بسم الله الرحمن الرحیم

پایگاه داده‌ها

مدرس

محمد جلالی

Email: Mohammadjalaly@Live.com

مؤلف: فرشید شیرافکن

فهرست مطالب

فصل ۴ : جبر رابطه ای

عملگرهای جبر رابطه ای

مثال هایی برای عملگرهای گزینش، پرتو ،تقسیم و ضرب

عملگر پیوند

پایگاه داده دانشجو و درس

پایگاه داده تهیه کننده و قطعه

عملگر نیم پیوند

عملگر نیم تفاضل

ضریب گزینش عملگر پیوند

عملگرهای فراپیوند

دسته بندی عملگرها

عملگر گروه بندی

حساب رابطه ای

آزمون

فصل ۵ : زبان رابطه ای SQL

زبان رابطه ای SQL

دستورهای SQL

توابع جمعی

مرتب سازی رکوردها

عملگر LIKE

عملگر UNION

عملگر BETWEEN

گروه بندی اطلاعات

استفاده از HAVING در دستور SELECT

پیوند رابطه ها

پرسش های تودرتو

پایگاه داده تهیه کننده – قطعه

پایگاه داده تهیه کننده – قطعه - پروژه

پایگاه داده بانک

مجوز در SQL

دید (VIEW)

متغیرها- ساختارهای تصمیم – رویه های ذخیره شده- توضیحات

آزمون

فصل ۶ : انواع وابستگی ها

انواع وابستگی ها

وابستگی تابعی

وابستگی تابعی کامل

وابستگی با واسطه

فواعد استنتاج آرمسترانگ

پیدا کردن کلید کاندید

وابستگی چند مقداری

وابستگی پیوندی

آزمون

فصل ۷ : نرمال تر سازی رابطه ها

آنومالی

صورتهای نرمال

رابطه 1NF

رابطه 2NF

رابطه 3NF

رابطه BCNF

رابطه 4NF

رابطه 5NF

ضوابط ریسانن برای تجزیه مطالوب

قضیه هیث

آزمون

فصل ۱:
مفاهیم اولیه

فایل های اطلاعاتی که به نوعی به هم مرتبط هستند، تشکیل یک بانک اطلاعاتی را می دهند .فایل شامل مجموعه ای از رکوردها می باشد و رکورد مجموعه ای از فیلدهای به هم مرتبط است و فیلد کوچکترین جزء یک بانک اطلاعاتی می باشد. مثلا در بانک اطلاعاتی دانشگاه چندین فایل وجود دارد مانند فایل دانشجویان که شامل چندین رکورد است، هر رکورد شامل اطلاعات یک دانشجو می باشد که از چند فیلد مانند شماره دانشجویی، نام، آدرس، معدل و.... تشکیل شده است.

تعاریف اولیه

داده	نمایش پدیده هاومفاهیم به صورت صوری و مناسب برای برقراری ارتباط یا پردازش.
اطلاع	داده پردازش شده می باشد.
شناخت	نمایش نمادین جنبه هایی از بخشی از جهان واقع می باشد.به عبارتی نوعی اطلاع سطح بالاتر است.
پایگاه داده ها	مجموعه ای از داده های ذخیره شده و پایا به صورت مجتمع و بهم مرتبط، با کمترین افزونگی،تحت مدیریت یک سیستم کنترل متمرکز، مورد استفاده یک یا چند کاربر به صورت همزمان و اشتراکی.

- داده همان مقدار واقعا ذخیره شده و اطلاع معنای داده است. یعنی اطلاع و داده با هم فرق دارند. اطلاع دارای خاصیت ارتباط دهندگی و انتقال دهندگی دارد، در حالیکه داده این خواص را ندارد.

- اطلاع و شناخت حاصل عملیاتی روی داده هستند ولی نوع عملیات لازم برای به دست آوردن آنها متفاوت است.

- منظور از پایایی داده ها، این است که پس از پایان اجرای برنامه کاربر، داده ها در سیستم باقی می مانند.

