (و بایگانی شده) را برگرداند. برنامه باید چنین توانایی را فراهم کند تا کاربران با سطح دسترسی بالا بتوانند کار پشتیبان گیری و برگرداندن پشتیبان را به سادگی انجام دهند. همچنین برنامه بهتر است بتواند به طور خودکار در بازه‌های زمانی تنظیم شود تا کار پشتیبان گیری به طورِ خودکار انجام شود. در کار مستقیم با پرونده‌ها برای کار با داده‌ها به کمکِ برنامه نویسی این کار چندان ساده نیست.

نمونه **پایگاه داده‌ی ساده**

کتابخانه‌ای را در نظر بگیرید که تعدادی کتاب و عضو دارد و عضوها می‌توانند کتاب امانت بگیرند. برای نگهداری اطلاعاتِ این کتابخانه دستِ کم سه دفتر(یا جدول) نیاز است: دفترِ کتاب‌ها، دفترِ عضوها و دفترِ امانت‌ها. درونِ دفتر کتاب‌ها درونِ هر سطر نام هر کتاب به همراهِ اطلاعات کامل کتابشناختی و جریمه‌ی دیرکرد آن کتاب به ازای هر روز دیر کرد به ریال وارد می‌شود. همچنین به هر کتاب شماره‌ی یکتایی داده می‌شود که معمولا شماره مدرک یا شماره ثبت بوده و در آغاز سطرِ آن کتاب درونِ دفترِ کتاب‌ها نوشته می‌شود(جدولِ۲). جریمه‌ی دیرکرد به ازای هر کتاب برای این است که بتوان برای کتاب‌های گوناگون با ارزش‌های گوناگون یا درخواست‌های کم یا زیادِ عضوها بتوان جریمه‌های گوناگون در نظر گرفت تا برای کتابی که عضوهای کمتری به آن علاقه‌مند هستند (کمتر آن را امانت می‌گیرند) جریمه‌ی کمتری در نظر گرفته شود و برای کتاب‌های که درخواست کننده‌ی بیشتری دارند یا اینکه بسیار گران هستند یا به دلایل های دیگر بتوان جریمه‌ی دیرکرد بیشتری در نظر گرفت.

جدول۲: کتاب(book)

جریمه‌ی دیرکرد به ازای روز به تومان(fpd)	موضوع کتاب (category)	نو اشر	نو یسنده	عنوان کتاب (bname)	شماره کتاب(bn)
10	هنر	وزگار نو	جا ن راسکین	در باب هنر و زندگی	b1

باید توجه داشت اگرچه امروزه می‌توان نام‌های فارسی را برای نام ستون‌ها برگزید ولی برای سادگی در این کتاب نام اغلب کوتاه شده‌ای برای ستون‌ها برگزیده شده است تا در بخش‌هایی که بناست دستورهای SQL نوشته شود بسادگی بتوانید آن دستورها را اجرا نمایید.

جدول ۳: عضو(member)

شماره کتاب اهدایی(bn)	موضوع مورد علاقه (category)	نام عضو (Fmname)	نام خانوادگی عضو (Lmname)	شماره عضو(mn)
b1	هنر		پارسایی	m1

نام هر عضو و موضوعِ مورد علاقه‌ی او در هنگام ثبت نامِ عضو درونِ دفترِ عضو نوشته می‌شود و عضو می‌تواند کتابی به کتابخانه اهدا نماید و شماره‌ی در نظر گرفته شده برای کتابِ اهدایی او در کنار نام و موضوعِ موردِ علاقه‌ی او نگه داشته می‌شود. به هر عضو نیز شماره‌ی یکتایی داده می‌شود. اگر عضو کتابی اهدا ننماید به جای شماره‌ی کتاب تهی (پوچ یا هیچ (null گذاشته می‌شود تا روشن شود این عضو کتابی اهدا نکرده است (جدول ۳). می‌توان به جای به کارگیری null ، صفر (شماره‌ای که نمی‌تواند شماره‌ی کتابی باشد) درون آن گذاشت. در دفترهای معمولی(کاغذی) اغلب هیچ چیزی گذاشته نمی‌شود و آن بخش خالی گذاشته می‌شود ولی در حالتِ رایانه‌ای (به کارگیری پرونده) باید همواره درون جدول چیزی گذاشته شود و بنابراین برای هماهنگی با دیگر بخش‌های کتاب همان null برای این بخش‌ها به کار گرفته می‌شود که در اینجا به معنای نبودن مقداری برای این بخش از داده‌ها است.

جدول ۴: امانت(borrow)

برگردانده شده یا خیر(ret)	مجموع روز دیرکرد(fd)	تاریخ امانت(bdate)	شماره کتاب(bn)	شماره عضو(mn)
n	5	13950812	b1	m1

هنگامِ امانت کتاب شماره‌ی کتاب به همراه شماره‌ی عضو امانت گیرنده و تاریخ امانت نوشته می‌شود. تاریخ امانت همان روزی است که کتاب امانت گرفته می‌شود و به صورت ۱۳۹۵۰۸۱۲ (دوازدهم آبان ماه ۱۳۹۵) نوشته می‌شود و همواره هشت نویسه (کاراکترcharacter) عددی دارد که تاریخ را نشان می‌دهد(۰۱ تا ۰۹ به جای ۱ تا

۹ نوشته می‌شود). روبروی امانت علامتی مانند n گذاشته می‌شود که نشان می‌دهد این کتاب هنوز برگردانده نشده است. در هنگامِ برگرداندنِ کتاب اگر کتاب دیر برگردانده شده باشد (دیرکرد داشته باشد) تعدادِ روزِ دیرکردِ کتاب روبروی آن امانت در ستون مجموع روز دیرکرد نوشته می‌شود و گرنه صفر روبروی آن نوشته می‌شود. همچنین علامتی مانند y روبروی امانت به جای n گذاشته می‌شود که نشان دهد کتاب برگردانده شده است(n پاک می‌شود و y به جای آن نوشته می‌شود). بارِ دیگری که این عضو بخواهد همین کتاب را امانت بگیرد آن‌گاه به جای اینکه سطر جدیدی به دفتر امانت افزوده شود؛ nجایگزینِ yدر همین سطر می‌شود و تاریخ امانت کنونی جایگزین تاریخِ امانت پیشین می‌شود. هنگامی که عضو جریمه‌ی دیرکرد را پرداخت کرد تعداد روز دیرکرد صفر گذاشته می‌شود. هنگامی که عضو این کتاب را برگرداند yبه جای nنوشته می‌شود. همچنین اگر دیرکردی داشته باشد تعدادِ روزِ دیرکرد با تعدادِ دیرکرد از پیش جمع می‌شود و درون ستونِ دیرکرد گذاشته می‌شود. در جدول ۴ چهار کتاب به امانت گرفته شده است.

روشن است که این تعداد دفتر (جدول) نمی‌تواند به طور کامل برای یک کتابخانه‌ی واقعی به کار گرفته شود ولی همین شکل ساده به خوبی مهم‌ترین داده‌هایی را نگهداری می‌کند که برای یک کتابخانه‌ی واقعی نیاز است. دردسرهای کارکردن با این دفترها روشن است و نیاز به جستجو، پاک کردن، جایگزینی، نوشتن، گرفتن کپی و همانند آن در یک کتابخانه واقعی که این دفترها را به کار بگیرد به خوبی دیده می‌شود. با افزایش تعداد کتاب‌ها و عضوها و امانت‌ها کار کردن با این دفترها سخت‌تر و زمان‌بر خواهد شد. بنابراین اگر بتوان از رایانه برای نگهداری این دفترها و کار کردن بر روی آنها کمک گرفت کار کردن با آنها ساده‌تر می‌شود. به این دفترها در جاهایی پرونده گفته می‌شود ولی در این کتاب منظور از پرونده فقط پرونده‌ی ذخیره شده در رایانه فایل (file) است.

Comment [A2]: تعریف صفات

همان گونه که عنوان شد هر دفتر از چند سطر و چند ستون ساخته می‌شود. اغلب سطرها هستند که ویرایش می‌شوند و کم یا زیاد می‌شوند. در دنیای واقعی ستون‌های دفترها به سادگی تغییر نمی‌یابند زیرا نیاز است دفتر دوباره از نو ساخته شود و ستون‌ها تغییر یابند و همه‌ی داده‌ها در دفتر جدید بازنویسی شوند و این کار دشواری است. بنابراین در آغاز باید در گزینش تعداد و نوع ستون‌های هر دفتر دقت کرد زیرا ستون‌های دفتر بخشی از ساختار دفتر یا به طور کلی ساختارِ طرح کلی شِما، (schema) داده‌ها را مشخص می‌کنند. پس تعریف طرح کلی داده‌ها (در اینجا تعداد و نام دفترها و تعداد و نام ستون‌های هر دفتر) ارزشِ زیادی دارند و باید دقت

شود تا طرح کلی شایسته‌ای برای داده‌ها آماده شود و نیازهای آینده را نیز تا اندازه‌ای پیش بینی کند. مدل رابطه‌ای (relational model) که امروز در بیشتر کاربردهای پایگاه داده‌های تجاری به کار گرفته می‌شود بسیار به همین شکلِ سنتی دفتری و جدولی نزدیک است.

مدل رابطه ای

Comment [A3]: به طور کامل به آن پرداخته شود

کمی دقیق‌تر درباره‌ی مدل رابطه‌ای

مدل رابطه‌ای بسیار به مفهوم جدول نزدیک است ولی تفاوت‌های عمده‌ای نیز با آن دارد که کوشش می‌کنیم به طور فشرده در زیر فهرست کنیم بدون این که بیشتر از این به جزییات تفاوت‌ها بپردازیم.

رابطه

رابطه مفهومی کاملا ریاضی است و به طور دقیق‌تر هر رابطه یک مجموعه از چندتایی‌ها هست. در زیر نمونه‌ای از یک رابطه نشان داده شده است.

اعضای کتابخانه =

{

(«شماره‌ی عضویت»: ۲ ، «موضوع مورد »:«نام»:«کوروش» ، «نام خانوادگی» کاظمی» ، «علاقه»:«هنر») ،

(«شماره‌ی عضویت»: ۳ ، «موضوع مورد » نام»:«زیبا» ، «نام خانوادگی»:«سروش» ، «علاقه»:«ریاضی») ،

(«شماره‌ی عضویت»: ۳ ، «موضوع مورد » نام»:«علی» ، «نام خانوادگی»:«انتشاری» ، «علاقه»:«ریاضی») ،

(«شماره‌ی عضویت»: ۶ ، «موضوع » نام»:«کامران» ، «نام خانوادگی»:«خداپرستی» ، «مورد علاقه»:«داستان») ،

(«شماره‌ی عضویت»: ۷ ، «موضوع » نام»:«کیانوش» ، «نام خانوادگی»:«محمدی» ، «مورد علاقه»:«شیمی»)

}

این شکل اصلی مدل رابطه‌ای در واقع به شکل کنونی پایگاه داده‌های غیر رابطه‌ای که json را به کار می‌برند تا اندازه‌ای نزدیک است و البته تفاوت‌های مهمی میانشان وجود دارد که خارج از موضوع این کتاب است.

