ایجاد بانکهای اطلاعاتی

فصل اول:
مفاهیم پایگاه داده

مفاهیم پایگاه داده ها

اصطلاح پایگاه داده‌ها یکی از رایج‌ترین اصطلاحات در دانش و فن کامپیوتر است. همچنین خیلی اوقات بجای این واژه از اصطلاح معادل بانک اطلاعات استفاده می‌شود. بطور خیلی ساده پایگاه داده محلی است برای نگهداری داده و علم پایگاه داده‌ها کلیه عملیات و مفاهیم مرتبط در این خصوص را شامل می‌شود. اما سوالی که در همین ابتدا پیش می‌آید این است که منظور از نگهداری داده چیست؟ بعبارتی نوع این نگهداری یک ذخیره پایدار (مانند فایل) است یا یک ذخیره موقت در حافظه (مانند آرایه‌ها و ...)؟ برای پاسخ به این سوال باید بگوییم که سیستم مدیریت بانک اطلاعات همانند سیستم فایل یکی از سیستم‌های ذخیره و بازیابی اطلاعات است. سوال دومی که ممکن است برای افرادی که تجربه کار با فایلها را دارند پیش بیاید این است که چه تفاوتی میان فایل و پایگاه داده وجود دارد. در این فصل، ضمن معرفی مفاهیم اولیه تفاوت این دو سیستم را مورد بحث قرار خواهیم داد.

تعاریف اولیه:

سیستم ذخیره و بازیابی اطلاعات:

هر سیستمی که به کاربر عادی یا برنامه نویس امکان دهد تا اطلاعات خود را ذخیره، بازیابی و پردازش کند.

تعریف داده:

تعریف اول- هر مجموعه‌ای از داشته ها (دانستنیها)

تعریف دوم- نمایش ذخیره‌شده اشیاء فیزیکی، چیزهای مجرد، داشته‌ها، رویدادها یا چیزهای قابل مشاهده که در تصمیم‌سازی بکار می‌آیند.

تعریف سوم- دانستنیهای خام که معنای اندکی دارند؛ مگر اینکه به صورت منطقی سازمان‌دهی شده باشند.

تعریف چهارم (از دیدگاه ANSI) - نمایش پدیده‌ها، مفاهیم یا شناخته‌ها به طرزی صوری و مناسب برای برقراری ارتباط، تفسیر یا پردازش توسط انسان یا بطور خودکار

البته تعاریف دیگری هم برای داده ارائه شده است که به همین موارد اکتفا می‌کنیم.

تعریف اطلاعات (اطلاع)

بطور خیلی ساده اطلاعات، داده پردازش‌شده است. یعنی بر خلاف داده خام، اطلاعات داده سازمان یافته‌ای است که شناختی را منتقل می‌کند و برای تصمیم‌گیری نیز بکار می‌رود.

تعریف پایگاه داده‌ها (بانک اطلاعات)

مجموعه‌ای است از داده‌های ذخیره شده و پایا، به صورت مجتمع(یکپارچه) (نه لزوما فیزیکی، بلکه حداقل به طور منطقی)، بهم مرتبط، با کمترین افزونگی، تحت مدیریت یک سیستم کنترل متمرکز. پایگاه داده می‌تواند مورد استفاده یک یا چند کاربر به طور همزمان و اشتراکی قرار گیرد.

تعریف سیستم مدیریت پایگاه داده‌ها (DBMS)

یک نرم‌افزار واسط بین محیط فیزیکی ذخیره و بازیابی اطلاعات و محیط منطقی برنامه‌سازی در سیستم بانک اطلاعات است که هرگونه ارتباط بین کاربر و داده را کاملا کنترل و مدیریت می‌کند.

DBMS به کاربر امکان می‌دهد تا پایگاه داده‌های خود را تعریف کند، در پایگاه داده‌های خود عملیات انجام دهد و روی پایگاه داده‌های خود تا حدی کنترل داشته باشد.

پایگاه داده در مقابل فایل

برای ایجاد یک برنامه کاربردی دو روش وجود دارد. روش اول همان روش سنتی یا اصطلاحا فایلینگ است. در این روش، با استفاده از دستورات موجود در یک زبان برنامه نویسی، عملیات ایجاد فایل، خواندن و نوشتن اطلاعات در فایل، حذف فایل و ... انجام می‌شود. در واقع در یک برنامه، ساختاری توسط برنامه نویس تعریف می‌شود، فایلی با آن ساختار بر روی دیسک ایجاد شده و این فایل جهت عملیات ذخیره و بازیابی اطلاعات مورد استفاده قرار می‌گیرد. مدیریت و کنترل اجرای عملیات روی فایل برعهده بخشی از سیستم عامل بنام سیستم فایل است. سیستم فایل تنها بخش مدیریتی است که در این روش بر روی عملیات کنترل دارد. اما این کنترل یک کنترل ضعیف است و همچنین محدودیت‌هایی را برای استفاده همزمان کاربران ایجاد می‌کند که در ادامه به آنها اشاره خواهیم کرد.

اما روش دوم روش پایگاهی است. در این روش از پایگاه داده‌ها جهت ذخیره و بازیابی داده‌ها استفاده می‌کنیم و عملیات خود را تحت کنترل یک سیستم مدیریتی قدرتمند بنام سیستم مدیریت بانک اطلاعات (DBMS) انجام می‌دهیم.