روش های ایجاد سیستم های کاربردی

یک سیستم کاربردی را می توان به دو روش ایجاد کرد:

۱- روش فایلینگ(ناپایگاهی)

در روش فایلینگ(سنتی)، نیازهای اطلاعاتی و پردازشی هر قسمت از محیط برآورده می شوند. مراحل اولیه طراحی و تولید برای هر قسمت به طور کلاسیک انجام شده و بعد از طراحی، مشخصات هر سیستم همراه با وظایف آنها مشخص می شود. در این روش، برای ایجاد محیط ذخیره سازی اطلاعات از یک سیستم فایل (FS) و برای برنامه سازی از یک زبان سطح بالا استفاده می شود و در نهایت برای هر قسمت، یک سیستم کاربردی ایجاد می شود.

معایب روش فایلینگ

۱- عدم وجود محیط مجتمع ذخیره سازی

۲- عدم وجود سیستم کنترل متمرکز

۳- عدم وجود ضوابط ایمنی کارا

۴- عدم امکان اشتراکی شدن داده ها

۵- تکرار در ذخیره سازی اطلاعات

۶- مصرف نامناسب امکانات سخت افزاری و نرم افزاری

۷- وابسته بودن برنامه های کاربردی به محیط ذخیره سازی داده ها

۸- حجم زیاد برنامه سازی

۲- روش پایگاهی

در این روش نیازهای اطلاعاتی تمامی قسمتها مورد مطالعه قرار می گیرد تا بتوان یک سیستم یکپارچه (integrated) طراحی کرد. داده های سازمان مدلسازی معنایی (SDM) می شوند و مشخصات سیستم یکپارچه تعیین می شود. برای سیستم مدیریت متمرکز از یک یا چند DBMS استفاده می شود. طراحی پایگاه داده ها در سطوح لازم انجام می شودو کاربران هر قسمت، پایگاه داده های خود را تعریف می کنند و با آن کار می کنند.

در واقع در روش پایگاهی یک محیط ذخیره سازی واحد، مجتمع واشتراکی، تحت کنترل متمرکز وجود دارد که کاربران براساس نیاز خاص خود، پایگاه خود را تعریف کرده و هر کاربر تصور می کند که پایگاه خود را دارد.

- کاربران در روش پایگاهی بطور همزمان از سیستم استفاده می کنند.
- در روش پایگاهی نسبت به داده های ذخیره شده، تنوع و کثرت دید وجود دارد.
- در روش پایگاهی نسبت به روش فایلینگ، حجم برنامه ها کمتر و برنامه سازی آسانتر است.
- پایگاه داده ها بر حسب تعداد رکوردهای آن، به دسته های کوچک، متوسط، بزرگ (LDB) و خیلی بزرگ (VLDB) تقسیم می شوند.

نسل های ذخیره و بازیابی اطلاعات

پنج نسل ذخیره و بازیابی اطلاعات عبارتند از:

۱- نسل فایلهای ساده ترتیبی

در این نسل از نوار به عنوان رسانه خارجی استفاده می شد. ساختار فایلها ترتیبی بود و ساختار فیزیکی و منطقی فایل یکسان بود. تنها روش پردازش فایلها، یکجا بود و نرم افزار واسطی برای مدیریت پردازش فایلها وجود نداشت امکان اشتراکی کردن داده ها وجود نداشت و تکرار در ذخیره سازی داده ها در بالاترین حد بود. هر نوع تغییر در ساختار داده ها یا رسانه ذخیره سازی موجب بروز تغییر در برنامه می شد و نسخه های متعددی از یک فایل نگهداری می شد.

۲- نسل AM

در این نسل یعنی شیوه های دستیابی (Access Methods)، نرم افزاری به نام شیوه های دستیابی ایجاد شد و برنامه کاربر را از پرداختن به جنبه های فیزیکی محیط ذخیره سازی مستقل کرد. رسانه این نسل دیسک بود و امکان دسترسی ترتیبی و مستقیم به رکوردها (نه فیلدها) وجود داشت و تا حدودی ساختار فیزیکی و منطقی فایلها از یکدیگر جدا شدند.

