موسسه بابان

انتشارات بابان و انتشارات راهیان ارشد

درس و کنکور ارشد

پایگاه داده‌ها ۱

(مفاهیم اولیه)

ویژه‌ی داوطلبان کنکور کارشناسی ارشد مهندسی کامپیوتر و IT

براساس کتب مرجع

آبراهام سیلبرشاتز، راما کریشنان و رامز المصری

ارسطو خلیلی‌فر

۱ مفاهیم اولیه

مقدمه

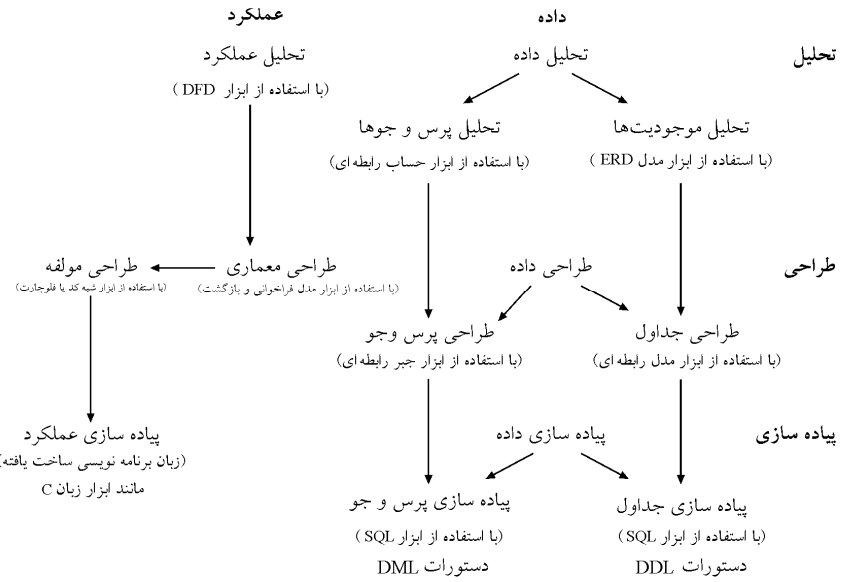
یک محصول نرم‌افزاری به وا سطه‌ی فرآیند تولید نرم‌افزار که شامل فعالیت‌های مدل تحلیل، مدل طراحی، پیاده‌سازی و تست می‌باشد، ایجاد می‌گردد.

به طور کلی هر محصول نرم‌افزاری از دو وجه اساسی زیر تشکیل شده است:

۱- **ساختار داده:** محل نگهداری داده‌های محیط عملیاتی به شکل جداول.

۲- **عملکرد:** شامل برنامه‌ها و دستورالعمل‌ها یا همان برنامه کاربردی

مراحل ایجاد یک برنامه کاربردی به همراه بانک اطلاعات آن مطابق روال زیر می‌باشد:



توجه: از آنجا که ما قصد داریم در این کتاب مفاهیم مربوط به پایگاه داده را تشریح نماییم، بنابراین از بیان مطلب مربوط به بخش عملکرد نرم‌افزار صـــرف‌نظر می‌نمائیم. بخش عملکرد را در کتاب مهند سی نرم‌افزار مورد بحث و برر سی قرار داده‌ایم. پس در ادامه به طور مف صل به مفاهیم مربوط به پایگاه داده می‌پردازیم. در ادامه با ذکر یک مثال، به طور اجمالی مراحل ایجاد یک سیستم بانک اطلاعاتی را بیان می‌کنیم.

مثال: بانک اطلاعات و پرس‌وجوهای مربوط به نگهداری اطلاعات سیستم تولیدکنندگان و قطعات را تحلیل، طراحی و در نهایت پیاده‌سازی نمایید.

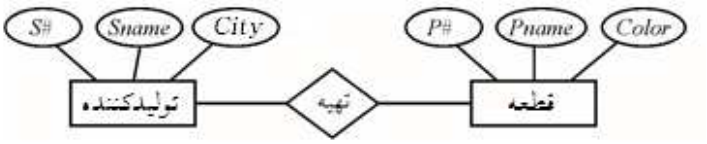
پاسخ:

مدل تحلیل

مدل تحلیل داده شـامل تحلیل موجودیت‌ها و تحلیل پرس و جوها می‌باشـد. تحلیل موجودیت‌ها توسط ابزار مدل ER و تحلیل پرس و جوها توسط ابزار حساب رابطه‌ای مدل می‌شوند.

الف) تحلیل موجودیت ها (توسط مدل ER)

در یک محیط عملیاتی ا سامی عام، موجودیت‌ها می‌با شند. بنابراین موجودیت‌های تهیه‌کننده (S) و قطعه (P) موجود می‌باشند.



توجه: مبحث مدل ER در فصل مدل ER تشریح می‌گردد.

ب) تحلیل پرس و جوها (توسط حساب رابطه ای)

پرس و جو: نام تولیدکنندگان ساکن شهر C2 :

$$\{t \mid \exists\, sx \in s(t[sname] = sx[sname] \text{ and } sx[city] = 'c۲')\}$$

توجه: حساب رابطه‌ای به دو طبقه کلی حساب رابطه‌ای دامنه‌ای و حساب رابطه‌ای تاپلی تقسیم می‌شود. پرس و جوی فوق بر اساس حساب رابطه‌ای تاپلی نوشته شده است.

توجه: مبحث حساب رابطه‌ای(دامنه‌ای و تاپلی) در فصل حساب رابطه‌ای تشریح می گردد.

مدل طراحی

پس از مدل تحلیل داده، نوبت به مدل طراحی داده می‌رســـد. طراحی داده بر دو بخش طراحی جداول و طراحی پرس و جو می‌باشـــد. طراحی جداول از بخش طراحی داده، تحلیل موجودیت (ERD) از مدل تحلیل را به عنوان ورودی دریافت کرده و تو سط مدل رابطه‌ای، طراحی جداول را

انجام می‌دهد. طراحی پرس و جو از بخش طراحی داده، تحلیل پرس و جو(حساب رابطه‌ای) از مدل تحلیل را به عنوان ورودی دریافت کرده و توسط جبر رابطه‌ای، طراحی پرس و جو را انجام می‌دهد.

الف) طراحی جدول (توسط مدل رابطه‌ای)

S#	Sname	City
S1	Sn1	C1
S2	Sn2	C2
S3	Sn3	C2

جدول **S**

(تولیدکنندگان)

S#	P#	QTY
S1	P1	10
S1	P2	20
S2	P1	30

جدول **SP**

(تولید)

P#	Pname	Color
P1	Pn1	Red
P2	Pn2	Blue

جدول **P**

(قطعات)

توجه: مبحث مدل رابطه‌ای در فصل مدل رابطه‌ای تشریح می‌گردد.

ب) طراحی پرس و جو (توسط جبر رابطه‌ای)

پرس و جو: نام تولیدکنندگان ساکن شهر C2 :

$$\Pi_{sname}(\sigma_{city=c2}(s)) \overset{\text{خروجی}}{\Rightarrow} \begin{array}{|c|}\hline \text{Sname} \\ \hline \text{Sn2} \\ \text{Sn3} \\ \hline \end{array}$$

توجه: مبحث جبررابطه‌ای در فصل جبررابطه‌ای تشریح می‌گردد.

توجه: مدل تحلیل، مدل سازی عالم خارج به زبانی شبیه انسان است، اما مدل طراحی مدل سازی عالم داخل ماشین به زبانی شبیه زبان ماشین است. بنابراین برای اینکه ساخت برنامه کامپیوتری که به زبان ماشین است، امکان‌پذیر باشد، باید مدل تحلیل به مدل طراحی نگاشت شود تا مدل طراحی بتواند به عنوان نقشه راهی شبیه به زبان ماشین، راهگشا باشد.

مدل‌های تحلیل، کلی‌تر و انتزاعی‌تر هستند، و برای مدل‌سازی عالم خارج مورد استفاده قرار می‌گیرند، بدون آنکه به جزئیات نحوه پیاده‌سازی بپردازند، در واقع، مدل تحلیل به کلی‌گویی به زبان انسان و حذف جزئیات نحوه پیاده‌سازی می‌پردازد. اما مدل های طراحی، جزئی‌تر و غیرانتزاعی‌تر هستند، و برای مدل سازی عالم داخل ماشین مورد استفاده قرار می‌گیرند، و به بیان جزئیات نحوه پیاده سازی می‌پردازند، در واقع مدل طراحی به جزئی‌گویی به زبان شبیه ماشین و درج جزئیات نحوه پیاده‌سازی می‌پردازد.

با نگاهی سطح به سطح، به حل مساله، سطوح مختلفی از انتزاع را خواهیم داشت. در بالاترین سطح انتزاع، راه حل به صورت کلی به زبان محیط مساله بیان می‌گردد. در سطوح پایین‌تر، راه حل به جزئیات پیاده‌سازی نزدیک‌تر می‌شود و در پایین‌ترین سطح انتزاع راه حل به صورتی بیان می‌شود تا مستقیما قابل پیاده‌سازی باشد.

پیاده‌سازی

پس از مدل طراحی نوبت به پیاده‌سازی می‌رسد. پیاده‌سازی جداول از بخش پیاده‌سازی داده، طراحی جداول از مدل طراحی را به عنوان ورودی دریافت کرده و توسط دستورات DDL در SQL، پیاده‌سازی جداول را انجام می‌دهد. پیاده‌سازی پرس و جو از بخش پیاده‌سازی داده، طراحی پرس و جو از مدل طراحی را به عنوان ورودی دریافت کرده و توسط دستورات DML در SQL پیاده‌سازی پرس و جو را انجام می‌دهد.