(«نام»:«کوروش» ، «نام خانوادگی»:«کاظمی» ، «شماره‌ی عضویت»: ۲ ، «موضوع مورد علاقه»:«هنر»)

این یک چندتایی(تاپل) است که در جدول ستون نامیده می‌شود. «نام» یک ویژگی از رابطه‌ی «اعضای کتابخانه» است که مقدارش در این چندتایی «کوروش» است. در حالت جدولی «نام» یک ستون از جدول است. هر رابطه در ویژگی خودش دارای دامنه‌ی ویژه‌آی است که به شکل دقیق‌تر یک مجموعه است. برای نمونه در اینجا «نام» مجموعه‌ای از نام‌های کوچک ممکن افراد است. در حالت جدولی نام ستون خود باید متعلق به دامنه‌ای(نوعی) باشد.

با توجه به رابطه‌ای که نشان داده شد می‌توان گزاره‌های زیر را به دست آورد.

- ترتیب چندتایی‌ها اهمیتی ندارد.
- ترتیب ویژگی‌ها اهمیتی ندارد و می‌توان به صورت («شماره‌ی عضویت»: ۲ ، «موضوع مورد علاقه»:«هنر» ، «نام»:«کوروش» ، «نام خانوادگی»:«کاظمی») نیز آن را نوشت.
- هیچ چندتایی تکراری نمی‌تواند در یک رابطه وجود داشته باشد زیرا که رابطه یک مجموعه است و مجموعه عضو تکراری ندارد.

جدول

برخلاف رابطه در جدول

- ترتیب رکوردها اهمیت دارد.
- ترتیب ستون‌ها اهمیت دارد.
- رکوردهای تکراری می‌تواند در جدول وجود داشته باشد.

نگاه این کتاب به مدل رابطه‌ای

در این کتاب مفهوم‌ها رابطه و جدول را به جای هم به کار می‌بریم و در واقع گرچه جدول را می‌کشیم و زبان اِس کیو اِل (SQL) نیز جدول را به کار می‌برد ولی برای فهم بهتر مدل رابطه‌ای و بهتر نوشتن دستورهای SQLمی‌کوشیم قاعده‌های اصلی مدل رابطه‌ای را در نظر داشته باشیم و خود را محدود به جدول نکنیم.

هنجارسازی یا نرمال سازی داده(normalization)

نرمال سازی پایگاه داده روندی برای ساماندهی ویژگی‌ها (ستون‌ها) و رابطه‌ها (جدول‌ها) در یک پایگاه داده‌ی رابطه‌ای به منظور کاهش افزونگی داده است.

هدف از نرمال سازی؟

فرض کنید که قرار است پایگاه اطلاعاتی برای ذخیره و بازیابی مقالات یک حوزه ایجاد گردد. برای ایجاد چنین پایگاهی بایستی ابتدا تمامی اطلاعات کتابشناختی و غیر کتابشناختی مقاله ها با عنوان فیلد یا ویژگی موجودیت مقاله شناسایی شود. این فیلدها عبارتند از:

- عنوان مقاله
- نویسنده / نویسندگان
- عنوان نشریه
- جلد نشریه
- شماره نشریه
- سال انتشار
- شماره صفحه
- آدرس الکترونیکی
- فهرست منابع

نرمال سطح ۱

هنجار نوع یکم

در هر خانه از جدول فقط یک مقدار گذاشته می‌شود و چند مقدار با هم در یک خانه از جدول قرار نمی‌گیرد. گرچه مفهوم غیر قابل تجزیه بودن یک خانه از جدول کمی پیچیده است و امروز نیز کمی متفاوت از گذشته است با این همه اکنون برای سادگی می‌توانیم فرض کنیم که هر ستون فقط از یک نوع ساده مانند عدد صحیح یا رشته یا همانند آن است و بنابراین در هر خانه از جدول فقط یک عدد یا رشته یا تاریخ یا همانند آن بسته به نوع ستون ذخیره می‌شود و نمی‌توان دو عدد را در یک خانه از جدول گذاشت و آنها را با کاما (،) یا همانند آن جدا کرد و فقط یک عدد صحیح در هر خانه از یک ستون با نوع عدد صحیح می‌تواند گذاشته شود. بنابراین در مثال مربوط به مقاله بعضی از اطلاعات مربوط به یک مقاله را نمی توان در جدول اصلی ذکر کرد، بلکه باید یک جدول دومی را ایجاد نمود که با استفاده از رابطه بین داده ها ارتباط برقرار نمود.

تبدیل شده به نرمال ۱

روشن است که این جدول مشکل دارد و روشن این که باید دو جدول باشد ولی چه قانونی این را می‌گوید. بنابراین همانطور که قبلا توضیح داده شد، باید فلیدها یا ویژگیهای مرکب و ساده را از همدیگر تفکیک کرد. در مثال فوق ویژگیهای مرکب عبارتند از:

- نویسنده: در بعضی از موارد یک مقاله ممکن است بیش از یک نویسنده داشته باشد، بنابراین امکان ورود اسامی نویسندگان در یک خانه از جدول وجود ندارد. در چنین شرایطی برای فیلد های مرکب، جدول جداگانه ای ایجاد می شود و اطلاعات تمامی نویسندگان در آن ذخیره می شود (جدول ۵).

شماره نویسنده	نام نویسنده	نام خانوادگی نویسنده	شماره مقاله	عنوان مقاله
۱	محمد	توکلی زاده	۱	**استناد به**

		راوری		انتشارات ملی
۲	لیلا	حسین میرزایی	۲	داده کاوی و کاربرد آن در تصمیم گیری
۳	فرامرز	سهیلی	۳	استفاده از تکنیک داده کاوی
۴	طاها	منصوری	۳	استفاده از تکنیک داده کاوی

- فهرست منابع: هر مقاله معمولا دارای بیش از یک فهرست منابع است، بنابراین با توجه به تکرار شدن این ویژگی در هر مقاله، همچون نویسنده مقاله، امکان ورود در جدول اصلی وجود ندارد.

ویژگیهای ساده عبارتند از:

فیلد هایی از مقاله که ماهیت تکرارپذیری ندارند، می توانند در جدول اصلی مقاله اطلاعات آنها ذکر شود. این فیلدها در موجودیت مقاله عبارتند از:

- عنوان مقاله
- عنوان نشریه
- جلد نشریه
- شماره نشریه
- سال انتشار
- شماره صفحه
- آدرس الکترونیکی
- DOI

دو جدول پیشنهادی

بنابراین با توجه به توضیحات ذکر شده در خصوص ویژگیهای مرکب و ساده دو جدول جداگانه برای هر گروه از ویژگیها پیشنهاد می شود.

جدول ویژه فیلدهای ساده: این جدول (۶) در این نوشتار به عنوان جدول اصلی در مثالها یاد می شود.

جدول۶: ویژگی های ساده مقاله

شماره صفحه پایان مقاله	شماره صفحه آغاز مقاله	سال انتشار مقاله	شماره نشریه	جلد نشریه	عنوان نشریه	عنوان مقاله	شماره مقاله
۳۵۶	۳۳۷	۱۳۹۳	۲	۳	پردازش و مدیریت اطلاعات	استنناد به انتشارات ملی	۱
۱۴	۳	۱۳۸۶	۴	۷۹	دانش مدیریت	داده کاوی و کاربرد آن در تصمیم گیری	۲
۴۶	۵۵	۱۳۹۲	۵۴	۱۶	مدیریت سلامت	استفاده از تکنیک داده کاوی	۲

جدول ویژه فیلدهای مرکب: در این مثال فقط ویژگی نویسنده به عنوان یکی از ویژگیهای مرکب در جدول جداگانه ذکر شده است.

جدول ۷: ویژگی های نویسندگان به عنوان یکی از ویژگیهای مرکب مقالات

نام خانوادگی	نام	شماره نویسنده
توکلی زاده راوری	محمد	۱
حسین میرزایی	لیلا	۲

سهیلی	فرامرز	۳
منصوری	طاها	۴
ناصرزاده	محمدرضا	۵
فراست	علیرضا	۶
تقوی فرد	محمدنقی	۷
علیزاده	سمیه	۸
ابوالمعصوم	فرانک	۹
اصغری	محسن	۱۰

مشکل‌های جدول‌های پیشین چه بودند؟

ناهنجاری افزودن داده، حذف داده، به روز رسانی داده

وابستگی تابعی

در جدول اصلی، شماره‌ی مقاله تعیین کننده‌ی (determinant) ویژگی های دیگر مقاله یعنی «عنوان مقاله» و «عنوان نشریه»، «جلد نشریه»، «شماره نشریه»، «صفحه آغازین»و «صفحه پایانی»، است. از طرف دیگر می‌توان گفت این ویژگی ها وابسته[1] به شماره‌ی مقاله هستند.

به این نوع وابستگی، «وابستگی تابعی (Functional Deppendancy)»یا FD گفته می‌شود. وابستگی تابعی میان ویژگی‌های جدول برخاسته از تعریف و نوع ارتباط آنها با هم دیگر در جدول است و طراح پایگاه داده باید بتواند وابستگی‌های تابعی میان ویژگی‌های درون جدول‌های خود را بیابد.

[1] dependant

بنابراین در هر جدول یک یا چند وابستگی تابعی وجود دارد. برای نمونه تعدادی از وابستگی‌های تابعی دو جدول بالا به قرار زیر است

- در جدول اصلی (مقاله)
  - شماره مقاله نام مقاله
  - شماره مقاله نام نشریه
  - یا به صورت دیگر : شماره مقاله «نام مقاله ، نام نشریه»
- در جدول نویسندگان
  - «شماره نویسنده » «نام نویسنده»

همان گونه که روشن است وابستگی‌ها به کلید هستند ولی وابستگی‌های دیگری نیز می‌تواند وجود داشته باشد.

ویژگی‌های ریاضی وابستگی تابعی

به این ترتیب وابستگی تابعی می‌تواند به صورت یک عمل ریاضی تعریف شود و ویژگی‌های آن از دید ریاضی بر روی رابطه‌ها بررسی گردد. بنابراین تعدادی اصل و تعدادی ویژگی و قضیه ار روی آن به دست می‌آید که در اینجا به بخشی از آن پرداخته می‌شود.

اگر X ، Yو Z مجموعه‌هایی از ویژگی‌ها در یک رابطه‌ی R باشند برخی از قانون‌هایی که نتیجه می‌شود در ادامه نوشته شده است.

اصول آرمسترانگ

- بازتابی بودن: اگر Y زیر مجموعه‌ی X باشد آن‌گاه X Y
- بسط پذیری : اگر Y X آن‌گاه XZ YZ
- تراگذری : اگر Y X و Z Y آن‌گاه X Z

قانون‌های زیر می‌تواند از روی سه قانون بالا به دست آید:

- خود تعیینی : همواره داریمX X
- اجتماع : اگر Y X و Z X آن‌گاهYZ X
- تجزیه : اگر YZ X آن‌گاه Y X وZ X
- شبه تراگذری: اگر Y X و Z WY آن‌گاهZ WX
- ترکیب : اگر Y X و W Z آن‌گاهYW XZ
- اتحاد کلی : اگر Y X و W Z آن‌گاهYW (Z - Y) X

بستار وابستگی تابعی

به مجموعه‌ی همه‌ی وابستگی‌های تابعی که می‌تواند از روی چند وابستگی تابعی به دست آید بستار آن وابستگی‌های تابعی گفته می‌شود. اگر F یک مجموعه وابستگی باشد F^+مجموعه‌ی بستار F است.