۳- نسل DMS

در این نسل یعنی نسل سیستم مدیریت داده ها ، نرم افزار کاملتری نسبت به نرم افزار دستیابی ایجاد شد و واسط بین برنامه های کاربردی و فایلهای محیط فیزیکی شد. در این نسل:

۱- از داده های اشتراکی در برنامه استفاده شد.

۲- آدرس دهی به داده ها در سطح فیلد ممکن شد.

۳- امکان بازیابی براساس چند کلید مهیاگشت.

۴- میزان افزونگی کاهش یافت.

۵- برنامه های کاربردی در قبال رشد فایلها مصون شدند.

۴- نسل DBMS

در نسل چهارم، برنامه های کاربردی از ویژگی های محیط فیزیکی ذخیره سازی مستقل شدند. کاهش افزونگی در ذخیره سازی داده ها، افزایش سرعت دستیابی به داده ها، بالا رفتن امنیت، امکان استفاده اشتراکی از داده ها و وجود یک محیط انتزاعی (Abstractive) از دیگر ویژگی های این نسل می باشد.

۵- نسل KBS

در این نسل یعنی نسل بانک معرفت(شناخت) به کمک مفاهیم هوش مصنوعی و سیستم های خبره، سیستمی طراحی شد که قادر به استنتاج منطقی از داده های ذخیره شده می باشد. در نسل پنجم سیستم بانک معرفت (KBS) ایجاد شد که مسئولیت ذخیره سازی شناخت، تامین جامعیت و امنیت بانک و تامین نیازهای کاربران را بر عهده دارد.

بانک معرفت، بانکی حاوی واقعیت های ساده و قواعد عام است که به طور صریح بیان شده باشند.

پایگاه بصیرت(شناخت) پویا و پایگاه داده ها ایستا می باشد.

میزان وابستگی برنامه ها به داده ها در نسل DBMS ها از نسل DMS ها کمتر است و در نسل DMS ها از نسل FS ها کمتر است.

عناصر محیط پایگاه داده ها

عناصر چهارگانه محیط پایگاه داده ها عبارتند از:

۱- نرم افزار(DBMS- نرم افزار شبکه- برنامه های کاربردی - رویه های ذخیره شده)

۲- سخت افزار (ذخیره سازی- ارتباطی- پردازشگر)

۳- کاربر(موردی (نامنظم)- همیشگی (منظم))

۴- داده

تذکر: اصلی ترین سخت افزار ذخیره سازی داده ها، دیسک است. همچنین از نوار مغناطیسی هم به عنوان رسانه کمکی برای تهیه Backup استفاده می شود.

به هر استفاده کننده از سیستم پایگاه داده ها، کاربر می گویند.

کاربر منظم یا نامنظم خود بر دو نوع برنامه ساز و نابرنامه ساز تقسیم می شوند. کاربر نابرنامه ساز از طریق منو، هدایت می شود. کاربر برنامه ساز می تواند سیستمی یا کاربردی باشد.

برنامه ساز کاربردی، برنامه های بهره برداری از پایگاه داده ها را می نویسد و برنامه ساز سیستم، برنامه های ایجاد و کنترل پایگاه داده ها را می نویسد.

یک نوع کاربر نیز به نام کاربر پایانی (End–user) وجود دارد که به هر دو نوع کاربر برنامه ساز کاربردی و کاربر نابرنامه ساز گفته می شود.

ساختارهای داده ای

یک مدل داده ای شامل یک ساختار داده (DS) است. در واقع طراحی منطقی پایگاه داده ها به کمک مفاهیم اساسی یک مدل داده ای و در چار چوب ساختار داده ای آن مدل انجام می گیرد. ساختار داده ای امکانی است برای نمایش داده های موجودیت ها و انواع ارتباطات بین آنها.