الف) پیاده‌سازی جدول (توسط SQL:DDL)

پیاده‌سازی جدول تولیدکنندگان (S)

```
Create Table S
        (
        S# char (5),
        Sname char (20),
        City char (15),
        Primary key (S#)
        )
```

پیاده‌سازی جدول قطعات (P)

```
Create Table P
        (
        P# char (5),
        Pname char (20),
        Color char (10),
        Primary key (S#)
        )
```

پیاده‌سازی جدول تولید (SP)

```
Create Table SP
        (
        S# char (5),
        P# char (5),
        QTY numeric (10),
```

```
Primary key (S#, P#),
Foreign key (S#) References S.(S#)
  on delete cascade
  on update cascade,
Foreign key (P#) Reference P.(P#)
  on delete cascade
  on update cascade,
Check (QTY>1 AND QTY<1000)
)
```

توجه: مبحث SQL:DDL در فصل SQL:DDL تشریح می‌گردد.

ب) پیاده‌سازی پرس و جو (توسط SQL : DML)

پرس و جو: نام تولید کنندگان ساکن شهر C2 :

```
Select  Sname                خروجی      Sname
From S                       ⇒           Sn2
Where  City = 'C2'                       Sn3
```

توجه: مبحث SQL:DML در فصل SQL:DML تشریح می گردد.

حال که به طور اجمالی با مراحل ایجاد یک سیستم بانک اطلاعاتی آشنا شدید، در ادامه به تشریح دقیق‌تر مفاهیم پایگاه داده می‌پردازیم.

تعریف افزونگی

افزونگی در معنای عام به معنی تکرار ذخیره‌سازی داده‌ها می‌باشد. به عبارت بهتر تکرار مقادیر یک یا چند صفت خاصه در نمونه‌های مختلف یک نوع رکورد از یک فایل، به بیانی دیگر ذخیره‌سازی آن مقادیر در بیش از یک نقطه از فایل، نوع دیگری از تکرار اطلاعات در فایل های مختلف می‌باشد.

توجه: افزونگی سبب هدر رفتن حافظه می‌گردد.

انواع افزونگی

افزونگی بر دو نوع است:

۱- افزونگی طبیعی(محتوایی)

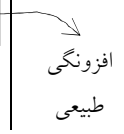
در افزونگی طبیعی، یک مقدار مشخص از صفت خاصه در تعدادی از نمونه رکوردها وجود دارد که بر دو نوع زیر است:

الف) تک فایلی(در یک فایل داده‌ای رخ می‌دهد.)

ب) چند فایلی (در چند فایل داده‌ای رخ می‌دهد.)

مثال: جدول studclg با مقادیر زیر را در نظر بگیرید:

S#	Sname	Clg#	Clgname
8621	Ali	12	Computer
8442	Reza	12	Computer
8731	Abbas	NULL	NULL



در جدول studclg این اطلاعات که «کد دانشکده‌ی کامپیوتر برابر ۱۲ است.» در سطرهای اول و دوم تکرار شده است، پس افزونگی داده (طبیعی) وجود دارد. اگر شخصی بخواهد که دانشکده‌ی کامپیوتر را به ۲۲ تغییر دهد و به اشتباه، فقط مقدار #Clg در سطر اول جدول studclg را عوض کند، از این به بعد دانشکده کامپیوتر دو کد خواهد داشت: ۱۲ و ۲۲. این موضوع یعنی **بی‌نظمی یا آنومالی یا ناهنجاری**.

در سطر آخر از این جدول، مقادیر NULL وجود دارد. به طور کلی، به ازای هر سطر از جدول studclg که #Clg آن برابر NULL باشد یک سطر در جدول studclg خواهیم داشت که مقادیر دو ستون آخر آن NULL خواهد بود. پس مشکل مقادیر NULL نیز وجود دارد.

بنابراین اگر تمامی اطلاعات جداول را در یک جدول بریزیم، مشکلات زیر را خواهیم داشت: مانند جدول studclg.

۱- افزونگی داده‌ها (Data Redundancy)

۲- آنومالی یا بی‌نظمی یا ناهنجاری (Anomaly)

۳- مقادیر تهی (NULL Values)

حال اگر جدول studclg را به دو جدول clg و stud به شکل زیر تجزیه و نرمال کنیم:

S#	Sname	Clg#
8621	Ali	12
8442	Reza	12
8731	Abbas	NULL

جدول Stud

Clg#	Clgname
12	Computer

جدول clg

مشکلات فوق مرتفع و نتایج زیر حاصل خواهد شد.

- کاهش افزونگی طبیعی (محتوایی) مانند تکرار بی‌دلیل سطر(12,Computer)
- افزایش افزونگی تکنیکی به دلیل تعریف کلید خارجی (در قالب ستون های مشترک)

توجه: مبحث نرمال‌سازی در فصل نرمال‌سازی تشریح می‌گردد.

افزونگی تکنیکی

تکرار بعضی از مقادیر یک یا چند صفت خاصه در محیط ذخیره‌سازی جهت ایجاد یک شیوه دستیابی کاراتر برای جداول را افزونگی تکنیکی می‌گویند. مثل کلید خارجی برای ارتباط بین جداول یا وقتی که روی صفت خا صه‌ای از یک جدول، شاخص ایجاد می‌کنیم، مقادیر آن صفت در جدول شاخص تکرار خواهد شد.

توجه: مبحث کلید خارجی در فصل مدل رابطه‌ای تشریح می‌گردد.

توجه: مبحث شاخص در فصل SQL:DDL تشریح می‌گردد.

به منظور ایجاد یک سیستم نگهداری اطلاعات در یک محیط عملیاتی دو شیوه رایج می‌باشد:

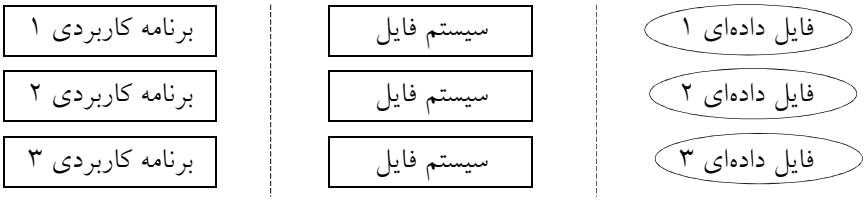
۱- سیستم فایلینگ (پرونده‌ای)

۲- سیستم بانکی (پایگاه داده)

۱- سیستم فایلینگ (پرونده‌ای)

در این روش با بهره‌گیری از امکانات سیستم فایل که در اختیار سیستم عامل می‌باشد و یک زبان برنامه‌نویسی سطح بالا و با نگرش به ورودی‌ها و خروجی‌های موردنظر، برنامه‌های کاربردی به منظور کنترل و پردازش فایل‌های داده‌ای، تحلیل، طراحی، و پیاده‌سازی می‌شوند.

همچنین برای هر فایل داده‌ای رکورد موردنظر با فیلدهای موردنیاز تعریف می‌شود:



سیستم حساب

آدرس	موجودی	نام و نام‌خانوادگی	شماره حساب	شماره مشتری
تهران	۴۰۰	علوی	۰۳۱۳	۶۰۳۷
تهران	۵۰۰	علوی	۱۳۱۴	۶۰۳۷

سیستم اعتباری

آدرس	بدهی	مقدار وام	نام و نام‌خانوادگی	شمار حساب	شماره مشتری
تهران	۴۰۰	۵۰۰	علوی	۰۳۱۳	۶۰۳۷

به طور کلی بسیاری از داده‌های مورد نیاز یک برنامه کاربردی، همان داده‌های مورد نیاز برنامه‌های دیگر است، در روش فایلینگ هر برنامه کاربردی فایل داده‌ای ویژه خود را دارد، بنابراین داده‌های مشترک بین برنامه‌های مختلف باید مجدداً ذخیره شوند که سبب تکرار اطلاعات (افزونگی چند فایلی) و موجب هدر رفتن رسانه ذخیره‌سازی می‌گردد. در سیستم‌های فایلینگ به علت عدم **تجمع**

داده‌های ذخیره شـده (عدم اشـتراک داده‌ها برای همه برنامه‌ها)، عدم وحدت ذخیره‌سـازی و عدم نرمال‌سازی، پدیده افزونگی تک فایلی و چند فایلی را شاهد خواهیم بود.

مثال تک فایلی: مانند تکرار نام و نام‌خانوادگی

مثال چند فایلی: مانند آدرس یک شخص که در فایل داده‌ای هر برنامه کاربردی باید تکرار گردد.

آنومالی داده‌ای یا ناهنجاری داده‌ای

مقدمه ناهنجاری داده‌ای افزونگی اسـت. آنومالی داده‌ای یا بی‌نظمی داده‌ای یا ناهنجاری داده‌ای زمانی بروز می‌کند که بنابر دلایلی یک فقره اطلاع در بیش از یک نقطه ذخیره گردد و لازم باشـد بروزر سانی شود. اگر عمل بروزر سانی در تمام نقاطی که آن فقره اطلاع وجود دارد، انجام نشود، ناهمگونی در اطلاعات و به عبارت دیگر پدیده ناهنجاری داده‌ای رخ داده‌است. به عنوان مثال برای حالت تک فایلی، اگر در فایل داده‌ای سیستم حساب، نام و نام‌خانوادگی «علوی» در سطر اول تغییر کند و فراموش گردد تا در سطر دوم نیز برورسانی گردد، آنگاه شماره مشتری «۶۰۳۷» دارای دو نام و نام‌خانوادگی متفاوت می‌باشد، که ناهنجاری داده‌ای رخ داده‌است. به عنوان مثال برای حالت چند فایلی، زمانی که آدرس «علوی» در فایل داده‌ای سیستم حساب تغییر کند و آدرس «علوی» در فایل داده‌ای سیستم اعتباری بروزرسانی نگردد، ناهنجاری داده‌ای رخ داده‌است.