سوال مهمی که در اینجا باید به آن پاسخ داد این است که مگر اطلاعاتی که می خواهد بر روی دیسک ذخیره شود در قالب فایل ذخیره نمی‌شود؟ پس پایگاه داده چه فرقی با فایل دارد؟ پاسخ این است که بله، درست است که هر چیزی که می خواهد ذخیره شود، در نهایت بصورت فایل ذخیره می شود. اما تفاوت اینجاست که در روش فایلینگ ما مستقیما با فایل کار می‌کنیم و فایل ما دقیقا همان ساختاری را دارد که ما در برنامه تعریف می‌کنیم. اما در هنگام استفاده از سیستم مدیریت بانک اطلاعات، ما از طریق یک واسط، ساختار داده‌های مورد نظر خود را تعریف می‌کنیم. بعد از آن، این سیستم مدیریت بانک اطلاعات است که داده های ما را پس از نگاشتهای مختلف در فایلهایی ذخیره می‌کند که استفاده از محتوای آن فایلها فقط از طریق خود DBMS امکان پذیراست.

برای روشن‌تر شدن بحث، در ادامه مهمترین تفاوت‌های این دو نوع سیستم ذخیره و بازیابی را شرح می دهیم.

معایب روش فایلینگ (پرونده ای) نسبت به روش پایگاهی

۱- عدم وجود محیط مجتمع ذخیره‌سازی اطلاعات و عدم وجود سیستم یکپارچه

در سیستم فایلینگ، یک ساختار واحد و یکپارچه برای ذخیره تمام فایل‌های مورد نیاز وجو ندارد و فایلهایی که تعریف می‌شوند، بصورت مجزا بر روی دیسک ذخیره می‌شوند. اما در روش پایگاهی اطلاعات بصورت یکپارچه و مرتبط ذخیره می‌شوند.

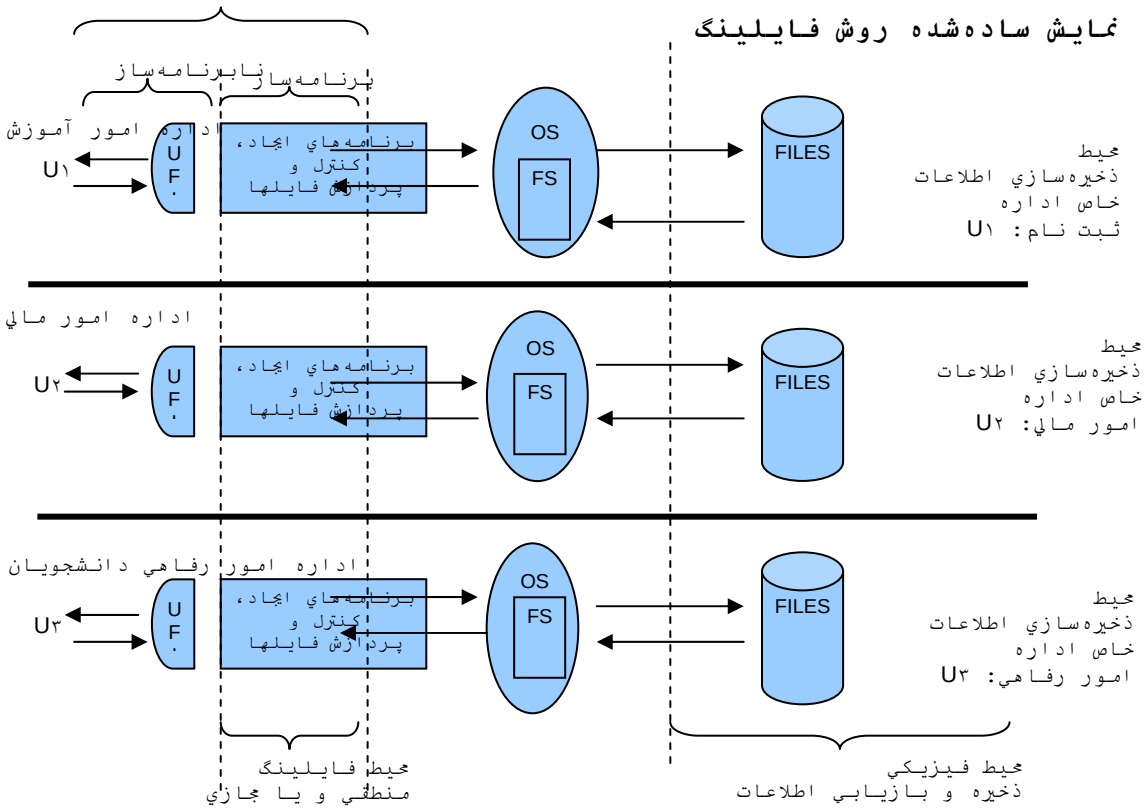
۲- افزونگی داده و عدم وجود سیستم کنترل متمرکز