برای نمونه اگر در رابطه‌ی R ویژگی‌های A,B,C,D را به همراه قانون‌های وابستگی زیر داشته باشیم

۱. $A \rightarrow B$
۲. $B \rightarrow C$
۳. $AB \rightarrow D$

آن‌گاه قانون‌های زیر (بخشی از بستار این قانون‌ها) می‌تواند به دست آید:

a. $A \rightarrow A$
b. $A \rightarrow AB$
c. $A \rightarrow ABD$
d. $A \rightarrow ABCD$
e. $B \rightarrow B$
f. $B \rightarrow BC$
g. $A \rightarrow D$

مجموعه‌ی کاهش ناپذیر وابستگی‌ها

در یک رابطه (جدول) به مجموعه‌ای از وابستگی‌های تابعی گفته می‌شود که کمینه باشد یا به عبارت دیگر هیچ کدام از قانون‌های وابستگی درون این مجموعه نتواند به شکلی از روی دیگر قانون‌ها به دست آید. مجموعه‌ای کاهش ناپذیر از وابستگی‌ها است اگر و فقط اگر سه قانون زیر را داشته باشد.

- سمت راست(وابسته) هر وابستگی تابعی فقط یک ویژگی داشته باشد.
- سمت چپ (تعیین کننده) هر وابستگی تابعی کاهش ناپذیر باشد. یعنی اگر سمت چپ یکی از قانون‌های وابستگی را کاهش دادیم حتما مجموعه‌ی بستار به دست آمده از آن تغییر کند(نه اینکه بتواند از قانون‌های دیگر به دست آید). به عبارت دیگر همه‌ی وابستگی‌ها باید کاهش ناپذیر چپ باشند.
- هیچ وابستگی تابعی نتواند بدون تغییر بستار از این مجموعه حذف شود.

نرمال سطح دوم

اگر ویژگی‌هایی به بخشی از کلید اصلی وابسته باشند به آن *وابستگی جزئی* گفته می‌شود. جدولی در سطح دوم نرمال است که درون آن وابستگی جزئی وجود نداشته باشد. یعنی هیچ قانون وابستگی که سمت چپ آن (تعیین کننده) آن بخشی از کلید باشد در وابستگی‌های آن جدول وجود نداشته باشد. به عبارت دیگر بخشی از کلید در سمت چپ هیچ وابستگی تابعی نباشد.

اکنون به جدول اولیه‌ی مقاله دقت کنید. در این جدول کلید ویژگی شماره مقاله و شماره نویسنده در حالی که وابستگی تابعی شماره نویسنده □ {نام نویسنده و بقیه ویژگی ها} را داریم که در سمت چپ آن (تعیین کننده آن) فقط بخشی از کلید اصلی است. ویژگی یا ویژگی‌هایی از جدول به بخشی از کلید اصلی وابسته هستند و بنابراین باید جدا شوند.

برای به دست آوردن فرم نرمال دوم باید همه‌ی وابستگی‌های جزئی را به دست آوریم و از آنها جدول‌های جداگانه ایجاد کنیم. دقت کنید که وابستگی‌های جزئی که در آنها سمت چپ (تعیین کننده) یکسان است باید با هم ترکیب شوند.

نرمالBCNF

۱ .هر دانشجو ممکن است در چندین رشته تحصیل کند.

۲ .برای هر رشته، یک دانشجو فقط یک استاد راهنما دارد

۳ .در هر رشته چندین استاد راهنما وجود دارد

۴ .هر استاد راهنما فقط در یک رشته راهنمایی می‌کند.

استاد راهنما رشته

}شماره دانشجویی و رشته { استاد راهنما

جدول راهنمایی

استاد راهنما	رشته	شماره دانشجویی
یوسفان	کامپیوتر	123
عرب بیگی	هنر	243
عباسیان	مکانیک	342

یک مشکل مهم این جدول این است که در آن وابستگی مهم میان استاد راهنما و رشته نشان داده نشده است. روشن است که ساختار جدول‌ها بهتر است به گونه‌ای باشد که تا جایی که بشود بتواند شرایط واقعی را نشان دهد و وابستگی تابعی را نشان دهد.

شماره دانشجویی و استاد راهنما

جدول شماره دانشجویی - استاد راهنما

استاد راهنما	شماره دانشجویی
یوسفان	123
عرب بیگی	243
عباسیان	342

رشته و استاد راهنما

جدول رشته - استاد راهنما

رشته	استاد راهنما
یوسفان	کامپیوتر
عرب بیگی	هنر
عباسیان	مکانیک

همه‌ی وابستگی‌ها در جدول فقط به کلید اصلی باشد.

نرمال سطح چهارم

فرض کنید در فروشگاهی امکان خرید گروهی نیز باشد یعنی یک گروه از خریداران با نام‌های گوناگون یک خرید را با هم انجام دهند و یک فاکتور برایشان صادر شود. برای نمونه اگر سه نفر از اعضا خانواده با هم برای خرید آمدند نام هر سه نفر در فاکتور خرید آورده شود در حالی که یک فاکتور برای جنس‌های درون آن زده شود. بنابراین قانون زیر نیز افزوده شود:

برای یک فاکتور یک یا چند قلم جنس و یک یا چند خریدار می‌تواند وجود داشته باشد.

با این فرض، جدول زیر برای فروش پیشنهاد شده است.

جدول فروش همراه خریدار

نام خریدار	میزان فروش	شماره فاکتور فروش	شماره جنس
Ali	12	1	1
Kamran	14	4	1
Hamid	5	5	1
Koroush	10	2	2
Kamran	20	4	2
Hamid	30	5	2
Reza	40	3	2
Shahin	5	5	1
Shahin	30	5	2

میزان فروش فقط به شماره‌ی جنس و شماره‌ی فاکتو بستگی دارد بنابراین جدول در نرمال سطح دوم نیست پس به دو جدول زیر شکسته می‌شود.

جدول فروش

میزان فروش	شماره فاکتور فروش	شماره جنس

جدول فروش

میزان فروش	شماره فاکتور فروش	شماره جنس
12	1	1
14	4	1
5	5	1
10	2	2
20	4	2
30	5	2
40	3	2

جدول فروش همراه خریدار

نام خریدار	شماره فاکتور فروش	شماره جنس
Ali	1	1
Kamran	4	1
Hamid	5	1
Koroush	2	2
Kamran	4	2
Hamid	5	2
Reza	3	2

جدول فروش همراه خریدار

نام خریدار	شماره فاکتور فروش	شماره جنس
Shahin	5	1
Shahin	5	2

اکنون اگر این جدول را بدون توجه به دیگر جدول‌های نگاه کنیم روشن است که جدول خوبی نیست ولی تا نرمال BCNF مشکلی ندارد. روشن است که تکرار داده دارد در واقع نوعی گروه بندی یا عدم وابستگی میان آنها وجود دارد که باعث می‌شود بهتر ببینم به دو جدول زیر آن را بشکنیم بدون این که اطلاعاتی از دست برود.

جدول فروش: فاکتور - جنس

شماره فاکتور فروش	شماره جنس
1	1
4	1
5	1
2	2
4	2
5	2
3	2

جدول فروش: فاکتور - خریدار

نام خریدار	شماره فاکتور فروش
Ali	1
Kamran	4
Hamid	5
Koroush	2
Kamran	4
Hamid	5
Reza	3

مشخص است که با توجه به جدول فروش که از سه ستون شماره جنس، شماره فاکتور فروش و میزان فروش ساخته شده است دیگر نیازی به جدول فروش: فاکتور - جنس نیست و می‌تواند حذف شود و دو جدول زیر باقی بماند.

جدول فروش

میزان فروش	شماره فاکتور فروش	شماره جنس
12	1	1
14	4	1
5	5	1
10	2	2
20	4	2

جدول فروش

میزان فروش	شماره فاکتور فروش	شماره جنس
30	5	2
40	3	2

جدول فروش: فاکتور - خریدار

نام خریدار	شماره فاکتور فروش
Ali	1
Kamran	4
Hamid	5
Koroush	2
Kamran	4
Hamid	5
Reza	3

نمونه نرمال سازی

جدول فروش زیر را در نظر بگیرید:

جدول فروش همراه خریدار

دیگر اطلاعات خریدار	نشانی خریدار	نام خریدار	میزان فروش	شماره فاکتور فروش	شماره جنس
other	Street 1	Ali	12	1	1
other	street 2	Kamran	14	4	1
other	street 3	Hamid	5	5	1
other	street 1	Koroush	10	2	2
other	street 2	Kamran	20	4	2
other	street 3	Hamid	30	5	2
other	street 2	Reza	40	3	2

باز هم این جدول دلچسب نیست و مشکل دارد. اولین مشکل این است که نرمال دوم نیست زیرا نام خریدار فقط به شماره فاکتور وابسته است بنابراین دو جدول زیر به دست می‌آید.

جدول فروش یک فروشگاه

میزان فروش	شماره فاکتور فروش	شماره جنس
12	1	1

جدول فروش یک فروشگاه

میزان فروش	شماره فاکتور فروش	شماره جنس
14	4	1
5	5	1
10	2	2
20	4	2
30	5	2
40	3	2

جدول فاکتور - خریدار

دیگر اطلاعات خریدار	نشانی خریدار	نام خریدار	شماره فاکتور فروش
other	Street1	Ali	1
other	street2	Kamran	4
other	street3	Hamid	5
other	street1	Korush	2
other	street2	Kamran	4

جدول فاکتور - خریدار

دیگر اطلاعات خریدار	نشانی خریدار	نام خریدار	شماره فاکتور فروش
other	street3	Hamid	5
other	street2	Reza	3

باز هم این جدول مشکل دارد و به خوبی می‌توان این مشکلات را به ویژه زمانی که تعداد خریداران و فاکتورها افزایش یابد دید زیرا نشانی خریدار و دیگر اطلاعات خریدار بی‌جهت تکرار می‌شود. بنابراین جدول فاکتور-خریدار به دو جدول زیر شکسته می‌شود.

جدول فاکتور - خریدار

نام خریدار	شماره فاکتور فروش
Ali	1
Kamran	4
Hamid	5
Koroush	2
Kamran	4
Hamid	5
Reza	3

جدول خریدار

نام خریدار	نشانی خریدار	دیگر اطلاعات خریدار
Ali	Street1	other
Kamran	street2	other
Hamid	street3	otherخ
Koroush	street1	other
Kamran	street2	other
Hamid	street3	other
Reza	street2	other

حفظ وابستگی‌ها و کنار هم گذاشتن ویژگی‌هایی که به هم وابسته هستند.