توجه: در سـیسـتم‌های فایلینگ کنترل پدیده ناهنجاری داده‌ای به دلیل عدم مجتمع بودن داده‌ها و عدم نرمال سازی بسیار مشکل و به صورت دستی است که خطای انسانی را به همراه خواهد داشت.

توجه: در سیستم‌های فایلینگ برقراری امنیت منطقی داده‌ها در برابر خطراتی از قبیل آتش‌سوزی و دستیابی غیرمجاز به دلیل عدم وجود یک مکانیزم پشتیبانی و حفاظتی و در نتیجه دسترسی ساده به داده‌ها بسیار مشکل است.

توجه: در سـیسـتم‌های فایلینگ به علت جدا بودن فایل داده‌ای برنامه‌ها از یکدیگر امنیت فیزیکی داده‌ها در سـطح مناسـبی برقرار اسـت. زیرا داده‌های همه برنامه‌های کاربردی به دلیل عدم مجتمع بودن فایل‌های داده‌ای در یک دیسک به یکباره در معرض تهدید قرار نمی‌گیرند.

توجه: در سـیسـتم‌های فایلینگ هر برنامه کاربردی برای ایجاد و پردازش فایل داده‌ای مختص به خود نوشـته شـده‌اسـت، بنابراین هرگونه تغییری در سـاختار فایل داده‌ای منجر به تغییر در برنامه کاربردی می‌گردد. به بیان دیگر برنامه‌های کاربردی به جنبه و خصـوصـیات محیط فیزیکی ذخیره‌سـازی داده‌ها وابسـته اسـت. زیرا کلیه تعاریف ایجاد فایل‌های داده‌ای و کار با آن‌ها، درون برنامه‌های کاربردی قرار دارد. که این موضوع به معنی عدم استقلال داده‌ای نیز می‌باشد.

توجه: با توجه به ماهیت سـیسـتم‌های فایلینگ و محلی بودن برنامه‌ها و اسـتفاده از الگوی صـف

جهت پردازش درخواست‌ها، دسترسی همزمان و اشتراکی به داده‌ها معنا و مفهومی ندارد.

توجه: در سیستم‌های فایلینگ، به دلیل عدم وجود سطوح دسترسی و مسائل امنیتی زمان اجرای برنامه‌های کاربردی کاهش می‌یابد، به دلیل عدم عبور از گیت امنیتی. اما زمان پاسخ به درخواست‌ها به دلیل وجود صف و عدم استفاده اشتراکی از داده‌ها افزایش می‌یابد.

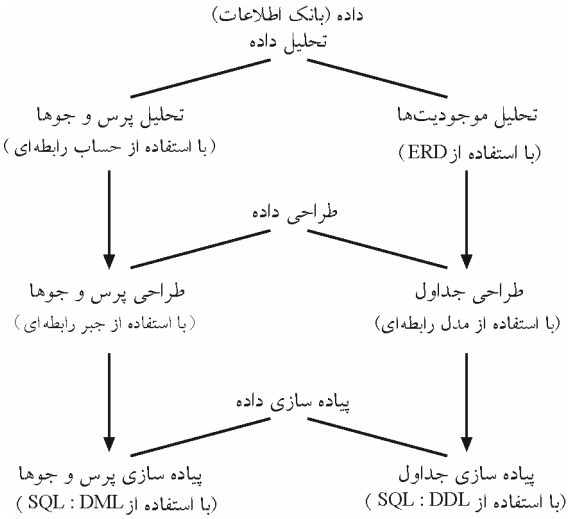
مثال: یک سیستم فایلینگ به صورت زیر است:

```
    type
        phone=record
        name:string [20];
        no:integer;
var
    fp:text;   یا   file of phone;
    i,n: integer;
begin
    assign (fp, 'sample.dat');
    rewrite (fp);
    readln (n);
for i:=1 to n do begin
    readln (no,name)
    writeln (fp,no,name)
end;
reset (fp);
    while TRUE do begin
    readln (fp,no,name);
    write (no,name)
if EOF (fp) then break;
    end;
    close (fp);
    end.
```

توجه: همانطور که در برنامه فوق ملاحظه می‌کنید، تعاریف مربوط به **عملکرد و داده** با هم در یک فایل قرار دارد.

۲- سیستم بانکی (پایگاه داده)

در این روش به منظور ایجاد بانک اطلاعات روال زیر را خواهیم داشت:



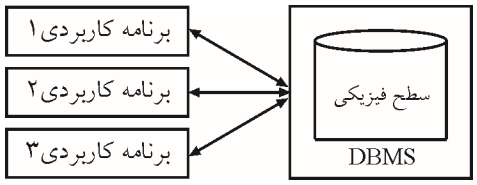
۱- تحلیل
(طراحی ادراکی (مفهومی) یا ادراکی عام)
توسط DA

۲- طراحی
(طراحی منطقی یا ادراکی خاص)
توسط DBA

۳- پیاده‌سازی
توسط DBP

در مرحله بعد یک برنامه کاربردی (عملکرد) مطابق فرآیند تولید نرم‌افزار باید ایجاد گردد. تا کاربران نهایی بتوانند به کمک DBMS در نهایت به داده‌های ویژه خود با توجه به سطوح دسترسی تعریف شده در DBMS برای آنها دسترسی یابند.

توجه: در واقع DBMS رابط بین برنامه‌های کاربردی و داده‌ها ست و هرگونه د ستیابی به داده‌ها از طریق DBMS صورت می‌گیرد. DBMS همچون قلعه‌ای می‌ماند که محل زندگی پادشاه، ملکه و شاهزادگان را درون خود محصور کرده‌است، که هرگونه آمد و شد به محل زندگی پاد شاه باید با عبور از قلعه و اجازه قلعه‌بانان صورت گیرد. پادشاه، ملکه و شاهزادگان همان داده‌ها هستند و قلعه و قلعه‌بانان همان DBMS.



توجه: در سیستم‌های بانکی به دلیل مجتمع بودن داده‌های ذخیره شده (اشتراکی بودن داده‌ها برای برنامه‌ها)، وحدت ذخیره‌سازی و نرمال‌سازی، پدیده افزونگی به معنی آنچه در سیستم‌های فایلینگ دیدید رخ نمی‌دهد. و به تبع ناهنجاری داده‌ای نیز ایجاد نمی‌گردد. مجتمع بودن بدین معنی است که بانک اطلاعات مجموعه‌ای از جداول است که بخشی از ستون‌های اضافی بین آنها حذف شده است. غیر از کلیدهای خارجی که نقش ارتباط بین جداول را بازی می‌کنند که به آن افزونگی تکنیکی گفته می‌شود.

مثال: جداول زیر را در سیستم تولیدکنندگان و قطعات در نظر بگیرید.

S#	Sname	City
S1	Sn1	C1
S2	Sn2	C2
S3	Sn3	C2

جدول **S**

(تولیدکنندگان)

S#	P#	QTY
S1	P1	10
S1	P2	20
S2	P1	30

جدول **SP**

(تولید)

P#	Pname	Color
P1	Pn1	Red
P2	Pn2	Blue

جدول **P**

(قطعات)

برای‌مثال در بانک فوق که از سه جدول تشکیل شده است، لازم نیست که در جدول SP، فیلد City ذخیره شود، چرا که در صورت نیاز به رجوع به جدول S می‌توان آن را به دست آورد.

توجه: در سیستم‌های بانکی برقراری امنیت داده‌ها در برابر خطراتی از قبیل آتش‌سوزی و دستیابی غیرمجاز به دلیل وجود یک سیستم مدیریت بانک اطلاعات (DBMS) در سطح بسیار مناسبی قرار دارد. مانند پشتیبان‌گیری خودکار در یک رسانه ذخیره‌سازی دیگر، تعریف سطوح امنیتی و دسترسی ویژه کاربران و مدیران. برای مثال در SQL Server تا زمانی که در حال اجرا می‌باشد، امکان کپی و سرقت فایل‌های داده‌ای نمی‌باشد، مگر اجرای SQL Server یا همان نگهبان داده‌ها متوقف شود، که این عمل با مجوز مدیران انجام می‌گیرد.

توجه: در سیستم‌های بانکی به علت مجتمع بودن داده‌های ذخیره شده (اشتراکی بودن داده‌ها برای برنامه‌ها) و وحدت ذخیره‌سازی در یک دیسک امنیت فیزیکی داده‌ها در سطح مناسبی برقرار نیست. زیرا داده‌ها به دلیل مجتمع بودن و عدم توزیع‌شدگی در برخی موارد ممکن است به یکباره در معرض تهدید قرار گیرند. البته در اغلب موارد داده‌ها در دیسک‌های مختلف توزیع می‌شوند، که در این حالت مساله امنیت فیزیکی هم وجود نخواهد داشت. و به تبع امنیت فیزیکی در سیستم‌های بانکی نیز در سطح مناسبی برقرار خواهد بود.

توجه: مهمترین خصیصه استفاده از DBMS در سیستم‌های بانکی، مستقل شدن برنامه‌های کاربردی از جنبه و خصوصیات فیزیکی ذخیره‌سازی داده‌ها است. زیرا کلیه تعاریف و ایجاد داده‌ها و جداول در DBMS انجام می‌گیرد و مانند سیستم‌های فایلینگ تعاریف فایل‌های داده‌ای درون برنامه‌های کاربردی قرار ندارد که این موضوع به معنی استقلال داده‌ای نیز می‌باشد.