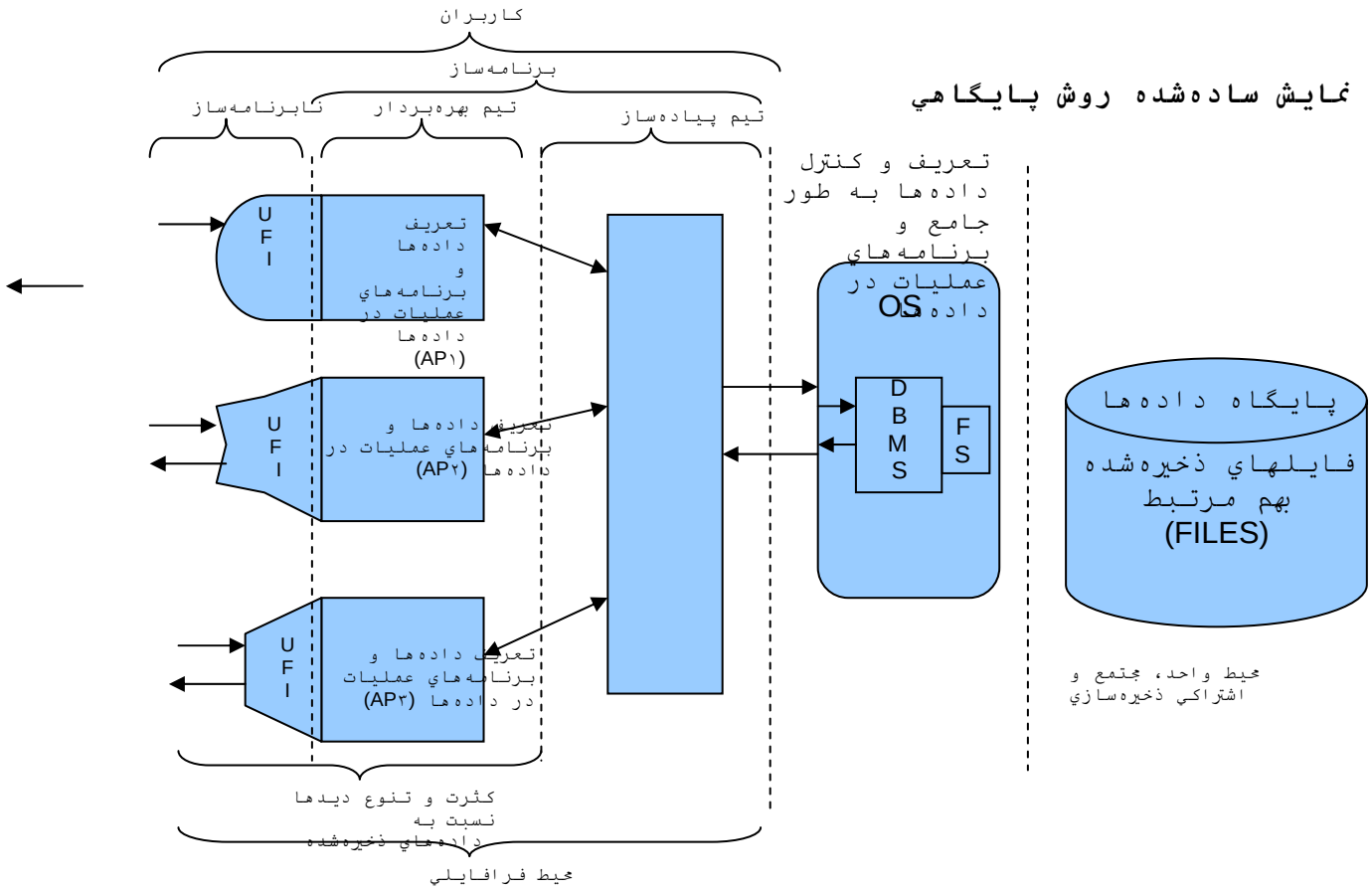
در روش فایلینگ یک سیستم مدیریتی قدرتمند، متمرکز و یکپارچه برای کنترل عملیات وجود ندارد و تنها مرجع کنترلی بخش سیستم فایل از سیستم عامل است. اما سیستم مدیریت پایگاه داده، یک سیستم مدیریت یکپارچه است که کنترل عملیات بر روی کل داده‌های بانک اطلاعات را بعهده دارد. حتی اگر داده‌ها بر روی یک شبکه توزیع شده باشند.

از آنجا که در روش فایلینگ خیلی اوقات امکان استفاده همزمان چندین کاربر از یک فایل وجود ندارد، در بعضی موارد ناگزیریم که یک فایل مشخص را در چندین محل مختلف (برای چند کاربر) ذخیره کنیم. ذخیره یک داده مشخص در چند جای مختلف (بجای یک محل) اصطلاحا افزونگی نامیده می‌شود. پدیده افزونگی غیر از مصرف فضای اضافی بر روی دیسک مشکلات مهمتری نیز دارد. مثلا در صورت نیاز به تغییر تغییر بخشی از داده مجبوریم آن تغییر را در تمام جاهایی که آن داده ذخیره شده اعمال کنیم. اما سیستم DBMS امکان استفاده همزمان کاربران متعدد از داده‌ها را بصورت کاملا کنترل شده فراهم می‌آورد.

برای مثال، یک نرم افزار اتوماسیون دانشگاه را در نظر بگیرید که شامل قسمتهای مختلف، مانند بخش آموزش، امور مالی و امور رفاهی است. هریک از این بخشها در یکی از قسمتهای دانشگاه توسط توسط کاربر جداگانه‌ای استفاده می شود. واضح است که هریک از این بخشها نیاز دارند به اطلاعات دانشجویان دسترسی داشته باشند. بنابراین اگر از روش فایلینگ استفاده شود، یک فایل برای ذخیره اطلاعات دانشجویان در هریک از این بخشها نیاز است. پس هر بخش فایل اطلاعات دانشجویان را بصورت تکراری و نیز تعدادی فایل مربوط به خود دارد که فایلهای بخشهای مختلف ارتباطی با یکدیگر ندارند و تحت کنترل یک مدیریت واحد نیز قرار نمی‌گیرند. اما در روش پایگاهی، اطلاعات دانشجو تنها یکبار ذخیره می‌شود و کل اطلاعات سیستم نیز بصورت یکپارچه و تحت مدیریت کامل DBMS ذخیره می شوند. شکل ۱و۲ ارتباط بین برنامه کاربردی و داده‌ها را در دو نوع سیستم نشان می دهند.

نوع دیگری از افزونگی از آنجا ناشی می‌شود که ساختار فایلها از هم مجزا هستند و نمی‌توان بین آنها ارتباطی برقرار کرد. بنابراین، اگر در ساختار یک فایل نیاز به بخشی از داده های یک فایل دیگر داشته باشیم، گاهی ناگزیر می‌شویم آن داده ها را مجددا در این بخش هم داشته باشیم.