مقدمه

معرفی و تاریخچه پایگاه اطلاعاتی (داده)

تعریف پایگاه اطلاعاتی (داده ها)

تعریف داده

تعریف اطلاعات

نوع واحد یا موجودیت

- تعریف سیستم مدیریت پایگاه داده
- روش سنتی یا مشی فایلینگ

نظام فایلی

در نظام فایلی معمولا داده ها به صورت مجزار و در فایلهای جداگانه و بعضا تحت مدیریت برنامه های کاربردی متفاوت نگهداری و در دسترس قرار می گیرند. آنچه که در این نوع از پایگاهها د راختیار استفاده کننده نهایی قرار می گیرد، به صورت مجزا و در قالب فایلهایی است که ارتباطی بین این فایلها وجود ندارد. بر این اساس ممکن است برای هر کدام از فایلها برنامه های کاربردی خاصی تهیه شود، بنابراین برنامه های کاربردی برای مدیریت و دسترس پذیری فایلها طراحی شده است. همانطور که در این تصویر مشخص است فایلها به صورت مجزا تهیه شده و ارتباطی نیز بین آنها برقرا رنیست.

به عنوان مثال قرار است پایگاهی برای گردآوری داده های استنادی طراحی شود. به طور معمول لازم است که داده های استنادی منابع از قبیل کتاب، مقاله، پایان نامه و غیره گردآوری شود. در این پایگاه برای هر کدام از قالبهای اطلاعاتی فایلهای مستقلی برای ذخیره و بازیابی اطلاعات تعریف شده است که هر کدام از فایلها با برنامه کاربردی مجزا مدیریت می شود.

در این نوع پایگاه، در صورت نیاز یکی از استفاده کنندگان به اطلاعات استنادی کتاب و همچنین اطلاعات استنادی مقالات یک نویسنده مشخص، بایستی از نرم افزارهای جداگانه ای برای دسترسی به

این اطلاعات استفاده کرد، به عبارت دیگر برای دسترسی به اطلاعات مرتبط با هم این امکان وجود ندارد که به صورت یکپارچه جستجو را انجام داد، بلکه باید به صورت مستقل از هر کدام از فایل منابع اقدام به جستجو نمود.

معایت روش فایلی

با توجه به عدم تمرکز داده ها در یک فضا و استفاده از برنامه های کاربردی مستقل برای هر جزء یک پایگاه، مسائل و مشکلات زیر به وجود می آید.

۱- افزونگی اطلاعات

افزونگی اطلاعات به معنی ذخیره بیش از یک بار اطلاعات یکسان در یک پایگاه اطلاعاتی است.این مفهوم را با یک مثال تشریح می کنیم: فرض نمایید که در پایگاه استنادی نویسندگان مقالات با اطلاعات کامل در فایل مقالات ذخیره شده است و همزمان نیز اطلاعات همان نویسنده را در فایل کتاب به عنوان نویسنده کتاب ذخیره شود، در این شرایط اطلاعات یکسان در یک پایگاه و در فایل مجزا تکرار شده است و در صورت تغییر اطلاعات نویسنده بایستی هر دو فایل را تصحیح نمود.

۲- وابستگی داده در مقابل استقلال داده

وابستگی داده به این معنی است که برنامه کاربردی وابسته به داده است. هر تغییری که در داده می شود بایستی برنامه نیز تغییر پیدا کند. استقلال داده در مقابل وابستگی داده قرار دارد و بدین معنا است که هر تغییری که در داده ها داده می شود بدون نیاز به تغییر در برنامه اتفاق می افتد. وابستگی داده در نظام های فایلی وجود دارد.

۳- ناسازگاری داده

در سیستم فایلی ممکن است هر برنامه ای به یک زبان خاص نوشه شده باشد، بنابراین یک نوع ناسازگاری در زمان استفاده و تبدیل و انتقال داده ممکن است ایجاد شود.

۴- جدایی و پراکندگی داده

همانطور که عنوان شد در سیستم فایلی داده ها در فایلهای جداگانه نگهداری می شوند، همین امر دسترسی به اطلاعات را مشکل و بعضی مواقع دچار خطا می سازد. به همین دلیل ممکن است نیاز باشد برنامه سومی نوشته شود تا صحت دو داده قبلی را کنترل کرده و دو فایل مستقل از هم را با همدیگر تطبیق دهد.

مشی پایگاهی

مسائل و مشکلات موجود در مشی فایلی باعث رویکردی شد که مسائل و مشکلات مشی فایلی را حل نماید. بنابراین رویکرد مدیریت پایگاه داده مطرح شد که دارای توانمندیهایی به صورت گسترده بود.

سامانه‌ی مدیریت پایگاه داده(DBMS)

با گذشتِ زمان و نیاز به بخشِ کار با داده‌ها تابع‌های گوناگونی برای کار با داده‌ها نوشته شد. سپس کتابخانه‌های گوناگونی با توانایی‌های گوناگون آماده شدند تا کار با داده‌ها را آسان‌تر نمایند. برنامه نویس نیاز داشت که روش کار با تابع‌های درونِ یکی از این کتابخانه‌ها و چگونگی فراخوانی آنها و گرفتنِ داده‌ها را از آنها یاد بگیرد تا بتواند با آنها کار کند. همچنین نیازهای کتابخانه‌ای را که به کار برده است به خوبی بشناسد و بر پایه‌ی آن کار با داده‌ها را انجام دهد. هر کدام از این کتابخانه‌ها روشِ جداگانه‌ای را به کار می‌گرفتند و روشِ استانداردی برای کار کردنِ با همه‌ی آنها به وجود نیامده است. این گوناگونی کتابخانه‌ها همراه با توانایی‌ها و کاستی‌های گوناگونِ آنها باعث شد تا کار با داده‌ها به این روش مدتها به صورت یک مشکل بزرگ باقی بماند.

Comment [A4]: منظور از کتابخانه

بنابراین برای برطرف شدن مشکل‌های کار با داده‌ها نرم افزارهای جداگانه‌ای به نامِ کلی «سامانه‌ی مدیریت پایگاه داده (Database Management System) «یا DBMS ساخته شد. این سامانه یک سطحِ انتزاعی برای کار با داده‌ها را فراهم می‌کند و برنامه‌ها برای کار با داده‌ها به جای کمک گرفتن از سامانه‌ی مدیریتِ پرونده‌ی سیستم عامل، از سامانه‌ی مدیریت پایگاه داده‌ها کمک می‌گیرند. برنامه‌ی کاربردی با فرستادنِ پرس و جو (درخواست، تقاضا (query به سامانه‌ی مدیریت پایگاه داده کارِ خواسته شده را انجام می‌دهد. برای نمونه برنامه‌ی کاربردی داده‌های یک کتاب را درخواست می‌کند (پرس و جویی به سامانه می‌فرستد) و سامانه‌ی مدیریت پایگاه داده

Comment [A5]: در این بخش تعاریف آورده شود

داده‌های آن کتاب را به برنامه می‌دهد یا برنامه درخواست می‌کند که یک کتاب حذف شود و سامانه‌ی مدیریت پایگاه داده آن کتاب را حذف می‌کند.

مجموعه‌ای از برنامه‌های واسط میان داده‌ها (و فراداده‌ها) و برنامه‌های کاربردی (و کاربران) سامانه‌ی مدیریت پایگاه داده نامیده می‌شود. برخی پایگاه داده‌ها (مجموعه‌ای از داده‌های در ارتباط با هم) را نیز در این تعریف می‌گنجانند و سامانه‌ی مدیریت پایگاه داده‌ها را مجموعه‌ای از داده‌های در ارتباط با هم و برنامه‌هایی برای دستیابی به آنها تعریف می‌کنند.

سامانه‌ی مدیریت پایگاه داده اغلب برای کار با داده‌ها، پرس و جوی داده شده به آن را به تعدادی درخواست برای سامانه‌ی مدیریت پرونده‌ی سیستم عامل تبدیل می‌کند و برای سامانه‌ی مدیریت پرونده‌ی سیستم عامل می‌فرستد و پاسخ‌های (یا داده‌های) گرفته شده از سامانه‌ی مدیریت سیستم عامل را به شکلِ دلخواه درهم می‌آمیزد (ترکیب می‌کند) و به شکلِ خواسته شده برای برنامه می‌فرستد.

برای سادگی بیشتر از این پس «سامانه» به جای «سامانه‌ی مدیریت پایگاه داده «(DBMS) به کار برده می‌شود و دیگر سامانه‌ها با نام کامل نوشته می‌شوند. گرچه گاهی همان نام کامل به کار برده می‌شود. همچنین «برنامه» به جای «برنامه‌ی کاربردی» به کار برده می‌شود.

مزایای رویکرد پایگاهی:

- مدیریت متمرکز داده ها: دراین نظام بدلیل اینکه امکان مدیریت متمرکز داده ها وجود دادر، معیابی همچون افزونگی داده ها دیگر وجود نخواهد داشت و در نتیجه کارایی داده ها بیشتر خواهد شد.
- استقلال داده: با تغییر در یک داده نیازی به تغییر در نرم افزار و یا بالعکس نیست
- یکپارچکی نظام: بدلیل اینکه تمامی نیازهای سازمان یکپارچه بررسی و به صورت یکپارچه نیز برای آن برنامه نوشته می شود و سپس توسط سامانه مدیریت پایگاه داده کنترل می شود، افزونگی داده و دیگر مسائل پیش نمی آید.

اجزاء یک سیستم پایگاه داده:

۱- سخت افزار: برای اجرای صحیح یک سیستم پایگاه داده به حداقل سخت افزارهایی برای ذخیره، شبکه است.

۲- نرم افزار: نرم افزارهای مورد نیاز برای اجرای صحیح سیستم پایگاه داده شامل سیستم عامل، نرم افزار مدیریت سیستم پایگاه داده و برنامه های کاربردی (رابط) است.

۳- کاربران: معمولا کاربران سیستمهای پایگاه داده شامل مدیران سیستم، طراحان و برنامه نویسان پایگاه داده و کاربران نهایی پایگاه داده است.

۴- داده: منظور مجموعه داده هایی که برای کارآمد و رسیدن به اهداف پایگاه مورد نیاز است، می باشد.

معماری پایگاه داده:

معماری پایگاه داده توصیفی از موقعیت و جایگاه همه اجزایی است که پایگاه داده را تشکیل می دهند و آن را عملیاتی می کنند. معمولا در پایگاه داده این عملیات در سه سطح انجام میگیرد:

- سطح فیزیکی: چگونگی ذخیره اطلاعات وداده در محیط فیزیکی
- سطح ادراکی یا منطقی: سطحی است که برنامه نویسان بر اساس نیازهای کلیه کاربران اقدام به برنامه نویسی و طراحی پایگاه داده میکنند
- سطح خارجی: سطحی است که هر کاربر نسبت به نیاز خود به اطلاعات ذخیره شده دسترسی دارد.

محیط عملیاتی

محیط عملیاتی در هر پایگاه، محیطی است که پایگاه به منظور انجام اهداف و وظایف آن محیط طراحی و ایجاد می شود. به عنوان مثال کتابخانه می تواند به عنوان یک محیط عملیاتی باشد که در آن تمامی وظایف مجموعه سازی تا اشاعه اطلاعات انجام می گیرد.