توجه: با توجه به ماهیت سیستم‌های بانکی و مبتنی بر شبکه بودن، کاربران مختلف می‌توانند به صورت همزمان با بانک کار کنند، یعنی هر کاربر بدون ایجاد محدودیت برای کاربر دیگر در هر لحظه می‌تواند با بانک کار کند.

توجه: با آنکه در روش بانکی وحدت ذخیره‌سازی داریم ولی هر کاربری دید خاص خود را نسبت به داده‌ها دارد، به دلیل تعریف سطوح دسترسی برای کاربران توسط DBMS.

توجه: در سیستم‌های بانکی، به دلیل وجود سطوح دسترسی و مسائل امنیتی زمان اجرای برنامه‌های کاربردی افزایش می‌یابد، به دلیل عبور از گیت امنیتی. اما زمان پاسخ به درخواست‌ها به دلیل عدم وجود صف و استفاده اشتراکی از داده‌ها کاهش می‌یابد.

بخش‌های مختلف سیستم‌های بانکی

محیط بانک اطلاعاتی از چهار بخش زیر تشکیل شده است:

(۱) سخت‌افزار، (۲) نرم‌افزار، (۳) داده و (۴) کاربر

۱- سخت‌افزار

در محیط بانک اطلاعات سه دسته سخت‌افزار وجود دارد:

الف) سخت‌افزار ذخیره سازی اطلاعات: منظور همان رسانه ذخیره سازی خارجی است. همانند دیسک که رسانه اصلی ذخیره‌سازی می‌باشد.

ب) سخت‌افزار پردازنده مرکزی: منظور همان کامپیوتر است.

ج) سخت‌افزار ارتباطی: منظور سخت‌افزارهای مورد نیاز جهت ارتباط بین کامپیوتر و دستگاه‌های جانبی و نیز سایر کامپیوترها می‌باشد.

۲- نرم‌افزار

در محیط بانک اطلاعاتی دو دسته نرم‌افزار وجود دارد:

نرم‌افزار کاربردی: واسط بین کاربر نهایی و سیستم مدیریت پایگاه داده‌ها (DBMS) می‌باشد. این نرم‌افزار به کمک یک زبان سطح بالا و یک زبان داده‌ای (مانند SQL) ایجاد می‌گردد.

نرم‌افزار سیستمی: واسط بین نرم‌افزار کاربردی و بانک اطلاعاتی می‌باشد. نرم‌افزار سیستمی از دو جزء DBMS و سیستم عامل تشکیل شده است. DBMS که نرم‌افزاری پیچیده است میهمان یک سیستم عامل است و از امکانات سیستم عامل در انجام وظایفش استفاده می‌کند. DBMS به برنامه‌ساز امکان می‌دهد تا :

- پایگاه داده خود را تعریف کند (توسط دستورات DDL در SQL)
- پایگاه داده خود را تغییر دهد (توسط دستورات DML در SQL)
- پایگاه داده خود را کنترل کند (توسط دستورات DCL در SQL)

۳- داده

منظور داده‌های مربوط به موجودیت‌های مختلف در یک محیط عملیاتی می‌باشد.

مثال: مانند موجودیت‌های تولیدکننده و قطعه در محیط عملیاتی تولیدکنندگان و قطعات.

۴- کاربر

در محیط بانک اطلاعاتی چهار نوع کاربر وجود دارد:

الف) مدیر داده‌ها (DA : Data Administrator)

انجام مدل تحلیل یا همان طراحی ادراکی(مفهومی) بر عهده DA است، مدیر داده‌ها یک شخصی مدیریتی است و نه یک شخص فنی. DA شخصی است که مسئولیت کنترل اصلی بر روی داده‌ها را بر عهده دارد. وظیفه مدیر داده‌ها آن است که در مرحله اول تصمیم بگیرد که چه داده‌هایی باید در بانک اطلاعاتی ذخیره شود و سپس هنگامی که داده‌ها ذخیره شدند، سیاست‌گذاری‌های لازم را جهت نگهداری و کار با آن‌ها تعیین کند. برای مثال اینکه چه کسی می‌تواند چه اعمالی را بر روی چه داده‌هایی انجام دهد، به معنی تعیین سطوح دسترسی به داده‌ها.

وظایف مدیر داده‌ها (DA)

- انجام تحلیل موجودیت‌ها توسط مدل ER.
- انجام تحلیل پرس و جوها توسط حساب رابطه‌ای.

ب) مدیر بانک اطلاعات (Data Base Administrator : DBA)

انجام مدل طراحی یا همان طراحی منطقی بر عهده DBA است، مدیر بانک اطلاعات فردی است فنی که پشتیبانی‌های تکنیکی لازم را برای پیاده‌سازی تصمیمات مدیر داده‌ها (DA) فراهم می‌سازد. وظیفه DBA ایجاد بانک اطلاعات و پیاده‌سازی کنترل‌های جامعیتی و امنیتی است که سیاست گذاری‌های مدیر داده‌ها (DA) را اعمال کند. همچنین DBA مسئول نظارت بر کارایی و پاسخ به تغییر نیازهاست، بدین معنی که DBA مسئول سازماندهی سیستم برای بدست آوردن بهترین کارایی برای سازمان می‌باشد و همزمان با تغییر نیازهای سازمان، تنظیمات متناسب با نظارت DA اعمال می‌گردد. همچنین DBA مجموعه‌ای از برنامه‌نویسان و سایر افراد فنی را همچون DBPها جهت پیاده‌سازی کارها در اختیار دارد.

وظایف مدیر بانک اطلاعات (DBA)

- انجام طراحی جداول توسط مدل رابطه‌ای.
- انجام طراحی پرس و جوها توسط جبر رابطه‌ای.

ج) برنامه‌نویسان بانک اطلاعات (Data Base Programming : DBP)

انجام فعالیت پیاده سازی بر عهده DBP است، برنامه‌نویسانی هستند که از طریق دستورات DML در SQL با بانک اطلاعات برای انجام عملیات درج، حذف و بروزرسانی در ارتباط هستند. این

افراد دستورات DML را درون یک زبان سطح بالا (زبان میزبان) قرار می‌دهند و برنامه‌های کاربردی را می سازند تا نیازهای کاربران نهایی را برای استفاده از داده‌های درون بانک اطلاعاتی بر طرف سازند.

وظایف برنامه‌نویسان بانک اطلاعات (DBP)

- انجام پیاده‌سازی جداول توسط دستورات DDL در SQL.
- انجام پیاده‌سازی پرس و جوها توسط دستورات DML در SQL.
- انجام پیاده‌سازی کنترل‌های امنیتی توسط دستورات DCL در SQL.

د) کاربر نهایی

فردی است که از برنامه‌های نوشته شده استفاده می‌کند. این افراد با سیستم مدیریت پایگاه داده‌ها از طریق برنامه‌های کاربردی که توسط DBP ایجاد گردیده در تعامل هستند.

توجه: تنها مدیران (DBA, DA) و برنامه‌سازان (DBP) می‌توانند بدون دخالت DBMS به داده‌ها دسترسی داشته باشند و تمام کاربران نهایی فقط از طریق DBMS به داده‌های داخل بانک اطلاعات دسترسی دارند.

معماری بانک اطلاعات

معماری ANSI برای پایگاه داده شامل سه لایه زیر است:

۱- لایه خارجی.

۲- لایه ادراکی شامل زیر لایه‌های مدل تحلیل (طراحی ادراکی یا ادراکی عام) و مدل طراحی (طراحی منطقی یا ادراکی خاص).

۳- لایه داخلی (فیزیکی).

توجه: در معماری بانک اطلاعات، کلمه تصویر، مترادف لایه می‌باشد. برای مثال لایه خارجی همان مفهوم تصویر خارجی را دارد.

یک محصول نرم‌افزاری به واسطه فرآیند تولید نرم‌افزار که شامل فعالیت‌های مدل تحلیل، مدل طراحی، پیاده سازی و تست می‌باشد، ایجاد می‌گردد. کاربران نهایی در لایه خارجی، مدل تحلیل و مدل طراحی در لایه ادراکی و فعالیت پیاده‌سازی در لایه داخلی قرار دارند.

شکل زیر گویای مطلب است:

	تصویر خارجی	کاربر n ... کاربر ۲ کاربر ۱	دیدهای کاربران مختلف (view)
تحلیل	تصویر ادراکی عام (طراحی ادراکی)	مدل ER موجودیت۱ ارتباط موجودیت۲	کل بانک بدون توجه به مدل خاصی
طراحی	تصویر ادراکی خاص (طراحی منطقی)	مدل رابطه‌ای A B C	کل بانک در قالب مدل انتخابی
پیاده‌سازی	تصویر فیزیکی	SQL ...	کل بانک روی رسانه

لایه خارجی

در این لایه، میزان دسترسی کاربران به پایگاه داده، مشخص و مدیریت می گردد. لایه خارجی، تنها لایه‌ای ا ست که به کاربران مربوط می شود. هر کاربر با بخ شی از بانک سر و کار دارد و فقط اجازه د ستیابی به بخ شی از بانک به او داده می شود. برای مثال در بانک اطلاعات مربوط به بانک مرکزی، کاربر نهایی (کارمندان) بخش ارز مسافری، حق دستیابی به بخش‌های دیگر بانک مثل حساب افراد و وام ها را ندارند. قانون «**پنهان سازی اطلاعات**» که می‌گوید «**به هر کس به همان اندازه اطلاعات بده که نیاز دارد و نه بیشتر**» در اینجا صادق است. بنابراین در لایه خارجی دیدهای مختلف کاربران مطرح است و هر کدام بخشی از بانک را می‌بینند.

توجه: لایه خارجی لایه‌ای است که کاربران با آن سر و کار دارند، لایه های دیگر به مدیر و برنامه‌سازان بانک اطلاعات مربوط می‌شود.