نمایش ساده‌شده روش فایلینگ
برنامه‌نویس برنامه‌ساز
برنامه‌ساز
اداره امور آموزش
U۱
UFD
برنامه‌های ایجاد، کنترل و پردازش فایلها
OS
FS
FILES
محیط ذخیره‌سازی اطلاعات خاص اداره ثبت نام: U۱
اداره امور مالی
U۲
UFD
برنامه‌های ایجاد، کنترل و پردازش فایلها
OS
FS
FILES
محیط ذخیره‌سازی اطلاعات خاص اداره امور مالی: U۲
اداره امور رفاهی دانشجویان
U۳
UFD
برنامه‌های ایجاد، کنترل و پردازش فایلها
OS
FS
FILES
محیط ذخیره‌سازی اطلاعات خاص اداره امور رفاهی: U۳
محیط فایلینگ منطقی و یا مجازی
محیط فیزیکی ذخیره و بازیابی اطلاعات



۳- عدم وجود ضوابط ایمنی کارا و مطمئن

یکی از نگرانی‌های اصلی کاربر هنگامی که داده‌های خود را در یک فایل معمولی ذخیره می‌کند، بحث عدم امنیت داده‌ها و امکان استفاده افراد غیر مجاز از داده‌ها می‌باشد. سیستم‌های مدیریت بانک اطلاعات، از قابلیتهای امنیتی قدرتمندی برخوردار است. امکان تعریف سطوح دسترسی مختلف برای کاربران مختلف بر روی هر بخشی از داده‌ها و استفاده از روش‌های رمزگذاری پیشرفته بر روی اطلاعات موجب می‌شود که تنها راه استفاده از داده‌های ذخیره شده بر روی دیسک از طریق DBMS باشد. در نتیجه امنیت اطلاعات در سطح بسیار بالایی تضمین می‌شود.

همچنین بر خلاف سیستم فایل، سیستمهای مدیریت بانک اطلاعات معمولا از امکانات ویژه‌ای جهت پشتیبان گیری داده‌ها برخوردارند. عملیات پشتیبان گیری خودکار نیز معمولا وجود دارد تا در صورت بروز حادثه‌ای که منجر به از بین رفتن اطلاعات یا خرابی داده‌ها می‌شود، نسخه‌های پشتیبان را در اختیار داشته باشیم. همچنین در صورت خرابی داده‌ها خود DBMS اقدام به ترمیم داده‌ها می‌کند که این کار با استفاده از داده‌های پشتیبان و نیز فایل حاوی گزارش تراکنش‌های صورت گرفته بر روی داده‌ها (فایل log) صورت می‌گیرد.

۴- خطر بروز پدیده ناسازگاری داده‌ها (عدم مدیریت تراکنش)

تراکنش به عملیاتی اطلاق می شود که می خواهد روی داده های بانک اطلاعات پردازشی انجام دهد. یکی از وظایف مهم DBMS مدیریت تراکنشها است بنحویکه چهار ویژگی اصلی زیر همواره برای تراکنشها حفظ شود:
- یکپارچگی (Atomicity): هر تراکنش می تواند خود از یکسری عملیات جزئی‌تر تشکیل شده باشد. گاهی یک تراکنش در حالیکه تنها چند دستور اول آن اجرا شده، بدلیلی (مانند قطع برق) اجرای ادامه دستوراتش متوقف می‌شود. در روش فایلینگ و حتی در پایگاه داده‌های فاقد DBMS در چنین مواقعی داده ها دچار ناهمخوانی می شوند و عملا داده‌ها نامعتبر می‌گردند. برای مثال فرض کنیم در یک نرم افزار بانک دستوری را اجرا کرده‌ایم که قرار است ۱۰۰۰۰ تومان از حسابی به حساب دیگر منتقل کند. در اینجا تراکنشی داریم که شامل ۲ عملیات اصلی است. اول کسر ۱۰۰۰۰ تومان از حساب اول و سپس افزودن آن به حساب دوم. حال اگر بعد از اجرای دستور اول و قبل از اینکه دستور دوم اجرا شود برق قطع شود، چه اتفاقی می‌افتد؟ روشن است که اطلاعات ما دچار اشکال می شوند. اما DBMS با مکانیزمهای مطمئن خود تضمین می کند که یک تراکنش یا بطور جامع اجرا شود و یا اینکه اصلا اجرا نشود (در واقع تغییرات اعمال شده طوری خنثی گردد که گویی تراکنش اصلا اجرا نشده است).
- همخوانی (Consistency): اجرای صحیح یک تراکنش باید سیستم را همواره از یک حالت صحیح به حالت صحیح دیگری ببرد.
- انزوا (Isolation): اجرای همروند چند تراکنش نباید تاثیری روی یکدیگر داشته باشد. در واقع اجرای دو تراکنش بطور همروند همان نتایجی را باید داشته باشد که اجرای پشت سرهم آنها خواهد داشت.
- مانایی (Durability): نتایج اجرای یک تراکنش بر روی داده‌ها نباید موقتی باشد و بایستی پایدار بماند.

۵- عدم امکان اشتراکی شدن داده‌ها

سیستم فایل، در صورتیکه یک کاربر (یا برنامه) در حال نوشتن در یک فایل باشد، هیچ برنامه دیگری نمی‌تواند به فایل دسترسی داشته باشد و باصطلاح تا پایان کار آن برنامه فایل قفل می‌شود. اما همانطور که قبلا هم اشاره شد، سیستم DBMS امکان استفاده همزمان کاربران متعدد از داده‌ها را بصورت کاملا کنترل شده فراهم می‌آورد. در واقع DBMS از روش‌های قفل‌گذاری پیشرفته‌تری استفاده می‌کند. بعنوان مثال، از چندین نوع قفل مختلف استفاده میکند که این قفل‌ها حالت سلسله مراتبی دارند. خیلی اوقات، DBMS بجای قفل‌گذاری بر روی کل فایل، بخش کوچکی از داده‌ها را که در حال تغییر است، قفل می‌کند.