پایگاه استنادی نیز به منظور انجام تحلیل استنادی منابع ایجاد و بایستی این اهداف را نیز به خوبی انجام دهد. پس محیط عملیاتی که برای یک پایگاه استنادی باید مورد تحلیل قرار گیرد، محیطی است که تمامی منابع

چاپی و غیر چاپی را شامل می شود. در این نوشتار منظور از محیط عملیاتی، محیطی است که شامل مقالات، کتابها و سایر منابع علمی به همراه داده ها و اطلاعات استنادی آنها است.

به منظور ایجاد یک پایگاه اطلاعاتی استنادی لازم است که علاوه بر اینکه تمامی اهداف سازمان متولی پایگاه شناسایی شود، بایستی قابلیت ها و انتظارات از پایگاه نیز بررسی و در طراحی آن لحاظ شود. به همین منظور بایستی تمامی عواملی که در پیاده سازی و دستیابی به اهداف سازمان در طراحی پایگاه می توانند نقشی را داشته باشند شناسایی شود. به این عوامل در پایگاه اطلاعاتی موجودیت گفته می شود.

مراحل بررسی محیط عملیاتی

- نیازمندی های کاربران، اهداف سازمان بررسی شود و به عبارتی تمامی وظایفی که از یک پایگاه داده انتظار دارند، مشخص شود.
- یک طرح کلی برای هر بخش طراحی شود.
- درمرحله بعدی تمامی نیازها و دیدگاهها در یک قالب پکپارچه و به صورت یک شمای واحد تهیه شود.
- دیاگرام یا فلوچارت بر اساس رابطه ها تهیه شود.
- طراحی و اجرای ساخت پایگاه در مرحله بعدی قرار دارد.

موجودیت:

هر جزئی که به عنوان یک مفهوم در طراحی پایگاه اطلاعاتی در محیط عملیاتی مورد بررسی قرار می گیرد به عنوان موجودیت نامیده می شود. به عبارتی موجودیت در هر پایگاه اطلاعاتی شامل تمامی مفاهیمی است که برای دسترسی به اهداف سازمان باید برای هر کدام یک جدول مستقل در پایگاه ایجاد نمود. بر این اساس یک موجودیت یک شی است که موجود است و قابل شناسایی برای دیگر اشیاء هم است. در محیط عملیاتی استنادی، عوامل مختلفی می توانند سازمان را در رسیدن به اهداف یاری نماید که عبارتند از :

- آثار علمی به طور کلی(شامل مقاله، کتاب، پایان نامه و ...)
- نویسنده به عنوان خالق آثار علمی
- مجلات و نشریات به عنوان منتشر کننده آثار علمی

- فهرست منابع آثار علمی به عنوان پیشینه هر اثر علمی

ویژگی ها (صفات هر موجودیت)

همانطور که در بخش موجودیت عنوان شد، برای ایجاد یک پایگاه اطلاعاتی نیاز به شناخت موجودیت ها است. به منظور معرفی هر موجودیت در پایگاه اطلاعانی نیازمند اطلاعاتی در باره هر کدام از موجودیت ها است که به این اطلاعات ویژگی یا صفت گفته می شود.

به عنوان مثال مقاله در یک پایگاه اطلاعاتی به عنوان یکی مفهوم و یا یک متغیر یک موجودیت به حساب می آید که دارای صفاتی است، به عبارت دیگر مقاله را با صفات یا ویژگیهای زیر می شناسند:

- عنوان مقاله
- نویسنده / نویسندگان
- عنوان نشریه
- جلد نشریه
- شماره نشریه
- سال انتشار
- شماره صفحه
- آدرس الکترونیکی
- فهرست منابع

انواع ویژگی ها

هر کدام از ویژگیهای یک موجودیت از نظر مفهوم، ماهیت، بسامد تکرار شدن در هر موجودیت و ؟؟؟؟ به انواع مختلفی تقسیم بندی می شود. صفات یک موجودیت بر حسب مفهوم آنها به دسته های زیر تقسیم می گردند:

صفت خاصه کلید (شناسه موجودیت)

صفت ساده یا مرکب

تک مقداری یا چند مقداری

ذخیره شده (واقعی یا مبنا) یا مشتق

هیچ مقدار پذیر یا هیچ مقدار ناپذیر

ویژگی کلید

یک یا چند صفت که در یک موجودیت قابلیت منحصر به فرد کردن مصداق های هرکدام از موجودیت ها را از همدیگر دارد. با توجه با اینکه این صفت دارای تقسیم بندی متعددی است و به منظور خاصی از آن استفاده می شود .

مثال: در موجودیت مقاله، شماره مقاله به عنوان ویژگی کلید در نظر گرفته می شود (جدول ۱)، که باعث منجصربه فرد شدن مقالات از یکدیگر می گردد.

ویژگی ساده

ویژگی ساده نوعی از ویژگی است که از یک جزء تشکیل شده است و تعداد داده های تعلق گرفته برای هر موجودیت تنها یکی است. به عبارت دیگر صفت ساده صفتی است که به اجزاء کوچکتر تجزیه پذیر نباشد.

مثال: عنوان اصلی مقاله

جدول ۱

شماره مقاله	عنوان اصلی مقاله
۱	استنناد به انتشارات ملی
۲	داده کاوی و کاربرد آن در تصمیم گیری
۳	استفاده از تکنیک داده کاوی

ویژگی مرکب:

نوعی از ویژگی که خود دارای چند جزء دیگر است و بتواند به اجزا کوچکتر تجزیه پذیر باشد. به عنوان مثال ویژگی آدرس برای یک موجودیت دارای اجزایی از قبیل شهر، خیابان، پلاک و غیره است (نمودار ؟؟؟؟)، یا شماره صفحات یک مقاله که از دو جزء صفحه آغاز مقاله و صفحه پایانی مقاله تشکیل شده است.

شماره صفحات مقاله		عنوان مقاله	شماره مقاله
شماره صفحه پایان مقاله	شماره صفحه آغاز مقاله		
۳۵۶	۳۳۷	استتناد به انتشارات ملی	۱
۱۴	۳	داده کاوی و کاربرد آن در تصمیم گیری	۲
۴۶	۵۵	استفاده از تکنیک داده کاوی	۳

ویژگی تک مقداری:

صفاتی که فقط یک مقدار را در هر لحظه از زمان به خود اختصاص دهند به صفات تک مقداری معروفند.

به عنوان مثال کد ملی، تاریخ انتشار مقاله در زمان ورود اطلاعات فقط یک مقدار را به خود اختصاص می دهند.

سال انتشار مقاله	عنوان مقاله	شماره مقاله
۱۳۹۳	استتناد به انتشارات ملی	۱
۱۳۸۶	داده کاوی و کاربرد آن در تصمیم گیری	۲
۱۳۹۲	استفاده از تکنیک داده کاوی	۳

ویژگی چند مقداری یا تکرار پذیر:

صفت چند مقداری صفتی است که برای حداقل یک نمونه از مصداق یک موجودیت، بیش از یک مقدار یا یک داده را به خود اختصاص می دهد. به بیان ساده تر، به ازاء یک ویژگی، چند مقدار، برای یک نمونه از موجودیت دارد.

مثال: صفت خاصه نام نویسنده برای هر مقاله صفت چند مقداری محسوب می شود. زیرا یک مقاله می تواند دارای چند نویسنده باشد،

ویژگی تکرار پذیر		
نام خانوادگی	نام	شماره مقاله
توکلی زاده راوری	محمد	۱
حسین میرزایی	لیلا	۱
سهیلی	فرامرز	۱

ویژگی مشتق:

صفتی است که در موجودیت وجود خارجی ندارد ولی در صورت لزوم می توان آن را بدست آورد. به صفت مشتق گاه صفت مجازی یا صفت محاسبه شدنی نیز می گوییم.

مثال: برای یک مقاله، از ویژگی صفحه آغاز و پایان آن می توان ویژگی جدیدی را با عنوان تعداد صفحات مقاله ایجاد کرد، درحالی که این ویژگی از ابتدا برای مقاله تعریف نشده است.

تعداد صفحات مقاله (مشتق)	شماره صفحه پایان مقاله	شماره صفحه آغاز مقاله	عنوان مقاله	شماره مقاله
۱۹	۳۵۶	۳۳۷	استناد به انتشارات ملی	۱
۱۱	۱۴	۳	داده کاوی و کاربرد آن در تصمیم گیری	۲
۹	۴۶	۵۵	استفاده از تکنیک داده کاوی	۳

ویژی هیچمقدار پذیر:

صفاتی که از نوع هیچ مقدار پذیر هستند این امکان را دارند که در صورت نبود اطلاعات مورد نیاز و یا به دلایل دیگر خالی بمانند، به عبارت دیگر داده ای ندارند که معمولا با عنوان Null شناخته می شوند. در واقع اگر مقدار یک صفت در یک یا بیش از یک نمونه از یک نوع موجودیت خالی و فاقد داده باشد آن را صفت هیچمقدار پذیر است.

مثال: ویژگی DOI یک مقاله که ممکن است در زمان ورود اطلاعات در دسترس نباشد بنابراین سیستم به نرم افزار اجازه می دهد که این ویژگی بدون مقدار ذخیره شود.

DOI	عنوان مقاله	شماره مقاله
	استناد به انتشارات ملی	۱
	داده کاوی و کاربرد آن در تصمیم گیری	۲
	استفاده از تکنیک داده کاوی	۳

آموزش ساخت پایگاه با استفاده از SQL

بعد از اینکه محیط عملیاتی بررسی و مولفه های تاثیر گذار در پایگاه به همراه اهداف پایگاه شناسایی شد، نوبت به اجرای پایگاه داده است. در این کتاب، از نسخه SQL Server R2 جهت طراحی و آموزش ایجاد پایگاه اطلاعاتی استفاده شده است. محیط عملیاتی که در ای کتاب مورد بررسی قرار گرفته و برای آن پایگاه داده نوشته شده است، پایگاه استنادی مقالات است. در این نوشتار سعی شده است که تمامی دستورات SQl در جهت طراحی پایگاه داده مذکور مورد استفاده قرار گیرد، ضمن اینکه تلاش شده است که دانشجویان با مطالعه این کتاب به طور عملی نسبت به طراحی پایگاه اطلاعاتی استنادی اقدام نمایند.

قبل از شروع آموزش SQL لازم است مفهوم نوع داده مشخص شود. در این بخش نوع داده هایی که بیشتر در حوزه علم سنجی مورد استفاده قرار می گیرد ارائه شده است.

داده ی رشته ای:Character Strings

از این نوع داده بیشتر برای نگهداری متن و به عبارتی تمامی علائم، نشانه ها، حروف و اعداد به صورت مستقل یا ترکیبی استفاده می شود. نوع داده رشته ای به انواع مختلف تقسیم می شود:

Char. :

اطلاعات متنی با طول ثابت از ۱ تا حداکثر ۸۰۰۰ حرف را در خود ذخیره می نماید. به طور مثال اگر برای عنوان مقاله ۱۰۰ حرف در نظر گرفته شود ولی عنوان مقاله ۵۰ حرف داشته باشد، فضای تمامی ۱۰۰حرف از حافظه اشغال خواهد شد. برای حل این مشکل از گونه دیگری از نوع داده استفاده می شود.