لایه ادراکی

لایه ادراکی شامل دو زیر لایه مدل تحلیل(طراحی ادراکی یا ادراکی عام) و مدل طراحی(طراحی منطقی یا ادراکی خاص) می‌باشد.

مدل تحلیل (طراحی ادراکی یا ادراکی عام)

مدل تحلیل داده شـامل تحلیل موجودیت‌ها و تحلیل پرس و جوها می‌باشـد. تحلیل موجودیت‌ها توسط ابزار مدل ER و تحلیل پرس و جوها توسط ابزار حساب رابطه‌ای مدل می‌شوند.

مدل طراحی (طراحی منطقی یا ادراکی خاص)

پس از مدل تحلیل داده، نوبت به مدل طراحی داده می‌رسـد. طراحی داده بر دو بخش طراحی جداول و طراحی پرس و جو می‌باشـد. طراحی جداول از بخش طراحی داده، تحلیل موجودیت (ERD) از مدل تحلیل را به عنوان ورودی دریافت کرده و تو سط مدل رابطه‌ای، طراحی جداول را انجام می‌دهد. طراحی پرس و جو از بخش طراحی داده، تحلیل پرس و جو(حسـاب رابطه‌ای) از مدل تحلیل را به عنوان ورودی دریافت کرده و توسـط جبر رابطه‌ای، طراحی پرس و جو را انجام می‌دهد.

توجه: مدل تحلیل (طراحی ادراکی یا ادراکی عام) به مدل خا صی وابستگی ندارد چون در شروع فعالیت‌ها قرار دارد، اما مدل طراحی (طراحی منطقی یا ادراکی خاص) به مدل خاصی بستگی دارد، بسته به اینکه در مدل تحلیل چه مدلی انتخاب شود، مدل طراحی باید تبعیت کند. اگر مدل تحلیل ساخت یافته بود، مدل طراحی نیز باید ساخت یافتگی را دنبال کند و اگر مدل تحلیل شیءگرا بود، مدل طراحی نیز باید شیءگرایی را دنبال کند.

لایه داخلی(فیزیکی)

پس از مدل طراحی نوبت به پیاده‌سـازی می‌رسـد. پیاده‌سـازی جداول از بخش پیاده‌سـازی داده، طراحی جداول از مدل طراحی را به عنوان ورودی دریافت کرده و توسـط دسـتورات DDL در SQL، پیاده‌سازی جداول را انجام می‌دهد. پیاده‌سازی پرس و جو از بخش پیاده‌سازی داده، طراحی پرس و جو از مدل طراحی را به عنوان ورودی دریافت کرده و توسـط دسـتورات DML در SQL پیاده‌سازی پرس و جو را انجام می‌دهد.

زبان‌های پیاده‌سازی

همانطور که گفتیم یک محصـول نرم‌افزاری از دو وجه عملکرد (برنامه کاربردی) و داده (بانک اطلاعات) تشکیل می‌شود. در ادامه به معرفی انواع زبان‌های برنامه‌سازی می‌پردازیم.

زبان پیاده‌سازی برنامه کاربردی (وجه عملکرد)

برنامه کاربردی نیز مانند بخش داده، حاصل مراحل تحلیل، طراحی و پیاده سازی می‌باشد که شرح این مراحل مربوط به درس مهندسی‌نرم‌افزار می‌باشد. مرحله پیاده‌سازی برنامه کاربردی توسط یکی از زبان‌های برنامه‌نویسی سطح بالا انجام می‌شود.

توجه: به زبان‌های سطح بالا، زبان میزبان یا زبان روالی(Procedural) نیز گفته می‌شود.

زبان پیاده‌سازی بانک اطلاعات (وجه داده)

در بانک اطلاعات از زبان‌های بیانی(Declarative) که به آنها زبان پرس‌وجو (Query Language) نیز گفته می‌شود، استفاده می‌شود. در زبان‌های بیانی کاربر برنامه‌ساز کافیست بگوید چه چیزی لازم دارد تا سیستم برای او ایجاد (مثل جداول) یا استخراج (مثل پرس و جوها) کند. در واقع چگونگی ایجاد جداول یا استخراج پرس‌وجوها از دید کاربر برنامه ساز وکاربر نهایی مخفی است.

برای مثال در یک سیستم کامپیوتری بخش آموزش یک دانشگاه برای استخراج «نام و شماره دانشجویانی که معدل آنها بالای ۱۸ است» کافی است در ابتدا نام جدول یا جداول به عنوان مکان یا محل پرس و جو، سپس شرط انتخاب سطر به عنوان ملاک انتخاب سطرها و در نهایت ستون‌های نام و شماره دانشجو به عنوان ستون‌های مورد نیاز بیان شود. و به همین دلیل است که کلمه «بیانی» به این نوع زبان‌ها اطلاق می شود، زیرا کاربر برنامه ساز فقط مشخصات کامل وکلی آنچه را که لازم دارد بیان می‌کند و به جزئیات چگونگی و نحوه استخراج کاری ندارد، این مشخصات کامل و کلی به ترتیب شامل سه بخش نام جدول یا جداول به عنوان مکان یا محل پرس و جو، شرط انتخاب سطر به عنوان ملاک انتخاب سطرها و مدل انتخاب ستون به عنوان ستون‌های مورد نیاز است. در مثال فوق لازم نیست از تعداد سطرهای جدول، حلقه تکرار و غیره حرفی به میان آید. اصولاً حلقه تکرار با زبان‌های بیانی بیگانه است.

توجه: به زبان‌های بیانی، زبان‌های میهمان نیز گفته می‌شود.

توجه: یک برنامه کاربردی نوشته شده به یک زبان روالی سطح بالا پس از ارتباط با یک سرویس‌دهنده زبان بیانی مانند SQL Server از طریق مکانیزم‌های ویژه اقدام به استفاده از بانک اطلاعات می‌نماید.

توجه: هر مدل داده‌ای (مدل رابطه‌ای)، زبان بیانی خاص خود را دارد. به طور مثال SQL برای مدل رابطه‌ای ایجاد گردیده است.

در مرحله پیاده‌سازی بانک اطلاعات سه مقوله ایجاد جداول، ایجاد پرس‌وجوها و ایجاد سطوح امنیتی کاربران نهایی انجام می‌گردد. بنابراین دستورات SQL به سه دسته زیر تقسیم می‌گردد :

۱ـ دستورات تعریف داده‌ها (DDL: Data Definition Language)

کارهای مربوط به ظرف‌سازی و ساختارسازی، کارهایی از قبیل ایجاد و حذف جداول

مثال: دستور Create Table برای ایجاد جداول و دستور Drop Table برای حذف جداول

۲ـ دستورات تغییر داده‌ها (DML: Data Manipulation Language)

کارهای مربوط به تغییرات محتوای جداول، کارهایی از قبیل ذخیره داده‌ها در جداول، بازیابی داده‌ها از جداول، بروزرسانی داده‌ها در جداول و حذف داده‌ها از جداول
مثال : دستورات Insert، Select، Update و Delete

۳ـ دستورات کنترل داده‌ها (Data Control Language : DCL)

کارهای مربوط به امنیت جداول، کارهایی از قبیل ایجاد سطوح دسترسی برای کاربران نهایی
مثال: دستور Grant برای ایجاد حق دسترسی و دستور Revoke برای حذف حق دسترسی
توجه: اجتماع مجموعه دستورات DDL، DML و DCL را DSL می‌گویند. به اجتماع مجموعه دستورات DDL، DML و DCL، زبان پرس و جو(QL: Query Language) نیز گفته می‌شود.
به طور کلی دو دسته زبان داده‌ای وجود دارد :

الف) زبان‌های داده‌ای نامستقل یا ادغام شده

در این نوع زبان‌ها، زبان داده‌ای حتماً باید میهمان یک زبان سطح بالا باشد مثل SQL که در ویژوال بیسیک استفاده می‌شود.

ب) زبان‌های داده‌ای مستقل

در این نوع زبان‌ها، زبان داده‌ای نیازی به زبان سطح بالا ندارد. مانند Access و Foxpro.

شِمای بانک اطلاعات

به مجموعه ساختارهای طراحی شده در یک بانک بدون توجه به داده‌هایی که در آن‌ها قرار می‌گیرند، شِمای (Schema) بانک اطلاعات گفته می شود، برای مثال در مدل رابطه‌ای، شِمای یک بانک را جداول تشکیل می‌دهند یا برای نمونه نوع داده هر ستون به شِمای بانک مربوط می شود ولی تعداد سطرهای موجود در جدول ربطی به شِمای بانک ندارد.
توجه: هر پایگاه داده دارای قالب یا شِما(Schema) است و مجموعه‌ای از اطلاعات ذخیره شده در پایگاه داده در یک زمان خاص را نمونه پایگاه داده می‌نامند. شِمای پایگاه داده مانند تعریف نوع متغیر و نمونه شِمای پایگاه داده، مشابه مقدار متغیر در یک لحظه خاص است.