۶- حجم زیاد برنامه‌سازی

در روش فایلینگ برای ساده ترین عملیات هم نیاز به کدنویسی نسبتا زیادی داریم. بعنوان مثال، برای تغییر نام یک دانشجو در فایل اطلاعات تعدادی دانشجو علاوه ب دستورات اولیه مربوط به باز کردن فایل و بردن مکان‌نما به ابتدای آن باید در یک حلقه رکوردهای فایل را یکی یکی بخوانیم تا به رکورد مورد نظر برسیم. سپس آنرا به حافظه بیاوریم، قسمت نام آن را تغییر دهیم و مجددا به محل مربوطه در فایل رفته و آنرا در فایل بنویسیم. می‌بینیم که همین عملیات ساده به نوشتن چندین خط برنامه نیاز دارد.

اما در سیستمهای پایگاه داده با توجه به اینکه از زبانهای نسل جدیدتری برخوردارند که بسیار نزدیک به زبان طبیعی است، تنها با یک دستور شبیه به زبان طبیعی به DBMS می گوییم که نام فلان دانشجو را به این نام جدید تبدیل کن.

۷- وابستگی برنامه‌های کاربردی به محیط ذخیره‌سازی داده‌ها

قبل از توضیح این مطلب و بحث در مورد استقلال داده‌ای در پایگاه داده، معماری سیستم پایگاه داده‌ها را معرفی می‌کنیم.

معماری پشنهادی ANSI برای پایگاه داده‌ها یک معماری سه لایه است. این سطوح در حقیقت به دیدهای مختلفی مربوط می‌شود که بر روی پایگاه داده وجود دارد که شامل سه نوع دید داخلی، ادراکی و خارجی است.

دید خارجی

۱- دید یک کاربر خاص نسبت به داده‌های ذخیره‌شده در پایگاه داده است و بنابراین از کاربری به کاربر دیگر می تواند متفاوت باشد. بعنوان مثال در شکل ۲ دیدیم که سه نوع کاربر مختلف با سیستم اتوماسیون دانشگاه کار می کردند. با توجه به اینکه حیطه کاری هر کاربر متفاوت است، دیدی متفاوت نسبت به داده‌ها دارد. در واقع دید هر کاربر یک دید جزئی روی بخشی از داده های بانک می باشد. به مجموع دید تمام کاربران، سطح خارجی بانک اطلاعات گفته می‌شود.

دید ادراکی (مفهومی)

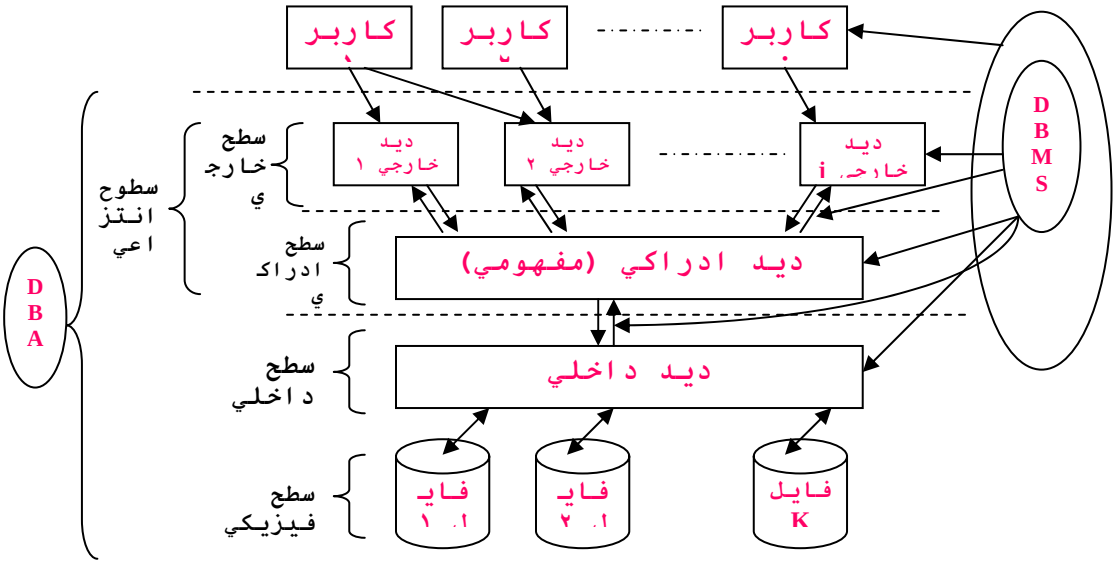
۱- دید طراح پایگاه داده‌ها نسبت به ساختار داده‌های ذخیره‌شده است. بر مبنای این دید است که طراح ساختارهای مورد نیاز برای داده‌ها را تعریف و بانک اطلاعات را بوجود می‌آورد. طراح بانک، دیدی کامل نسبت به ساختار کل بانک اطلاعات دارد و در حقیقت دید طراح جامع دید همه کاربران است. ساختار داده‌ها در این سطح شمای ادراکی نامیده می‌شود. سازنده شمای ادراکی طراح بانک اطلاعات است.

دید داخلی

این دید در سطح فیزیکی و محیط ذخیره سازی مطرح می شود. در این سطح به داده‌ها با دیدی سطح پایین‌تر نگاه می‌کنیم و مسائلی از قبیل شکل داده‌های ذخیره شده بر روی دیسک، نحوه خوشه بندی، نوع شاخص-گذاری، روش رمزگذاری و ... در این سطح مطرح است. ساختار داده‌ها در این سطح شمای فیزیکی نامیده می شود که توسط DBMS ساخته می‌شود.