عناصر تشکیل دهنده هر مدل

۱- ساختار داده ای

۲- امکانات عملیات در پایگاه داده ها

۳- امکانات کنترل جامعیت پایگاه داده ها

انواع ساختارهای داده ای

مدل های داده ای بر پایه رکورد بر سه نوع می باشند:

۱- رابطه ای (RDS)

۲- سلسله مراتبی (HDS)

۳- شبکه ای (NDS)

تذکر: مدلهای داده ای بر پایه شیء عبارتند از: (ER , Semantic , Functional)

مفهوم ساختار داده ای بخشی از مفهوم مدل داده ای است.

ساختار رابطه ای

ساختار رابطه ای (RDS) ، دارای ویژگی های زیر می باشد:

۱- مبنای تئوریک قوی دارد. (تامین کننده محیط انتزاعی به طور کامل)

۲- مسطح بودن محیط (مانند یک فایل ترتیبی ساده)

۳- دارای نمایش ساده از نظر کاربر

۴- دارای فقط یک عنصر ساختاری اساسی (جدول)

۵- امکان نمایش ارتباطات 1:1, 1:N, N:M

۶- ساده بودن منطق و دستور بازیابی

۷- دارای رویه پاسخگوی قرینه برای پرسشهای قرینه

ساختار سلسله مراتبی

ساختار سلسله مراتبی(HDS) ، قدیمی ترین ساختار داده ای برای طراحی منطقی پایگاه داده ها می باشد که دو عنصر اساسی دارد: نوع رکورد و نوع پیوند پدر– فرزندی (PCL).

بین هر دو رکورد پشت سرهم در یک مسیر درخت، پیوند پدر فرزندی وجود دارد که ارتباط 1: N را نمایش می دهـد. در ایـن ساختار، هر رکورد فرزند، تنها یک رکورد پدر دارد، یعنی در یک نوع PCL شرکت دارد.

ویژگی های ساختار سلسله مراتبی عبارتند از:

۱- مبنای ریاضی ندارد.

۲- مناسب برای ارتباط N : 1 .

۳- سادگی نمایش ساختار رابطه ای را ندارد.

۴- نمایش ارتباط N : M در آن مشکل است.

۵- جستجو حتماً باید از ریشه انجام شود.

۶- نداشتن تقارن ساختار جدولی.

۷- داشتن تعدادی محدودیت جامعیت.

۸- مشکل بودن دستور بازیابی در آن نسبت به ساختار جدولی.

۹- نمایش ارتباط با درجه بیشتر از دو.

پایگاه داده سلسله مراتبی، مجموعه ای منظم از نمونه های یک یا چند نوع سلسله مراتب می باشد.

در HDS ، برای نمایش ارتباط چند به چند بین دو نوع موجودیت، می توان:

۱- دو نوع سلسله مراتب جدا طراحی کرد.

۲- دو نوع سلسله مراتب به هم مرتب طراحی کرد.

۳- یک نوع سلسله مراتب طراحی کرد.

در HDS ، یک نوع رکورد می تواند چند نوع رکورد فرزند داشته باشد.

در HDS ، صفات نوع ارتباط R با چندی 1:N بین دو نوع موجودیت E و F با فیلدهایی در نوع رکورد **فرزند** نمایش داده می شوند.

ساختار شبکه ای

ساختار شبکه ای (پلکس) توسط گروه DBGT پیشنهاد شد و اولین سیستم مدیریت پایگاه داده ای شبکه ای IDMS نام دارد. این سیستم گاهی به نام کوداسیل نامیده می شود. شبکه، نوعی گراف جهت دار است. در ساختار شبکه هر گره فرزند می تواند بیش از یک گره پدر داشته باشد، بنابراین گسترش یافته ساختار سلسله مراتبی است.

ساختار شبکه ای از دو عنصر تشکیل شده است:

۱- نوع رکورد

۲- نوع مجموعه (مجموعه کوداسیلی)

مجموعه کوداسیلی از سه جزء تشکیل شده است:

۱- نام مجموعه

۲- یک نوع رکورد مالک

۳- یک نوع رکورد عضو

درج مالک، بدون عضو، ممکن نمی باشد و درج یک عضو بدون وجود مالک ممکن نمی باشد.