Varchar. :

اطلاعات متنی با طول متغیر از ۱ تا حداکثر ۸۰۰۰ حرف را در خود ذخیره می نماید. در این نوع از نوع داده، متن هایی ذخیره شده به اندازه مقدار وارد شده از حافظه را به خود اختصاص می دهد و بر خلاف نوع داده ای char ثابت نبوده، بلکه به صورت متغییر به اندازه حافظه مورد استفاده فضا را اشغال می کند.

Varchar Max :

اطلاعات از ۱ تا ۲ مگا حرف ذخیره می گردد. معمولا در پایگاه استنادی، برای فیلدهایی همچون چکیده یا متن کامل مقاله از این نوع داده استفاده می شود که محدودیت مقدارد ندارد.

تمامی مواردی که در بالا برای داده های رشته ای عنوان شد، برای زبان لاتین بود و به عبارتی از حروف نوشتاری زبانهایی پشتیبانی می کند که استاندارد ASCII از آنها پشتیبانی می کند. اما در شرایطی که داده ها به زبانی غیر از زبان نوشتاری غیر لاتین از قبیل فارسی نوشته می شود، را پشتیبانی نمی کند و در صورت ذخیره داده به زبان های مذکور داده ها به درستی نمایش داده نخواهد شد، بنابراین باید از استانداری استفاده کرد که این قبیل زبانها را شناسایی و پشتیبانی می کند که معمولا از استاندارد UNICOD استفاده می شود. بنابراین تمامی نوع داده بالا با اضافه نمودن n قابلیت ورود داده به زبان فارسی را پیدا می کنند.

nchar : در این نوع داده، اطلاعات از ۱ تا حداکثر ۴۰۰۰ حرف با طول ثابت ذخیره می گردد.

nvarchar : در این نوع داده، اطلاعات از ۱ تا حداکثر ۴۰۰۰ بایت با طول متغیر ذخیره می‌گردد.

nvarchar-max : در این نوع داده از ۱ تا ۱ مگا حرف ذخیره می گردد. مکانیزم آن هم بصورت Pointer است.

داده ی باینری:Binary types

این نوع فیلدها جهت حفظ اطلاعات بصورت باینری مثل تصاویر مناسب می باشد.

bit : یک فیلد دو بیتی می باشد و قادر می باشد ۰ و ۱ و Null را ذخیره نماید.کاربرد آن در زمان‌هایی می باشد که دو وضعیت وجود داشته باشد. مثل جنسیت ISI یا غیر ISI.

binary : این نوع فیلدها، از ۱ تا ۸۰۰۰ بایت را در خود ذخیره می نمایند.

varbinary : این نوع فیلدها هم از ۱ تا ۸۰۰۰ بایت را در خود حفظ می کنند. (متغیر).

داده ی عددی:Number types

این نوع فیلد جهت حفظ اعداد صحیح و با اعشار استفاده خواهد شد و دارای ۴ نوع به شرح زیر می باشد. از این نوع داده معمولا برای فیلدهایی شبیه به شماره مقاله، شماره نویسنده، تعداد استناد، تعداد فهرست منابع، تعداد صفحات و مانند آن که ماهیتی عددی دارند و نیازمند عملیات ریاضی در مواقع مورد نیاز است استفاده می شود. این نوع داده خود به انواع مختلفی تقسیم می شود:

tinyint : یک بایت را اشغال می نماید و قادر می باشد از ۰ تا ۲۵۵ را در خود ذخیره نماید.

smallint : یک عدد دو بایتی می باشد و قادر است از ۳۲۷۶۷ منفی تا ۳۲۷۶۷ مثبت را در خود ذخیره نماید.

int : یک عدد چهار بایتی می باشد که قادر است اعداد بین مثبت و منفی ۲ میلیارد را در خود حفظ نماید.

bigint : یک عدد ۸ بایتی می باشد که قادر است اعداد بین مثبت و منفی ۴ میلیارد را در خود حفظ نماید.

decimal-p,s : این نوع فیلد جهت حفظ نمودن اعداد اعشاری با تعداد اعشار مشخص استفاده می شود

float-n : یک عدد ۸ بایتی می باشد که اعداد بصورت توانی از ۱۰ در آن حفظ خواهند شد. .

real : یک عدد ۴ بایتی می باشد که اعداد بصورت توانی از ۱۰ حفظ خواهند شد.

داده ی تاریخ:Date types

این نوع فیلدها جهت حفظ تاریخ میلادی و ساعت استفاده خواهد شد و جهت تاریخ شمسی کاربرد نخواهد داشت، پیشنهاد می شود برای تاریخ شمسی از نوع داده ای char استفاده شود چرا که در تاریخ شمسی ممکن است از نشانه / استفاده شود.

datetime : این نوع فیلد، ۸ بایتی می باشد و از سال ۱۷۰۰ تا ۹۹۹۹ را با دقت هزارم ثانیه ذخیره می نماید.

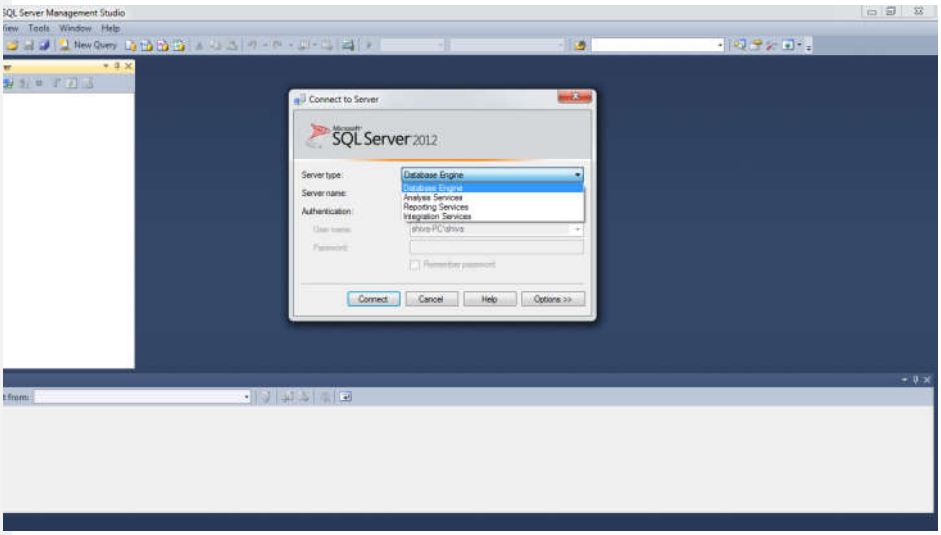
smalldatetime : این نوع فیلد، ۴ بایتی می باشد و از سال ۱۹۰۰ تا ۲۰۷۹ را با دقت هزارم ثانیه ذخیره می نماید.

date : این نوع فیلدها جهت نگهداری تاریخ میلادی استفاده خواهد شد.

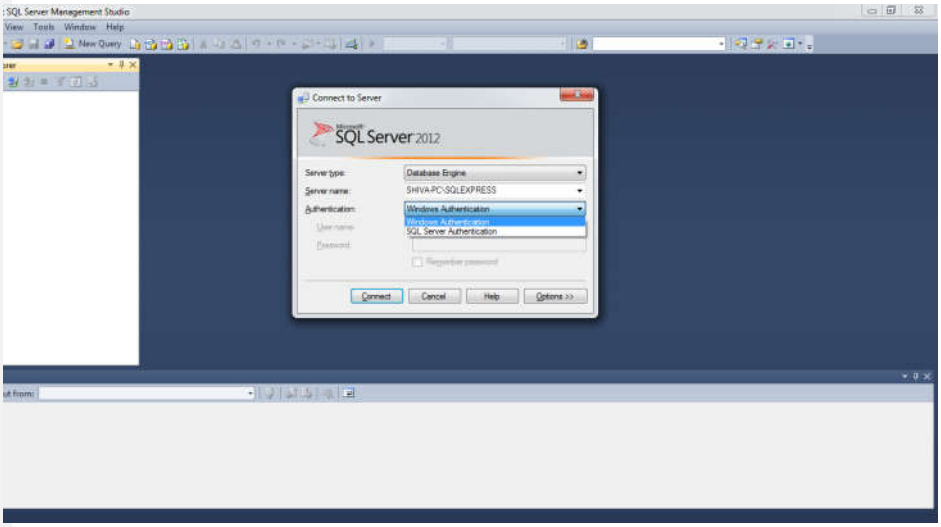
time : این نوع فیلدها جهت حفظ ساعت استفاده خواهد شد.

xml : این فیلد بیشتر جهت منتقل نمودن اطلاعات و دستورات تحت web استفاده می گردذ و شامل انواع MetaData های مختلف می باشد.

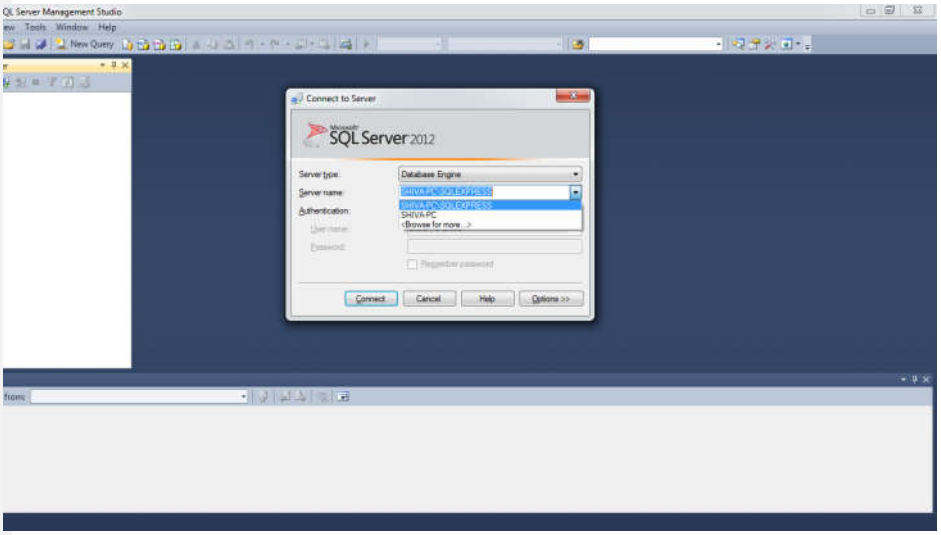
بعد از مطالعه محیط علمیاتی و شناسایی موجودیت های این محیط، در مرحله بعد باید نسبت به نوشتن دستورات مورد نظر در محیط برنامه ای اقدام شود. به همین منظور ابتدا باید بعد از نصب نرم افزار مذکور، اقدام به راه اندازی آن نمود. به همین منظور بعد از اجرای نرم افزار SQL نوبت به انتخاب نوع سرور می رسد که در با توجه به اهداف این درس (ایجاد پایگاه اطلاعاتی) گزینه Database Engine (تصویر ۱) را انتخاب می کنیم. بعد از انتخاب نوع سرور، بایستی نام سرور که در موقع نصب اعلام کرده بودیم ، انتخاب (تصویر ۲) و بعد از آن نحوه احراز هویت کاربر اعلام می شود که بستگی به سیاست مدیر پایگاه داده می تواند از طریق خود سیستم عامل (ویندوز) و یا از طریق SQL باشد (تصویر ۳). بعد از انتخاب تمامی گزینه ها، گزینه Connect را کلیک میکنیم، در این صورت برنامه آماده نوشتن و اجرای دستورات است.