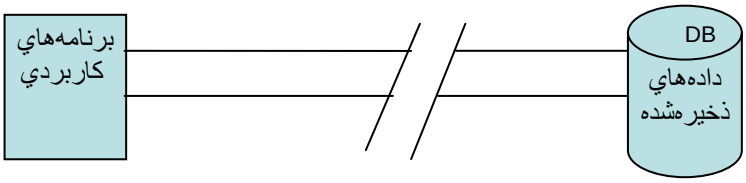
از آنجا که DBMS واسط بین کاربران و محیط فیزیکی است، این دید در درجه اول مخصوص خود DBMS است که وظیفه نگاشت داده‌ها از یک سطح به سطح دیگر را بعهده دارد. طراح پایگاه داده نیز ممکن است چنین دیدی داشته باشد؛ اما معمولا ضرورتی ندارد.

معماري پشنهادي ANSI



استقلال داده‌ای

استقلال داده‌ای یعنی وابسته نبودن برنامه‌های کاربردی به داده‌های ذخیره‌شده. برای ذخیره داده‌ها در فایل، ابتدا ساختاری در خود برنامه تعریف می شود و سپس فایلی با این ساختار ساخته می‌شود. در این حالت ساختار داده‌ها از محیط برنامه مجزا نیست و هر تغییری در ساختار داده ناگزیر در داخل خود برنامه بایستی صورت گیرد. اما در سیستم بانک اطلاعات داده از برنامه جداست. ساختار داده‌ها (همان شمای ادراکی) در محیط بانک اطلاعات تعریف می‌شود. سپس از طریق برنامه با ساختارهای تعریف شده ارتباط برقرار می‌گردد.



انواع استقلال داده‌ای

- استقلال داده‌ای فیزیکی

عبارتست از مصونیت دیدهای کاربران و برنامه‌های کاربردی در قبال تغییرات در سطح داخلی-فیزیکی پایگاه داده‌ها (شمای فیزیکی). بعنوان مثال، اگر نوع رسانه ذخیره سازی و یا محل ذخیره سازی عوض شود، این تغییر نباید تاثیری در دید کاربر برنامه (دید خارجی) داشته باشد. از آنجا که در سیستمهای بانک اطلاعات کاربران مستقیما با شطح فیزیکی ارتباطی ندارند، این نوع استقلال داده‌ای کاملا وجود دارد.

- استقلال داده‌ای منطقی

عبارتست از مصونیت دیدهای کاربران و برنامه‌های کاربردی در قبال تغییرات در شمای ادراکی پایگاه داده‌ها. یعنی اگر طراحی منطقی پایگاه داده‌ها شمای ادراکی تغییری ایجاد شود (که البته منجر به تغییر در شمای فیزیکی نیز خواهد شد) نباید تاثیری در دیدهای خارجی داشته باشد.

نقطه ضعف سیستم پایگاه داده نسبت به سیستم پرونده ای:

ایجاد هر مزیت و خدماتی مستلزم هزینه است. دستیابی به مزایایی که برای سیستم پایگاه داده و سیستم مدیریت پایگاه داده نام برده شد، مسلما بدون هزینه میسر نیست. بدلیل ساختار سه لایه ای پایگاه داده (نگاشتهایی که برای انجام هر عملیات بایستی بین لایه ها انجام گردد) و همچنین با توجه به کنترلهایی که سیستم مدیریت پایگاه داده برای تضمین جامعیت داده ها و تراکنشها، امنیت و ... انجام می دهد، عملیات کاربر شامل سربارهای زمانی می شود و لذا برنامه های کاربردی ممکن است کندتر شوند. اما در کنار تمامی مزایای ارزشمند ذکرشده برای سیستم پایگاه داده این نقطه ضعف که البته محسوس و آزاردهنده نیست را می توان تحمل کرد.

زبانهای کار با بانک اطلاعات

برای برنامه نویسی بانک اطلاعات دو رده از زبانها مورد استفاده قرار می‌گیرند:

۱) زبان میزبان (HL)

یکی از زبانهای برنامه‌سازی متداول مانند کوبول، PL۱، فرترن، پاسکال، C، C# و زبانهایی مثل ADA، LISP، JAVA و نیز زبان اسمبلی است. پس زبان میزبان را از قبل می‌شناسیم و چیز جدید و نامتعارفی نیست. اما تنها چیزی که ممکن است سوال برانگیز باشد این است که به چه دلیل در این مبحث به آنها زبان میزبان می‌گوییم؟! جواب این سوال بعد از معرفی نوع دوم زبانهای کار با بانک اطلاعات مشخص خواهد شد.

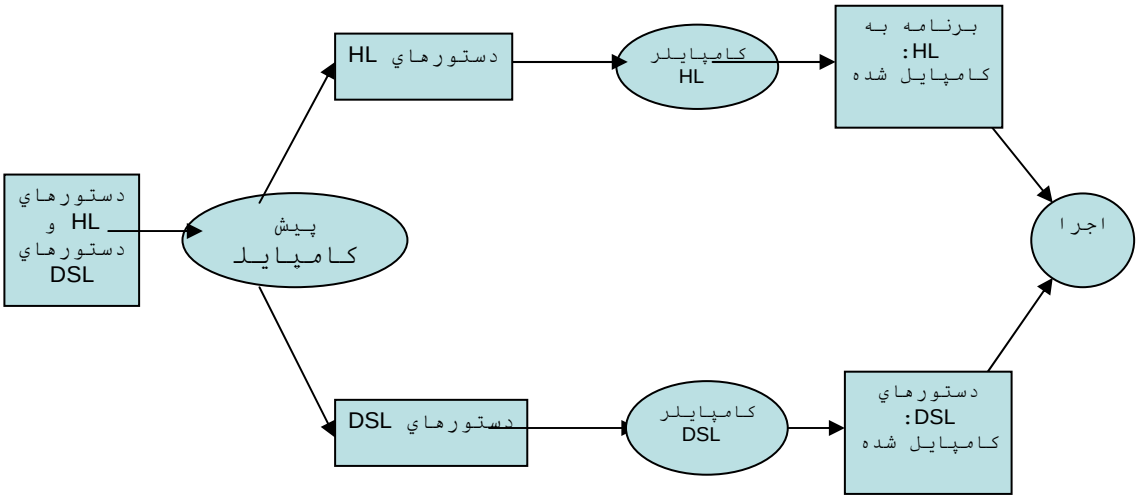
همانطور که می‌دانیم، زبان‌های میزبان زبان‌های خاص منظوره نیستند، بلکه معمولا زبانهای چند منظوره هستند. مثلا زبان C در زمینه‌های مختلف از جمله برنامه نویسی سیستمی، برنامه نویسی گرافیکی، برنامه نویسی بانک اطلاعات و ... کاربرد دارد.