دیکشنری داده‌ها (کاتالوگ سیستم)

در یک سیستم بانک اطلاعات، اسامی زیادی مورد استفاده قرار می‌گیرند، از آنجایی که افراد زیاد و متفاوتی در یک مجموعه بانک اطلاعات درگیر هستند، مرجعی برای ایجاد یکنواختی و هماهنگی در نام داده‌ها و معنای آنها ضروری است. این مرجع، دیکشنری داده‌ها نام دارد.
در اختصار کاتالوگ سیستم یا دیکشنری داده‌ها شامل تمامی اطلاعات سیستمی مربوط به پایگاه داده‌ها همچون مشخصات سیستمی جداول و سطوح دسترسی کاربران می‌باشد که به طور خودکار

توسط DBMS بروزرسانی می‌گردد. به بیان کامل‌تر هرگونه تغییرات حاصل از دستورات DDL در پایگاه داده همچون ایجاد جداول، حذف جداول، ایجاد شاخص، حذف شاخص، ایجاد View، حذف View و یا هرگونه تغییرات حاصل از دستورات DML در پایگاه داده همچون درج، حذف و بروزرسانی رکوردها که منجر به تغییر تعداد رکوردها و به تبع آن تغییر اندازه جداول پایگاه داده‌ها می شود و یا هر گونه تغییرات حاصل از ایجاد و تغییر سطوح دسترسی توسط دستورات DCL در پایگاه داده همچون تخصیص سطوح دسترسی به کاربران و هر آنچه که مربوط به مشخصات سیستمی پایگاه داده باشد در کاتالوگ سیستم نگهداری می‌شود. در واقع شناسنامه بانک اطلاعات، دیکشنری داده‌ها یا کاتالوگ سیستم است.

توجه: دیکشنری داده در جایگاه خود، پایگاه داده‌ای سیستمی شامل اطلاعاتی سیستمی در مورد پایگاه داده یک محیط عملیاتی می‌باشد که حاوی «داده‌هایی درباره داده‌ها» است که گاهی اوقات به نام «فراداده» یا «دادگان» به معنی داده در مورد داده معرفی می‌گردد.

توجه: از آنجا که فضای ذخیره سازی دیکشنری داده ارتباطی به پایگاه داده محیط عملیاتی ندارد، بنابراین منجر به استقلال داده (پایگاه داده) از دیکشنری داده‌ها می‌گردد.

توجه: کاتالوگ سیستم به واسطه اجرای تمامی دستورات DDL و برخی دستورات DML مانند Insert و Delete و نه Select و Update دستخوش تغییر می‌گردد.

محتویات کاتالوگ سیستم

- مشخصات سیستمی جداول (مانند نام جداول و نام و نوع ستون‌های جداول)
- مشخصات سیستمی ارتباطات جداول
- مشخصات سیستمی دیدهای بانک اطلاعات (viewها)
- مشخصات سیستمی شاخص‌های بانک اطلاعات (Indexها)
- مشخصات سیستمی روال‌های ذخیره شده (Stored Procedure)
- مشخصات سیستمی تراکنش‌های بانک اطلاعات
- مشخصات سیستمی تاریخ ایجاد و بروزرسانی جداول داده‌ای
- مشخصات سیستمی کاربران و چگونگی حق دستیابی آنها به داده‌ها و محدوده مجاز عملیات آنها
- مشخصات سیستمی پایانه‌های متصل به بانک

استقلال داده‌ای

یکی از مهم‌ترین مزایای تکنولوژی پایگاه داده‌ها، بلکه مهمترین هدف آن تأمین استقلال داده‌ای است، به معنی وابسته نبودن برنامه‌های کاربردی به داده‌های ذخیره شده.

استقلال داده‌ای بر دو نوع می‌باشد:

۱ـ استقلال فیزیکی داده‌ها

به معنی مصونیت برنامه‌های کاربردی در قبال تغییراتی که در سطح فیزیکی (رسانه ذخیره سازی) پایگاه داده‌ها بروز می‌کند. یعنی اگر تغییری در ذخیره‌سازی داده‌ها انجام گیرد (برای مثال نوع دیسک عوض شود) برنامه‌های کاربردی هیچ تغییری نکند.

۲ـ استقلال منطقی داده‌ها

به معنی مصونیت برنامه‌های کاربردی در قبال تعاریف و تغییراتی که در سطح مدل طراحی (مدل رابطه‌ای) پایگاه داده بروز می‌کند. یعنی تعریف و تغییر مدل طراحی بانک (ادراکی خاص یا طراحی منطقی) از دید برنامه‌های کاربردی آنها مخفی بماند.

برای مثال مدل رابطه‌ای از تجریدی به نام جدول استفاده می‌کند و داده‌ها هر چه باشند در قالب چند جدول ریخته می‌شوند و نحوه ذخیره‌سازی داده‌ها روی رسانه‌ها از دید برنامه کاربردی مخفی است. در حالی که در روش فایلینگ تعاریف مربوط به فایل‌های داده‌ای، در فایل برنامه کاربردی می‌آمد. از آنجاکه برنامه‌های کاربردی براساس مدل طراحی بانک (ادراکی خاص یا طراحی منطقی) تعریف می‌شوند، بنابراین به طور بالقوه در معرض تأثیرپذیری از **تغییرات** در مدل طراحی بانک (ادراکی خاص یا طراحی منطقی) قرار دارند.

توجه: در سیستم‌های امروزی، این نوع استقلال هم تا حدی (و نه صددرصد) تأمین شده است.

انواع تغییر در مدل طراحی (طراحی منطقی یا ادراکی خاص)

۱ ـ رشد پایگاه داده‌ها به دلیل مطرح شدن نیازهای جدید مشتری: مانند درج جدول جدید، ترکیب جداول، تجزیه جداول.

۲ـ سازماندهی مجدد: مانند تغییر در نوع صفات خاصه، تغییر در اندازه صفات.

مثال: اگر جدولی دارای چهار ستون باشد و ستون پنجمی نیز به آن اضافه گردد، در صورتی که برنامه کاربردی سابق نیاز به دستکاری و تغییر نداشته باشد، استقلال منطقی داده‌ها براساس تغییرات نیز لحاظ شده است.

توجه: از آن جا که با حذف جداول، داده‌ها هم از بین می‌رود، بنابراین برنامه‌های کاربردی نسبت به حذف جداول هیچگاه استقلال منطقی نخواهند داشت.

تراکنش

هر برنامه‌ای که در محیط بانک اطلاعاتی توسط کاربر اجرا گردد یک تراکنش نام دارد. (مانند عملیات کارت به کارت در یک دستگاه خودپرداز بانک) تراکنش واحد کار DBMS است. به طور

کلی هر عملیاتی در پایگاه داده در قالب یک تراکنش تعریف و اجرا می‌شود. هر تراکنش شامل دو یا چند دستور SQL است. تفاوت اصلی یک تراکنش با یک برنامه معمولی در محیط غیربانکی این است که تراکنش همواره به DBMS تسلیم می‌شود و DBMS در اعمال هر گونه کنترل و حتی به تعویق انداختن و ساقط کردن آن آزادی عمل دارد. هدف اصلی از اینگونه کنترل‌ها، حذف‌ها و تعویق‌ها، حفظ جامعیت داخلی و خارجی بانک اطلاعات است.

تضمین جامعیت داخلی و خارجی بانک اطلاعات

چهار کنترل زیر لازم است روی تمامی تراکنش‌ها در بانک اطلاعات اعمال گردد تا جامعیت داخلی و خارجی یعنی رعایت اصل سازگاری آن تضمین شود.

این کنترل‌ها به خواص ACID معروفند:

۱ـ یکپارچگی یا تجزیه‌ناپذیری (Atomicity)

این خاصیت به همه یا هیچ موسوم است. منظور این است که یا تمام دستورات یک تراکنش باید اجرا شود یا هیچکدام از آنها نباید اجرا شود. این به معنی تجزیه‌ناپذیر بودن بخش‌های مختلف یک تراکنش است.

برای مثال تراکنش انتقال پول از حساب A به حساب B از دو بخش جداگانه تشکیل یافته است:

الف) بخش اول (برداشت پول)

پول را از حساب A برداشت می‌کند.

ب) بخش دوم (واریز پول)

همان پول را به حساب B واریز می‌کند.

بخش اول حساب A را بدهکار و بخش دوم حساب B را بستانکار می‌کند. در شروع و پایان یک تراکنش سیستم باید سازگار باشد ولی در اثنای اجرای تراکنش ممکن است موقتاً نیاز به ناسازگاری باشد. برای مثال هنگام واریز پول از حساب A به B، پس از برداشت پول از حساب A سیستم به طور موقت ناسازگار است و پس از واریز آن به حساب B دوباره سیستم سازگار می شود. لذا برنامه برداشت از حساب A یا واریز به حساب B به تنهایی تراکنش نیستند.

این دو بخش ممکن است روی دو کامپیوتر جداگانه اجرا شوند. فرض کنید بخش اول تراکنش اجرا شود اما ناگهان ارتباط با ماشین دوم قطع گردد. و بخش دوم قابل انجام نباشد. بدیهی است که در این حالت باید پول برداشت شده دوباره به همان حساب اول بازگردانده شود تا جامعیت بانک اطلاعات حفظ شود. این عمل معادل این است که بگوییم هیچ دستورالعملی از تراکنش انجام نشده است.

به عنوان مثالی دیگر، هنگام خرید اینترنتی با کارت عضو شتاب ممکن است پول از حساب شما

کسر گردد، اما به حساب فروشگاه مورد نظر واریز نگردد، بنابراین خرید شما ناموفق اعلام می‌گردد، که در این حالت پول حداکثر تا ۴۸ ساعت دیگر به حساب شما بازمی‌گردد.
در بیانی دیگر تراکنش را می‌توان اینگونه تعریف کرد، تراکنش مجموعه‌ای از دستورات تعریف و دستکاری داده‌هاست که DBMS تضمین می‌کند یا همه آن دستورات اجرا شوند و یا هیچکدام از آن دستورات اجرا نشوند. برای محقق‌کردن خاصیت یکپارچگی (Atomicity) هر تراکنش می‌بایست بین دو دستور Begin Transaction و End Transaction قرار گیرد. به طور کلی هر عملیاتی در پایگاه داده در قالب یک تراکنش تعریف و اجرا می‌شود. پس تراکنش واحد کار DBMS است. هر تراکنش شامل دو یا چند دستور SQL است. بر این اساس می‌توان گفت که هیچ پرس و جویی برای پایگاه داده هویت مستقل ندارد، بلکه DBMS فقط تراکنش‌ها را می شناسد و اجرا می‌کند. به این ترتیب باید گفت که هر پرس و جویی در پایگاه داده ابتدا به یک تراکنش تبدیل می‌شود و سپس اجرا می‌گردد. در SQL برای نمایش ابتدای یک تراکنش از دستور Begin Transaction و برای نمایش خاتمه یک تراکنش از دستور End Transaction استفاده می‌شود. تراکنش با اجرای Begin Transaction شروع می‌گردد و در صورت اجرای موفق commit و در صورت عدم موفقیت یعنی عدم اجرای همه بخش‌های مختلف تراکنش با اجرای دستور Rollback خاتمه می‌یابد. با اجرای این دستور کلیه تغییراتی که تراکنش روی پایگاه داده اعمال نموده است، ابطال می‌شود و وضعیت پایگاه داده به آخرین وضعیت قبل از اجرای تراکنش برگردانده می‌شود.