شکل ۱. انتخاب source type



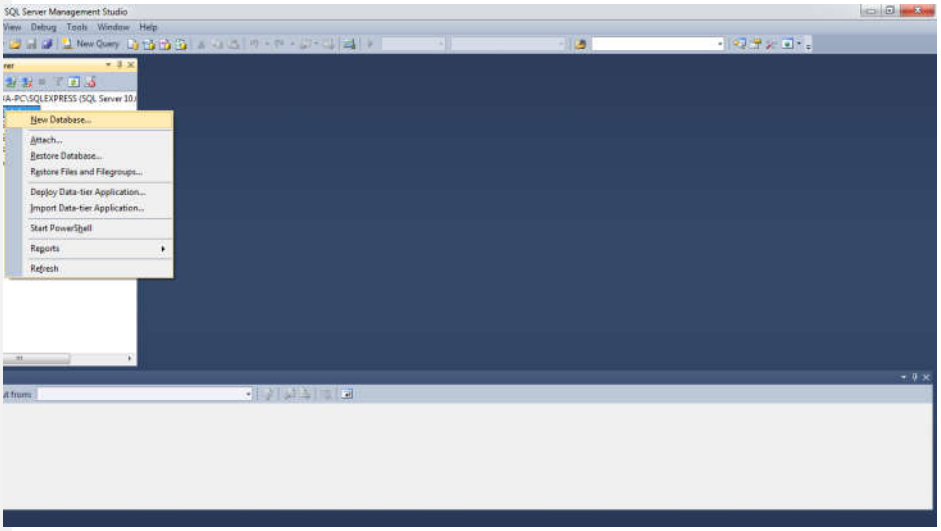
شکل ۳. انتخاب Authentication



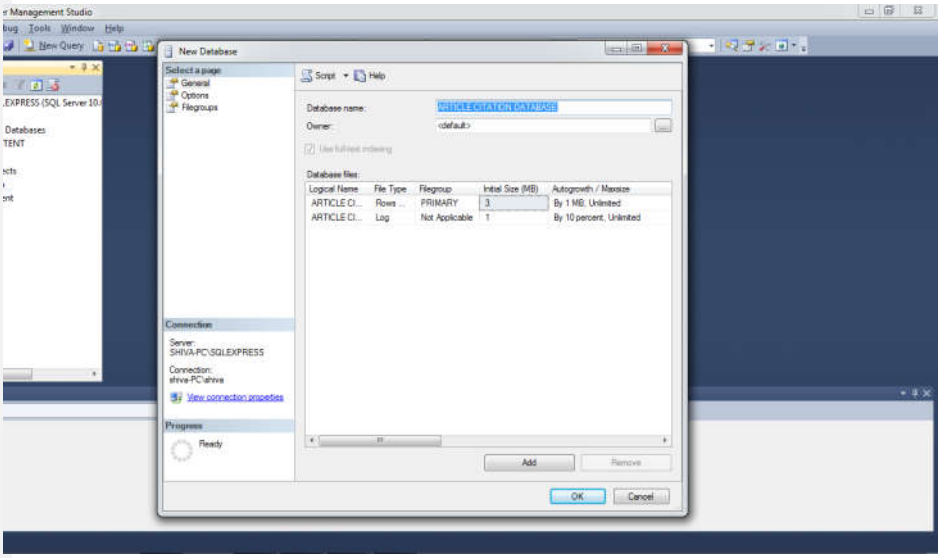
شکل ۲. انتخاب نام سرور

ایجاد پایگاه جدید

اولین قدم در اجرای پایگاه داده، ساخت پایگاه داده به نرم افزار است. به همین منظور ابتدا باید محیط عملیاتی در قالب پایگاه داده با یک نام معرفی کرد. به همین منظور از دو شیوه امکان معرفی پایگاه داده وجود دارد. روش اول: از طریق ویزارد: در این روش با کلیک راست روی Database گزینه New database را انتخاب (تصویر۵). سپس با وارد کردن نام پایگاه داده گزینه ذخیره را انتخاب می کنیم. در صورت ساخت پایگاه داده به درستی پیغام اجرای صحیح درخواست اعلام می شود.



شکل ۵. ایجاد پایگاه جدید



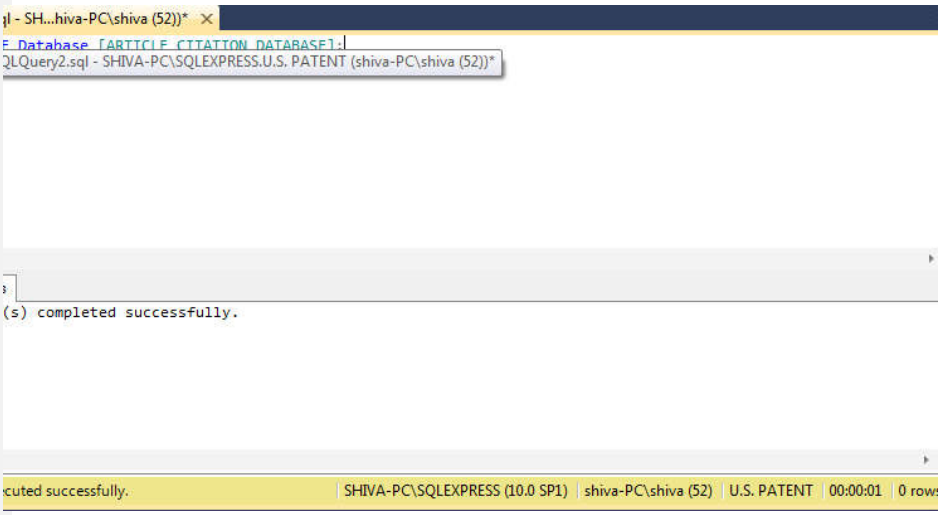
شکل ۶. ایجاد پایگاه با استفاده ازویزارد

روش ساخت پایگاه داده با استفاده از دستور

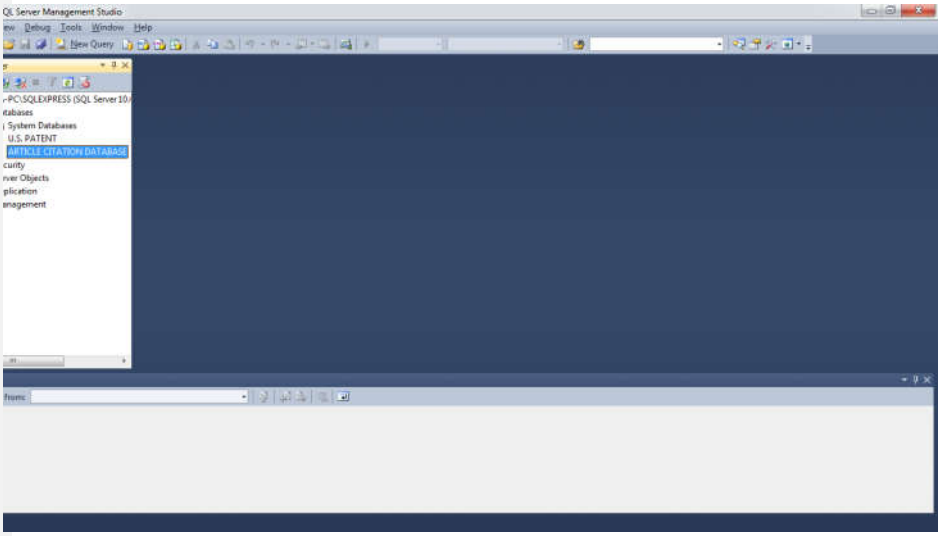
در این روش با استفاده از گزینه New Query فضای نگارش دستور آماده می شود. دستور ساخت پایگاه داده به صورت زیر است:

CREATE DATABASE DatabaseName

بعد از نگارش دستور، ایکون Execute را کلیک میکنیم. در صورت نگارش دستور به درستی، پیغام ساخت پایگاه داده در بخش پیام ها ظاهر می شود (تصویر ۷). بعد از ساخت پایگاه داده با کلیک علامت + کنار Database پایگاههای ساخته شده ظاهر می شود. در صورتی که پایگاه ساخته شده، در این بخش نمایش داده نشد با انتخاب Database و انتخاب کلید F5 و یا انتخاب گزینه Refresh تمامی پایگاهها نمایش داده خواهد شد(تصویر ۸).



شکل۷. ایجاد پایگاه جدید با استفاده از دستور

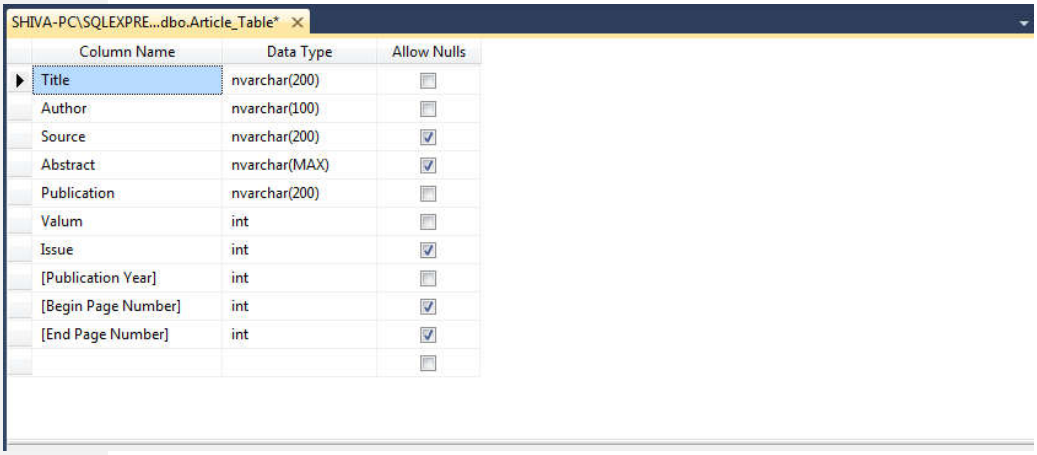


شکل ۸. پایگاه ایجاد شده

دستور CREATE یا ایجاد جدول

بعد از ساخت پایگاه داده، نوبت به ایجاد جداول برای هر کدام از موجودیت ها می رسد. شبیه به روشهای ساخت پایگاه داده، ایجاد جدول نیز از دو روش آماده و دستوری امکان پذیر است. به منظور ساخت جدول به روش آماده، بعد از انتخاب پایگاه داده ایجاد شده و انتخاب نشانه کنار نام پایگاه داده، بر روی گزینه Table کلیک راست نموده و گزینه New Table را انتخاب می کنیم (تصویر ۹). همانطور که قبلا عنوان شد هر موجودیت دارای ویژگی هایی است که در زمان ایجاد جدول باید این ویژگی ها در قالب ستونهایی در جداول لحاظ شود. در موقع ایجاد ستونها، به سه جزء اطلاعاتی نیاز است: نام ستون، نوع داده، مقدار داده. با رعایت این سه اصل، می توان به صورت آماده اقدام به ایجاد جدول با ستونهای مورد نیاز کرد.