زبان‌های میزبان در دسته بندی زبانهای برنامه نویسی از دسته زبان‌های رویه‌ای می‌باشند. دلیل این نامگذاری این است که برای هر عملیاتی که در این زبانها می خواهیم انجام دهیم، باید رویه انجام آنرا با جزئیات کامل ذکر کنیم.

۲) زبان داده‌ای فرعی (DSL)

زبان‌هایی هستند که بر خلاف زبان‌های میزبان تک منظوره هستند؛ یعنی فقط شامل دستوراتی برای کار با پایگاه داده می‌باشند و اساسا به همین منظور ساخته شده‌اند.
این زبانها نسل بعدی زبانهای رویه‌ای هستند که به زبان‌های توصیفی معروفند. این نوع زبان‌ها خیلی به زبان طبیعی نزدیک هستند و هنگامی که می-خواهیم عملیاتی را اجرا کنیم بر خلاف زبان‌های رویه‌ای، فقط انجام آن عملیات را درخواست می‌کنیم و به روش و جزئیات مراحل انجام آن کاری نداریم. بعنوان مثال، برای بازیابی لیست دانشجویان که در بانک اطلاعات ذخیره شده، بجای نوشتن یک دستور حلقه که یکی یکی اطلاعات دانشجویان را خوانده و نمایش دهد، با یک دستور شبیه به زبان طبیعی (و البته با ساختار خاص آن زبان) می-گوییم لیست دانشجویان را بازیابی کن.
زبان‌های DSL بر حسب نوع دستوراتشان سه نوع مختلف می‌توانند باشند:
۱- زبان تعریف داده‌ها (DDL): این زبان‌ها شامل دستورات لازم جهت تعریف و کار با ساختارهای داده‌ای می-باشند.
۲- زبان عملیات روی داده‌ها (DML): این زبان‌ها شامل دستوراتی هستند که برای کار با داده‌های بانک اطلاعات استفاده می‌شوند و به ساختار بانک اطلاعات کاری ندارند. عملیات اصلی DML عبارتند از: ورود داده جدید، ویرایش داده‌های قبلا وارد شده، حذف داده‌ها و بازیابی داده‌ها.
۳- زبان کنترل داده‌ها (DCL): این زبان‌ها شامل دستورات کنترلی هستند که برای کنترل صحت داده‌ها، جامعیت عملیات روی داده‌ها و ... استفاده می‌شوند.
البته برخی از زبان‌های DSL نیز هستند که هر سه نوع امکانات را شامل می‌شوند. SQL معروفترین و متداولترین زبان DSL است که شامل هر سه نوع این دستورات می‌باشد.
حال باید ببینیم که آیا زبانهای DSL نیز مانند زبانهای میزبان محیط برنامه نویسی خاص خودشان را دارند یا نه. بر این اساس، زبانهای DSL به دو دسته تقسیم می‌شوند:
۱- توکار (E.DSL): دستورات این ربانها بطور مستقل قابل اجرا نیست؛ بلکه باید برنامه ای به یک زبان میزبان نوشته شده باشد و در متن ن برنامه در جایی که نیاز هست، دستوری از زبان DSL (بصورت توکار) قرار داده شده باشد. مثل زبان Btrieve در C یا SQL در دلفی.
با این تعریف، دلیل نامگذاری زبانهای میزبان به این نام هم مشخص شد. در واقع این زبانها میزبان زبانهای DSL هستند.
۲- مستقل (I.DSL): این زبانها به زبان میزبان نیاز ندارد و به صورت مستقل استفاده می‌شوند. مثل زبان Foxpro که دارای یک محیط برنامه نویسی مستقل است و در آن می توان برنامه‌های کاربردی بانک اطلاعات را بدون نیاز به زبان دیگری بوجود آورد.
البته برخی از زبانهای DSL هم هستند که می توانند به هردو صورت مستقل و توکار بکار روند.
زمانی که در یک برنامه زبان میزبان دستورات زبانهای DSL بصورت توکار استفاده شده، عمل کامپایل دستورات اصلی زبان میزبان و دستورات زبان DSL بصورت جداگانه توسط کامپایلرهای خاص هریک از این دو

زبان انجام می‌شود. اما در نهایت نتایج این دو عملیات کامپایل با هم تلفیق می‌شوند تا برنامه بطور یکپارچه اجرا شود. مراحل کامپایل این نوع برنامه‌ها در شکل *** نشان داده شده‌است.