۲ـ سازگاری (Consistency)

به طور کلی **جامعیت** در سیستم‌های بانکی به دو طبقه‌ی جامعیت داخلی و خارجی تقسیم می‌گردد. به حفظ قوانین مطرح شده از سوی مدل رابطه‌ای و DBMS در سطح پیاده‌سازی، جامعیت داخلی و به حفظ قوانین مطرح شده از سوی طراحان و برنامه‌نویسان بانک در سطح پیاده‌سازی، جامعیت خارجی گفته می‌شود. در صورتی که در یک بانک جامعیت داخلی و خارجی هر دو توام باهم برقرار باشد، در آن بانک، **ا صل سازگاری** برقرار شده است. به عبارت دیگر سازگاری، به معنی رعایت قوانین داخلی و خارجی در بانک ا ست. DBMS م سئول کنترل قوانین داخلی و خارجی در بانک است و هرگونه عاملی که باعث نقض قوانین داخلی و خارجی و به تبع سازگاری بانک گردد را رد می‌کند. قوانین داخلی بانک شامل قانون جامعیت درون رابطه‌ای، قانون جامعیت موجودیت، قانون جامعیت ارجاعی و قانون جامعیت دامنه‌ای است، این موارد در فصل مدل رابطه‌ای بررسی خواهد شد. قوانین خارجی بانک هم شامل هر قانونی است که طراحان و برنامه‌نویسان بانک آنرا وضع می‌کنند، مانند تعریف زیردامنه برای ورود اطلاعات، تعریف بازه ۰ تا

۲۰ برای نمرات، تعریف بازه ۰ تا بینهایت برای حســـاب‌های بانکی به معنی عدم وجود موجودی منفی در حساب‌های بانکی...

در ســیســـتم‌های بانکی کنترل جامعیت داخلی و خارجی به صــورت خودکار توســط مکانیزم‌های موجود در DBMS انجام می‌گردد.

حال یکبار دیگر کد تعریف جدول SP را در نظر بگیرید:

```
Create Table SP
        (
        S# char (5),
        P# char (5),
        QTY numeric (10),
        Primary key (S#, P#),
        Foreign key (S#) References S.(S#)
          on delete cascade
          on update cascade,
        Foreign key (P#) Reference P.(P#)
          on delete cascade
          on update cascade,
        Check (QTY>1 AND QTY<1000)
        )
```

کارکرد قطعه کد زیر از کد تعریف جدول SP فوق به صورت زیر است:

```
Foreing key (S#) References S.(S#)
on delete cascade
on update casecade
```

این قطعه کد، سبب می‌گردد تا به طور خودکار هرگونه تغییری در ستون #S در جدول S به ستون #S در جدول SP نیز اعمال گردد. بنابراین جامعیت داخلی از نوع جامعیت ارجاعی نقض نمی‌گردد.

یا به طور مشابه، کارکرد قطعه کد زیر از کد تعریف جدول SP فوق به صورت زیر است:

```
Foreing key (P#) References P.(P#)
on delete cascade
on update casecade
```

این قطعه کد، سبب می‌گردد تا به طور خودکار هرگونه تغییری در ستون #P در جدول P به ستون #P در جدول SP نیز اعمال گردد. بنابراین جامعیت داخلی از نوع جامعیت ارجاعی نقض نمی‌گردد.

کارکرد قطعه کد زیر از کد تعریف جدول SP فوق به صورت زیر است:

```
Check (QTY>1 AND QTY<1000)
```

این قطعه کد، سبب می‌گردد تا به طور خودکار هرگونه مقداردهی در ستون QTY از جدول SP در بازه ۱ تا ۱۰۰۰ باشد، بنابراین جامعیت خارجی نقض نمی‌گردد.

توجه: همانطور که واضح است DBMS در حفظ جامعیت داخلی و خارجی به دقت نظارت دارد.

خاصیت سازگاری (Consistency) بیانگر این است که اگر یک تراکنش در محیط بانک اطلاعات انجام شود باید بانک اطلاعات را از حالتی سازگار به حالت سازگار دیگری منتقل کند.

به بیان دیگر هر تراکنش باید تمامی قوانین جامعیت داخلی و خارجی بانک اطلاعات را رعایت کند. خاصیت سازگاری می‌گوید انجام تراکنشی از سوی DBMS باید پذیرفته شود که جامعیت داخلی و خارجی به معنی حفظ قوانین داخلی و خارجی پایگاه داده را رعایت کند، یعنی پس از انجام تراکنش‌ها اصل سازگاری برقرار باشد، در غیراینصورت انجام تراکنش رد شود.

بنابراین تا به اینجا مشاهده شد که تراکنش ممکن است دو نوع پایان داشته باشد:

الف) پایان موفق که آن را انجام (commit) می‌نامند.

ب) پایان ناموفق که آن را سقوط (abort) می‌نامند.

۳ـ انزوا یا جداسازی (Isolation)

همزمانی در دسترسی به داده‌ها موجب بهبود کارایی و کاهش زمان پاسخ‌گویی سیستم می‌گردد. و این امر تسریع عملکرد برنامه‌ها را در پی دارد. بسیاری از سیستم‌ها اجازه می‌دهند که چندین کاربر به صورت همروند به داده‌ها دسترسی یابند و تغییرات مورد نظر خود را بر روی آنها اعمال نمایند. در چنین محیط هایی تغییرات همروند ایجاد شده بر روی داده ممکن است منجر به ایجاد ناسازگاری در داده‌ها گردد.

در سیستم‌های بانکی کاربران مختلف می‌توانند به صورت همزمان با بانک کار کنند. بنابراین اگر یک داده **خاص** بین کاربران مختلف به صورت اشتراکی مورد بازیابی و دستکاری قرار گرفت سیستم پایگاه داده باید محیطی را ایجاد نماید که مانع از بروز مشکلات و یا ایجاد نتایج نامطلوب گردد، مانند مطالب مربوط به ناحیه بحرانی در فرآیندهای همروند در درس سیستم عامل.

مثال : فرض کنید کاربر A و B به ترتیب در تهران و شیراز همزمان قصد برداشت وجه از حساب آقای ۶۰۳۷ از طریق برگ چک را دارند. بنابراین روال‌های زیر را خواهیم داشت، از آنجا که برداشت وجه از یک رکورد مشترک (عامل مشترک) صورت می‌گیرد، به تبع وقوع پدیده همزمانی برای هر دو تراکنش رخ می‌دهد. روال کار بدین صورت است که هر دو تراکنش مبلغ موجودی حساب که برابر مقدار ۵۰۰ هزارتومان می‌باشد را خوانده و با توجه به مبالغ برگ چک A و B به ترتیب مبالغ ۱۰۰ و ۵۰ هزارتومان را از حساب کسر می‌کنند و بر حسب اینکه کدام تراکنش آخرین بروزرسانی را انجام دهد مبلغ مانده حساب ۴۰۰ تا ۴۵۰ هزارتومان ذخیره می‌گردد. که در

هر دو صورت اطلاعات نادرستی ذخیره شده است. در حالی که مبلغ ۳۵۰ هزار تومان باید جهت مبلغ مانده حساب ذخیره می‌شد!

شماره مشتری	شماره حساب	نام	نام‌خانوادگی	موجودی
6037	0313	Ali	Alavi	500

Read (A)	Read (B)
A ← 500	B ← 500
R=500-100	R=500-50
R ← 400	R ← 450
Write (R) ← 400	Write (R) ← 450

بدین‌ترتیب مقادیر نادرستی توسط برنامه‌ها ذخیره شده است که این نادرستی به دلیل تداخل عملیات دو برنامه همروند است. برای جلوگیری از ایجاد چنین نتایج نامطلوبی سیستم باید نوعی نظارت بر عملکرد برنامه‌های همروند داشته باشد. مانند روش قفل‌گذاری.

در بانک اطلاعات ممکن است تراکنش‌های همروند وجود داشته باشد (مثل سیستم‌های چند برنامه‌ای) بر طبق خاصیت انزوا همروندی تراکنش‌ها باید کنترل شود تا اثر مخرب بر روی هم نداشته باشند به بیان دیگر اثر تراکنش‌های همروند روی یکدیگر چنان است که گویا هر کدام در انزوا انجام می‌شود.

به تعریفی دیگر تراکنش‌ها جدا از یکدیگر هستند. اگر چند تراکنش به طور همزمان اجرا شوند، به هنگام‌سازی‌های هر کدام از یکدیگر مخفی می‌مانند تا به اتمام برسند. به عبارتی دیگر، برای دو تراکنش مجزای A و B، تراکنش A می‌تواند بهنگام سازی‌های B را پس از پذیرفته شدن آن (commit) یا B می‌تواند بهنگام سازی‌های A را پس از پذیرفته شدن A ببیند. اما این دو تراکنش به طور همزمان نمی‌توانند، بهنگام‌سازی‌های یکدیگر ببینند.