به طور مثال اگر بخواهیم برای موجودیت مقاله جدولی را ایجاد کنیم، ابتدا باید فیلدها یا ستونهای جدول مذکور را شناسایی کنیم. هر مقاله دارای حداقل اطلاعاتی از قبیل عنوان، نویسنده، نشریه، جلد، شماره، صفحه، چکیده است که بایستی تمامی ستونها را درجدول مقاله ایجاد کرد. به همین منظور با رعایت اصل نوع داده و مقدار داده مورد اقدام به معرفی ستونها درجدول مذکورمی کنیم (تصویر ۱۰).



شکل ۱۰. ایجاد و نامگذاری فیلد ها در جدول

روش دیگر در ایجاد جدول استفاده از دستور است. به منظور ایجاد جدول باید از دستور Create استفاده کرد. ساختار دستور مذکور به روش زیر است:

Create Table tablename (Column name1, Column name2, …, Column name n)

برای معرفی هر کدام از ستونها باید ساختار زیر را رعایت کرد:

Column name Data Type (value)

به عنوان مثال اگر ستون "عنوان مقاله" را بخواهیم در جدول مذکور معرفی کنیم باید ابتدا نامی برای آن انتخاب کنیم در این مثال ما واژه "Title" را انتخاب می کنیم. سپس باید نوع داده انتخاب شود. با توجه به اینکه در این فیلد یا ستون قرار است که عناوین مقالات ذخیره شود و عناوین هم از نوع داده ای رشته است، به عبارتی حروف و ممکن است اعداد و یا علایم را شامل شود باید نوع داده ای Char را انتخاب کنیم. اما با توجه به اینکه مقدار هر عنوان به طور ثابت مشخص نیست باید از نوع داده ای Varchar به جای Char استفاده کنیم که نشان دهنده فضای نسبی برای هر کدام از داده ها خواهد بود. اما چون ممکن است عناوین مقاله های گردآوری شده برای ذخیره به زبان نوشتاری غیر از زبان لاتین باشد از نوع داده ای nvarchar استفاده می کنیم. از طرف دیگر

باید حداکثر مقدار برای این فیلد یا ستون باید انتخاب شود، بنابراین باید مقداری را انتخاب کنیم که تا آنجا که ممکن است تمامی داده های هر عنوان را کامل وارد کند. به عنوان مثال اگر تعدادی از عناوین مقاله ها بیش از ۲۰۰ کاراتر را شامل بشوند و ما در معرفی مقدار داده عدد ۱۵۰ را ثبت کنیم، درموقع ورود داده بخشی از داده ها ذخیره نخواهد شد. بنابراین بهتر است که مقداری را معرفی کنیم که تا آنجایی که ممکن است داده را از دست ندهیم. با این تفاسیر شیوه معرفی ستون عنوان مقاله به شیوه زیر است:

Title *nvarchar (200),*

در معرفی ستون های هر جدول بعد از نوشتن دستور هر ستون باید از علامت "،" باید استفاده کرد. در ادامه دستورایجاد جدول مقاله به همراه دستورات ایجاد ستونهای جدول مذکور آمده است.

```
CREATE TABLE Article_Table (
        Id_Article int,
     Title nvarchar (200),
    Author nvarchar (100),
[Source] nvarchar (200),
  Abstract nvarchar (max),
Publication nvarchar (200),
    Volume int,
     Issue int,
[Publication Year] int,
[Begin Page Number] int,
[End Page Number] int)
```

بعد از اجرای دستورات با کلید بر روی گزینه Execute دستورات اجرا می شود. در صورتیکه که اشکالی در نگارش دستورات نباشد پیام ایجاد جدول به درستی و با موفقیت اعلام میگردد.

قید NULL

در صورتی که این امکان وجود داشته باشد که در هنگام ورود داده بعلت عدم وجود داده، ردیف مربوطه می تواند خالی بماند از دستور NULL استفاده می شود. به منظور ایجاد امکان خالی بودن یک ستون در هنگام ورود داده می توان از دو گزینه آماده و دستوری استفاده کرد. معمولا در استانداردهای فهرستنویسی عنوان یک

اثر نمی تواند در هنگام ورود داده خالی باشد، بنابراین برای فیلد عنوان نمی توان اجازه داد که این فیلد خالی باشد، به همین خاطر بجای اینکه دستور NULL از NOT NULL استفاده می شود، به عبارت دیگر قید not null بر روی صفت Title از جدول Article تضمین می کند که نام مقاله نمی تواند خالی باشد. ساختار نوشتاری آن به صورت زیر است:

Title *nvarchar (200) NOT NULL*,

کلید اصلی Primary Key

همانطور که در بخش انواع کلید ها عنوان شد، برای ورود داده ها باید از امکانی استفاده کرد که داده ها تکراری وارد نشود و یا اینکه داده های مشابه به هم قابلیت شناسایی داشته باشند، به همین خاطر از کلیدی به نام کلید یکتا یا Primary Key استفاده میکنند که از ویژگی های آن عدم تکرار پذیری و تهی نبودن آن است. به عبارت دیگر درهنگام ورود داده در ستون کلید اصلی، داده های موجود در این ستون نمی توانند خالی باشند و همچنین نباید تکراری باشند. به همین خاطر معمولا یکی از صفات یا ویژگی های هر موجودیت را به عنوان کلید اصلی انتخاب میکنند که بتواند این دو قابلیت را داشته باشد. در مثال مقاله همانطور که عنوان شد صفات یا ویژگی های عنوان، نویسنده، منبع، چکیده و مانند آن وجود دارد که با دقت در ماهیت هر کدام از ویژگی ها ممکن است تکرار پذیر باشد. به عنوان مثال ممکن است مقالاتی باشند که عنوان مشابه داشته باشند یا اینکه نویسندگان آن یک فرد مشخص باشد و یا اینکه در یک منبع ممکن است چندین مقاله منتشر شود. بنابراین هیچ کدام از این ویژگی ها نمی توانند به عنوان کلید اصلی انتخاب شود، بایستی از یک ویژگی دیگر استفاده کرد. برای مثال فوق کلید شماره مقاله را انتخاب می کنیم که نسبت به هرمقاله مقاوت است و شماره ای منحصر به فرد را برای هر مقاله در نظر میگرد.

برای اعمال قید کلید اصلی به هر ستون می توان از طریق کلیک راست بر روی هر ستون و انتخاب گزینه primary Key و یا با اضافه کردن قید primary Key به دستور ایجاد هر ستون استفاده کرد. دستور فوق بدین صورت است:

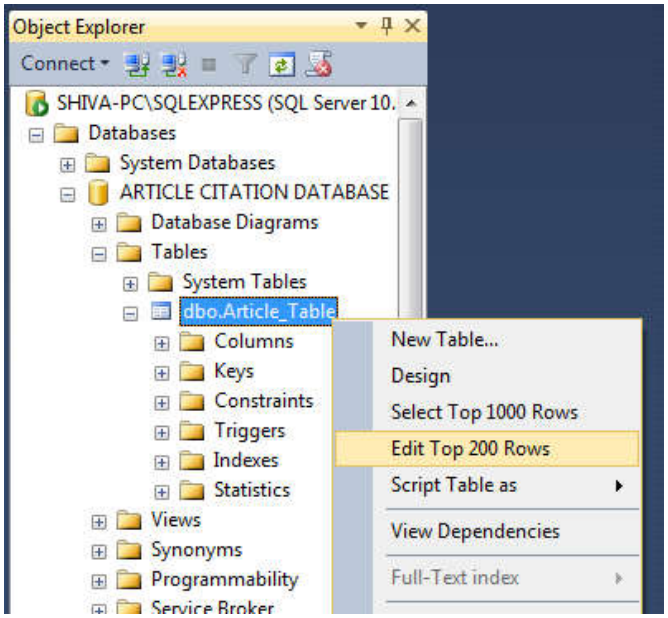
Article_Id Int *NOT NULL Primary Key*,

ورود داده

بعد از اینکه جدول ایجاد شد، بایستی داده ها در آن وارد شود تا قبلیت جستجو پیدا بکند. بنابراین در بخش نسبت به چگونگی ورود داده، بحث خواهد شد.

دستور INSERT

به منظور ورود داده همانند مراحل قبلی می توان از دو امکان آماده و دستوری عملیات ورود داده را انجام داد. در ابتدا روش ورود داده به استفاده از امکان ویزارد را توضیح می دهیم و سپس به ورود داده به وسیله دستور را توضیح خواهیم داد. به منظور ورود داده از طریق ویزارد با کلیک راست روی عنوان جدول که در مثال فوق Article است، گزینه Edit Top 200 Rows را انتخاب می نماییم (تصویر۱۳).



شکل ۱۳. ورود و اصلاح رکورد

برای ورد داده به شیوه دستوری از INSERT استفاده می شود. ساختار دستور مذکور به روش زیر است:

INSERT INTO TableName (ColumnName1, Column

نمونه دستور INSERT

دستورINSERT INTO

```
USE [ARTICLE CITATION DATABASE]
INSERT INTO Article_Table (Id_Article, Title, Author, [Source], Abstract, Publication, Valum, Issue , [Publication Year] , [Begin Page Number] , [End Page Number])
VALUES (4 ,N'ایران در کتابداری دانش ثمردهی و گیری شکل داستان از ای گوشه',N'دیانی محمدحسین مهراد، جعفر اطلاع و کتابداری فصلنامه',N'رسانی اطلاع و کتابداری فصلنامه',N'',N'رسانی',12,4,1388 ,5,7);
```

Author	Source	Abstract	Publication	Valum	Issue	Publication Year	Begin Page Number	End Page Number
عباس حری	ماهنامه کتاب ماه کلیات	در رشته ما که همیشه سروکار ش...	ماهنامه کتاب ماه کلیات	16	2	1391	2	3
عباس حری	ماهنامه کتاب ماه کلیات	دکتر استاد که آزاد دیگر در میان ما	ماهنامه کتاب ماه کلیات	NULL	2	1389	82	82
ابراهیم پورسلطانی راد، عباس حری	اطلاع شناسی	هدف اصلی این پژوهش مقایسه ک...	اطلاع شناسی	21	4	1387	133	164
جعفر مهراد، محمدحسین دیانی	فصلنامه کتابداری و ا...		فصلنامه کتابداری و اطلاع رسانی	12	4	1388	5	7

شکل ۱۵.

دستور SELECT

```
USE [ARTICLE CITATION DATABASE];
SELECT * FROM Article_Table;
```

ویا

```
USE [ARTICLE CITATION DATABASE];
SELECT Id_Article, Title, Author, [Source], Abstract, Publication, Valum, Issue , [Publication Year] , [Begin Page Number] , [End Page Number]
```

```
FROM Article_table;
```

شکل ۱۴. جستجوی کلی در پایگاه داده

I d_article	itle	A uthor	S ource	A bstract	Pu blication	V olume	ssue	]Pu blication Year[	] Begin Page Number[	] End Page [Number

Auth or_Id	Na me	Last] [Name	Affili ation	cou ntry	city	Ema il

Source_Id	Title	Publisher	Editor in] [Cheif