در شکل ** قسمتی از یک برنامه نوشته شده به زبان C# آورده شده است. کار این قطعه کد این است که اطلاعات ذخیره شده تمام دانشجویان را از بانک اطلاعات بازیابی کند. در خط سوم، یک دستور SQL در بطن دستور C# و در قالب یک رشته نوشته شده است. ساختار دستورات SQL در بخش ۵ بطور کامل مورد بحث قرار خواهد گرفت. در این مثال زبان میزبان C# و زبان DSL زبان SQL است که بصورت توکار استفاده شده است. برای تست کردن این برنامه‌ها از لحظ نحوی، کامپایل کردن صرف کافی نیست. با توجه به اینکه دستور SQL بصورت رشته در یک زبان میزبان استفاده می‌شود، در کامپایلر زبان میزبان تنها دستورات خود زبان بررسی می‌شوند. اما با اجرای برنامه، دستور SQL به DBMS مربوطه ارسال می‌شود تا در آنجا با استفاده ازکامپایلر خودش کامپایل و در صورت صحت، اجرا گردد.

```
Conn.Open()  ;
```

```
DataTable  tbl  = new DataTable()  ;
SqlDataAdapter   da  =  new SqlDataAdapter("Select * From
Students",conn) ;
Da.Fill(tbl);
Conn.Close();
```

اجزای DBMS

DBMS از قسمتهای مختلفی تشکیل می‌شود که هر بخش وظیفه خاصی را انجام می‌دهد. اصلی‌ترین اجزای تشکیل دهنده DBMS عبارتند از:

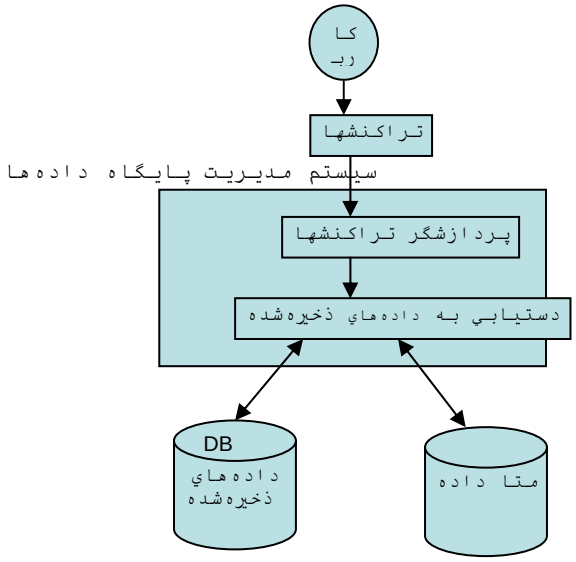
- **DML Processor**: این واحد دستورات DML (جهت تغییر در داده‌ها) را که از سوی کاربران و برنامه ها ارسال می‌شود، پردازش و کامپایل کرده و در صورت مجاز بودن و صحت دستورات آن را برای اجرا مهیا می‌کند.
- **DDL Compiler**: مشابه واحد قبلی است، اما برای زمانی که یک دستور DDL (جهت تعریف یک ساختار داده جدید در شمای ادراکی) از سوی کاربر ارسال می‌شود و کامپایل دستور بر عهده این واحد است.
- **File Manager**: این واحد با سطح فیزیکی بانک اطلاعات ارتباط دارد و وظیفه مدیریت فایلهایی را بعهده دارد که در سطح فیزیکی ذخیره می‌شوند. همچنین یکی از ارکان اصلی در نگاشت داده‌ها بین شمای ادراکی و داخلی است.
- **Database Manager**: این قسمت بیشتر با لایه ادراکی بانک اطلاعات مرتبط است. در واقع نمایش اطلاعات ذخیره شده در قالب شمای ادراکی (در نگاشت بین شمای داخلی و ادراکی) و نمایش منطقی داده‌ها و همچنین عملیات کنترلی مختلف روی بانک اطلاعات، مدیریت عملیات پشتیبان گیری و ... را بعده دارد.
- **Query Processor**: پردازنده پرس و جوهای کاربران است. بعبارت دیگر تنها پردازش دستوراتی را بعهده دارد که هیچ تغییری در ساختار بانک یا داده‌ها بوجود نمی‌آورند، بلکه صرفا برای بازیابی اطلاعات و گزارشگیری بکار می‌روند.
- **Catalog Manager**: وظیفه مدیریت و بروز رسانی اطلاعات موجود در بخش مهمی از پایگاه داده ها بنام کاتالوگ سیستم را بعهده دارد.

کاتالوگ سیستم شامل اطلاعات جامع در مورد سیستم پایگاه داده ها از قبیل حق دسترسی کاربران مختلف، مشخصات کاربران، تاریخ ایجاد و تغییر داده ها، سایز جداول و ... است. دیکشنری داده‌ها نیز جزئی از کاتالوگ سیستم است. دیکشنری داده ها شامل تمام نامهایی است که برای اشیاء مختلف در سیستم انتخاب می شود. به کاتالوگ سیستم اصطلاحا فراداده (متادیتا) گفته می شود؛ زیرا کاتالوگ سیستم شامل

خود داده‌های اصلی ما نیست، بلکه شامل اطلاعاتی است در مورد داده‌های اصلی (فراداده یعنی داده در مورد داده)

هر تغییری که در ساختار و کلیات پایگاه داده ایجاد می‌کنیم (مثلا هنگام تعریف یک کاربر جدید)، باید این تغییر در کاتالوگ سیستم ثبت شود. این وظیفه بعهده catalog manager می باشد.

نمای بیرونی (ساده‌شده) DBMS



با توجه به قابلیتهایی که در این بخش برای DBMS ها برشمردیم، باید این نکته را نیز خاطرنشان سازیم که همه سیستم های پایگاه داده معروفی که تامنون تولید شده‌اند، از یک سیستم مدیریتی قوی برخوردار نیستند؛ بلمه صرفا دارای یک رابط کاربر برای اجرای دستورات او بر روی داده‌ها می‌باشند. اما وظایف مهمی از قبیل کنترل جامعیت اجرای تراکنشها یا ایجاد امکان استفاده همزمان کاربران زیاد از داده‌ها را انجام نمی‌دهند. از معروفترین DBMS ها می‌توان از MS SQL Server، MySQL و Oracle نام برد. اما پایگاه داده‌های معروفی مانند Access و Foxpro در واقع DBMS نیستند و فقط یک سیستم بانک اطلاعاتی ساده تلقی می‌شوند.