با حذف مالک، اعضای آن نیز حذف می شوند.

ویژگی های ساختار شبکه ای

۱- تقارن دارد.

۲- مبنای ریاضی ندارد.

۳- سادگی ساختار رابطه ای را ندارد.

۴- عدم آنومالی در عملیات ذخیره سازی.

۵- فقط مخصوص نمایش ارتباطات 1:N نیست و هر نوع ارتباطی را می تواند نمایش دهد.

۶- امکان ناسازگاری داده ها در آن کمتر از سلسله مراتبی است.

۷- قواعد جامعیت ذاتی دارد.

۸- فزونکاری (به علت ایجاد و اصلاح اشاره گرها) دارد.

۹- دستور بازیابی پیچیده تری دارد.

۱۰- رویه جستجوی رکورد در آن نسبت به ساختارهای دیگر پیچیده تر است.

۱۱- اصل وحدت عملگر در یک عمل واحد مانند درج رعایت نمی شود.

در NDS ، نوع رکورد مالک یک نوع مجموعه، می تواند عضو یا مالک در نوع مجموعه دیگر باشد.

در NDS اگر n نوع موجودیت و احیانا دارای صفات چند مقداری، در یک نوع ارتباط شرکت داشته باشند، حداقل n مجموعه کوداسیلی برای طراحی پایگاه داده شبکه ای (NDB) لازم است.

در نمونه مجموعه مدل شبکه ای بر خلاف مجموعه ریاضی:

۱- حداقل یک عنصر وجود دارد.

۲- یک عنصر وجود دارد که از نظر نوع با عناصر دیگر فرق دارد.

۳- بین بعضی از عناصر نوعی نظم وجود دارد.

اگر n نوع موجودیت در یک نوع ارتباط شرکت داشته باشند و این نوع ارتباط a صفت داشته باشد، رکورد پیوند دهنده در NDS ، دارای n+a فیلد در سطح پیاده سازی است.

در NDS ، صفات نوع ارتباط R با چندی 1:N بین دو نوع موجودیت E وF با فیلدهایی در نوع رکورد **عضو** نمایش داده می شوند.

فقط در RDS ، مفهوم نظم مطرح نیست.

ساختار سلسله مراتبی تقارن ندارد، اما ساختار شبکه ای دارای تقارن است.

تعداد عناصر ساختاری اساسی در RDS برابر 1 و در HDS و NDS برابر 2 می باشد.

فقط در RDS ، جستجو در پایگاه داده ها به صورت اتوماتیک(خودکار) است.

با توجه به سه نوع ساختار رابطه ای، سلسله مراتبی و شبکه ای، سه دسته DBMS به ترتیب زمان پیدایش آنها به نام های HDBMS , NDBMS و RDBMS وجود دارد. بعد از آنها OODBMS است.

معماری پایگاه داده ها

معماری استاندارد پایگاه داده ها که توسط ANSI پیشنهاد شد، یک معماری سه سطحی می باشد. این سطوح عبارتند از:

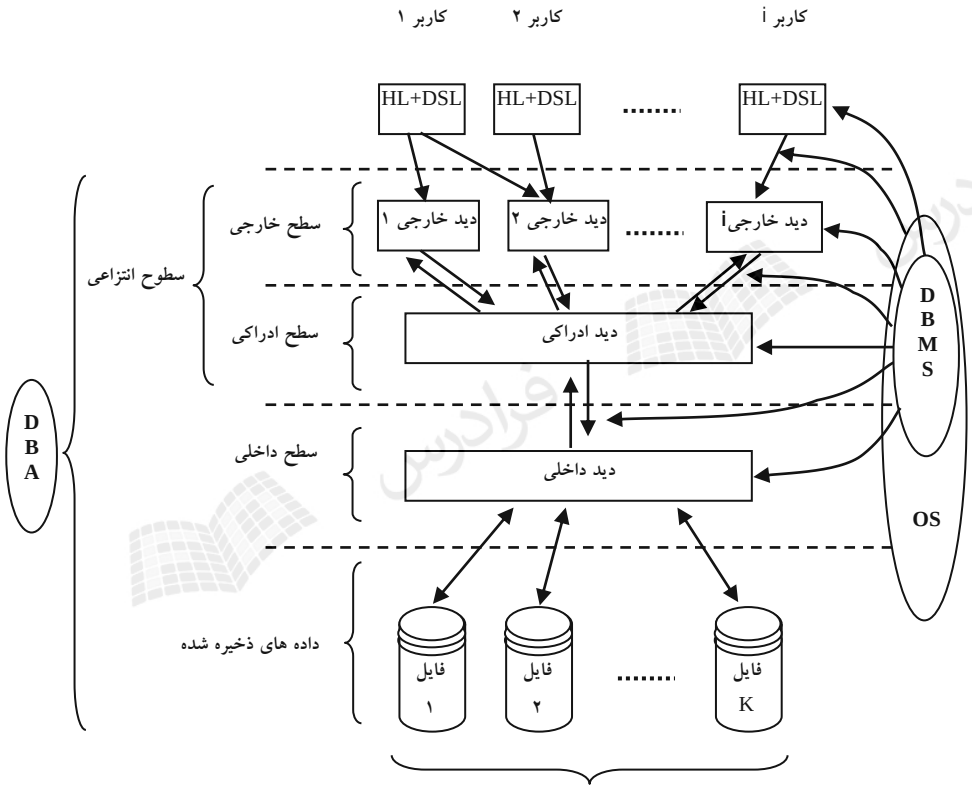
۱- سطح خارجی (External Level)

۲- سطح ادراکی یا مفهومی (Conceptual Level)

۳- سطح داخلی (Internal Level)

تذکر: به دو سطح خارجی و ادراکی، سطوح انتزاعی می گویند.

شکل زیر معماری پایگاه داده ها را با همه اجزاء آن نمایش می دهد:



شرح اجزای معماری پایگاه داده ها

۱- دید ادارکی (Conceptual View)

دید، پنجره ای است که از آن کاربر می تواند محدوده پایگاه خود را ببیند و خارج از این محدوده، چیزی نمی بیند. دید ادارکی، دید طراح نسبت به داده های ذخیره شده در پایگاه است که در برگیرنده تمام نیازهای کاربران در محیط عملیاتی می باشد. دید ادراکی چون در یک محیط انتزاعی مطرح است، بر یک ساختار داده ای مشخص بنا شده است.

شمای ادراکی:

به توصیف و شرح دید ادارکی، شمای ادراکی می گویند. در واقع شمای ادراکی برنامه ای است شامل دستورات تعریف داده ها و کنترل داده ها (نه دستورات عملیات بر روی داده ها).

۲- دید خارجی (External View)

دید خارجی، دید کاربر نسبت به داده های ذخیره شده در پایگاه داده ها می باشد. این دید:

۱- یک دید جامع نمی باشد.

۲- روی دید ادراکی طراحی و تعریف می شود.

۳- در سطح انتزاعی مطرح است.

۴- مبتنی بر همان ساختار داده ای می باشد که دید ادراکی بر اساس آن طراحی می شود.

- دید خارجی تامین کننده اشتراک داده ای در یک سیستم پایگاهی است.
- سیستمی که ساختار داده ای در سطح خارجی و سطح ادراکی آن یکسان نباشد را دو ساختاره می گویند. (مثلاً ساختار داده ای در سطح خارجی و سلسله مراتبی در سطح ادراکی)
- دید ادراکی و دید خارجی هر دو جدولی هستند، اما دید خارجی یک جدول مجازی است، یعنی داده ذخیره شده خاص خود را ندارد.
- یک کاربر می تواند چند دید خارجی داشته باشد.
- کاربر خارجی پس از تعریف دید خود، همیشه بطور پیش فرض مجاز به انجام عمل بازیابی است.