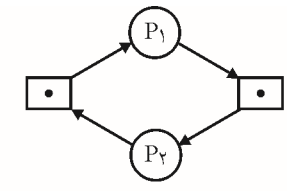
توجه: کنترل هم‌روندی توسط بخشی از DBMS به نام واحد کنترل همروندی (concurrency control) انجام می‌شود.

روش قفل‌گذاری

یکی از روش‌های اعمال خاصیت جدا سازی در تراکنش‌ها و جلوگیری از اثر مخرب تراکنش‌های همروند بر روی یکدیگر، روش قفل‌گذاری است. در این روش هنگامی که تراکنش به داده‌ای نیاز داشته باشد. تقاضای قفل کردن آن را می‌دهد و در این حالت بقیه تراکنش‌ها تا اتمام کار آن نمی‌توانند از آن استفاده کنند. در واقع استفاده از انواع قفل‌ها در مکانیسم قفل‌گذاری، روش‌های دسترسی به فیلدها و یا داده‌های اشتراکی را در کاربردهای دیگر، امکان‌پذیر می‌سازد. وجود قفل‌ها سبب می‌شود که در اجرای همروند چند تراکنش، یک رکورد مشترک (عامل مشترک) به صورت

همزمان توسط دو تراکنش مورد استفاده قرار نگیرد. این کار سرعت عملیات را افزایش می‌دهد زیرا داده اشتراکی به صورت انحصاری فقط در اختیار یک تراکنش است. ولی احتمال بروز بن‌بست را به دلیل برقراری شرط انحصار متقابل زیاد می‌کند.

بر اساس قاعده کافمن شرایط وقوع بن‌بست به صورت زیر است:



۱ـ انحصار متقابل

۲ـ انحصاری بودن

۳ـ نگهداری و انتظار

۴ـ انتظار چرخشی

۴ـ پایائی یا ماندگاری (Durabiltiy)

برا ساس این خا صیت تراکنش‌هایی که به مرحله انجام (commit) بر سند اثر شان ماندنی ا ست و هرگز به طور تصادفی از بین نمی‌روند. برای مثال اگر مبلغی به حسابی واریز شود و تراکنش مربوطه انجام یافته(commit) اعلام شود حتی در صورت وقوع حادثه در آن شعبه بانک، م شتری نباید متضرر شود، برای مثال عمل واریز قبل از اعلام انجام موفق (commit) باید در جای دیگر نیز ثبت شده باشد، مثل دیسک اصلی یا دیسک پشتیبان. یعنی تأثیرات تراکنش در پایگاه داده، ماندگار باشد. هنگامی‌که یک تراکنش دستور commit را اجرا می‌کند نتایج اجرای آن به نسخه‌ی اصلی پایگاه داده در دیسک منتقل می‌شود. بنابراین می‌توان گفت تا زمانی‌که یک تراکنش دستور commit را اجرا نکرده است یا به اصطلاح تثبیت نشده است، ویژگی ماندگاری در مورد آن تراکنش تضمین شده نیست. اما پس از اجرای دستور ماندگاری نتایج تراکنش تضمین می‌شود.

توجه: دو خاصیت یکپارچگی و پایایی توسط واحدی از DBMS به نام واحد مدیریت بازگرد (Recovery management) کنترل می‌گردد.

معماری‌های بانک اطلاعات

منظور از معماری بانک اطلاعات، پیکربندی اجزای سیستمی است که در آن حداقل یک بانک اطلاعات، یک سیستم مدیریت بانک اطلاعات (DBMS)، یک سیستم عامل، یک کامپیوتر با دستگاه‌های جانبی و تعدادی کاربر (کاربران نهایی، مدیران و برنامه‌سازان) وجود دارد و خدماتی به کاربران نهایی ارائه می‌کند. نوع معماری به عواملی همچون سخت‌افزار، نرم‌افزار مدیریت بانک اطلاعات، موقعیت جغرافیایی کاربران، نیازهای کاربران ماهیت تراکنش‌ها، حجم داده‌ها و موقعیت مکانی داده‌ها و ارتباطات بین آنها بستگی دارد.

به طور کلی دو نوع معماری برای سیستم‌های بانکی وجود دارد :

۱ ـ معماری متمرکز

۲ ـ معماری نامتمرکز
ـ معماری مشتری ـ سرویس دهنده
ـ معماری توزیع

معماری متمرکز

در این معماری یک پایگاه داده روی یک سیستم کامپیوتری و بدون ارتباط با سیستم کامپیوتری دیگر ایجاد می‌شود. و برای کاربردهای کوچک و با امکانات محدود است. سخت‌افزار این سیستم می‌تواند در حد یک کامپیوتر شخصی و یا بالاتر باشد. (مانند برنامه دفترچه تلفن شخصی)

معماری نامتمرکز

معماری مشتری ـ سرویس‌دهنده

هر معماری که در آن قسمتی از پردازش را یک برنامه، سیستم یا ماشین انجام دهد و انجام قسمت دیگری از پردازش را از برنامه، سیستم یا ماشین دیگری بخواهد، معماری مشتری ـ سرویس دهنده نامیده می‌شود.

در واقع وظایفی که باید سیستم انجام دهد به دو رده تقسیم می‌شوند: رده‌ای که انجام آنها بر عهده سرویس دهنده است و رده‌ای که توسط مشتری انجام می‌شود. بدین ترتیب یک معماری دو سطحی داریم که مشکل توزیع را تسهیل می‌کند. با این تعریف، در این معماری یک ماشین (یا سیستم یا برنامه) سرویسی را به ماشین (یا سیستم یا برنامه) دیگر ارائه می‌کند. بنابراین به این ماشین (یا سیستم یا برنامه) سرویس دهنده می‌گویند. خدماتی که سرویس‌دهنده ارائه می‌کند متنوع است، برای مثال خدمات چاپ، خدمات فایل، خدمات پست الکترونیک، خدمات اینترنت، خدمات بانک اطلاعات و بسته به نوع خدمت، سرویس‌دهنده را با نامی مناسب می‌نامند. برای نمونه سرویس‌دهنده چاپ و سرویس‌دهنده فایل، بنابراین منظور از معماری مشتری ـ سرویس‌دهنده آن نوع از معماری است که در آن مسئولیت‌ها به طور منطقی تقسیم شده است.

انواع معماری مشتری ـ سرویس‌دهنده از نظر تعداد مشتری و سرویس‌دهنده:

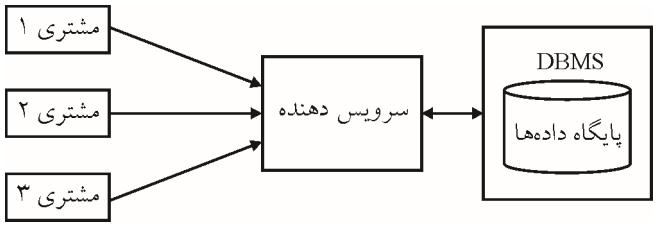
۱ـ چند مشتری ـ یک سرویس‌دهنده
۲ـ یک مشتری ـ چند سرویس‌دهنده
۳ـ چند مشتری ـچند سرویس‌دهنده

انواع معماری مشتری ـ سرویس‌دهنده از نظر پیکربندی سخت‌افزاری

معماری حول سرویس‌دهنده

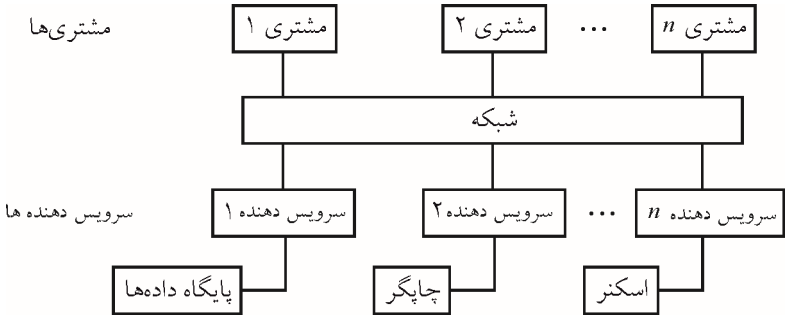
در این طرح، ماشین سرویس‌دهنده یک ابر کامپیوتر است و پایگاه داده‌ها روی همین کامپیوتر

ایجاد و مدیریت می‌شود و تعدادی مشتری، از طریق شبکه از خدمات آن استفاده می‌کنند.



معماری حول شبکه

در این طرح، تعدادی کامپیوتر شخصی به عنوان سرویس‌دهنده و تعدادی دیگر به عنوان مشتری از طریق شبکه بهم مرتبط هستند.



معماری توزیع شده

اصطلاح پردازش توزیعی بدین مفهوم است که ماشین‌های مجزا می‌توانند در یک شبکه ارتباطی به یکدیگر متصل شوند. به گونه‌ای که یک عملکرد پردازش داده منفرد می‌تواند بر روی چندین ماشین شبکه پراکنده شود. در سیستم پایگاه‌های توزیع شده، پایگاه داده‌ها بر روی چند کامپیوتر ذخیره می شود. کامپیوترها در یک سیستم توزیع شده از طریق رسانه‌های مختلف ارتباطی نظیر شبکه‌های با سرعت بالا یا خطوط تلفن به یکدیگر مرتبط هستند.

می‌توان گفت که در این معماری تعدادی پایگاه داده‌های ذخیره شده روی کامپیوترهای مختلف داریم که از نظر کاربران، پایگاه داده واحدی هستند.