شمای خارجی:

به وصف یا تعریف دید خارجی، شمای خارجی می گویند. در واقع شمای خارجی برنامه ای است حاوی دستورات تعریف داده ها و گاه کنترل داده ها.

۳- دید داخلی (Internal View)

دید DBMS و طراح پایگاه داده ها نسبت به داده های ذخیره شده را دید داخلی می گویند.

تعاریف ایندکس های انبوه و غیر انبوه در دید داخلی وجود دارد.

شمای داخلی:

به تعریف دید داخلی، شمای داخلی می گویند. در واقع نوعی برنامه است که توسط خود DBMS تولید می شود. انواع رکوردها در شمای داخلی تعریف می شوند و شامل دستورهای لازم جهت ایجاد فایلها و کنترل آنها می باشد.

۴-کاربر

هر استفاده کننده از پایگاه داده ها را کاربر می نامند. کاربر می تواند موردی یا همیشگی باشد.

کاربر موردی معمولا برنامه ساز است.

۵- زبان میزبان (HL : Host Language)

زبان میزبان می تواند یکی از زبانهای برنامه سازی چون پاسکال ، C ، ایدا ، ... باشد.

DBMS ای که تعداد HL های مورد پذیرش آن زیاد است، مطلوبتر است، چون موجب تنوع کاربردها و کاربران می شود.

هر چه تعداد HL مورد پذیرش DBMS بیشتر شود، نیاز حافظه ای سیستم بیشتر می شود.

۶- زبان داده ای فرعی (DSL : Data Sub Language)

دستورهای این زبان به سه قسمت تقسیم می شود:

۱- دستورات تعریف داده ها (DDL)

۲- دستورات کنترل داده ها (DCL)

۳- دستورات عملیات روی داده ها (DML)

DSL می تواند هم مستقل و هم ادغام شدنی باشد. (I/E . DSL)

۱- IDSL (مستقل از زبان میزبان)

۲- EDSL (ادغام شده با زبان میزبان)

- ادغام می تواند صریح یا ضمنی باشد. در حالت ادغام صریح، عین دستورات DSL در برنامه به زبان میزبان نوشته می شود و در حالت ادغام ضمنی، دستورهای DSL به صورت توابع فرا خوانده می شوند. در حالت ادغام صریح، محیط برنامه سازی دو زبانی است و به دو کامپایلر نیاز دارد.

- DSL بهتر است از نظر برنامه سازی کامل باشد، یعنی تمام ساختهای اصلی یک زبان برنامه سازی متعارف را داشته باشد. در این حالت DSL را نمی توان دیگر یک زبان داده ای فرعی در نظر گرفت و یک زبان برنامه سازی با امکانات پایگاهی است یعنیDSL با HL ، پیوند قوی دارد.

- برای یک DSL استاندارد، کمال ساختاری، الزامی است، اما کمال محاسبه ای(برنامه سازی) الزامی نیست.

- هر گاه DSL از نظر محاسبه ای کامل باشد، در اینصورت اصلا نیازی به ادغام آن در HL نیست.

- DSL بهتر است کامپایلری باشد نه مفسری.

- سیستمی با DSL کامپایلری،زمانی کارایی دارد که نرخ پرسش های موردی پایین و نرخ پرسش های عادی بالا باشد.

زبان رویه ای و ناورویه ای

زبان ها می توانند رویه ای و یاناورویه ای باشند. زبان رویه ای حالت دستوری و زبان ناورویه ای حالت توصیفی دارد. در زبان رویه ای، برنامه ساز رویه ای برای انجام عمل مورد نظرش می نویسد، اما در زبان ناورویه ای برنامه ساز الگوریتم دسترسی به داده را مشخص نمی کند. مثلاً در هنگام بازیابی فقط می گوید چه داده ای را می خواهد و روش بازیابی را مشخص نمی کند.

- زبان ناورویه ای باعث بهبود تولید برنامه های کاربردی و تسهیل افزایش تعداد کاربران پایانی می شود.

- از نظر کاربر برنامه ساز، وقتی DSL ناورویه ای است، جستجو در پایگاه داده ها، اتوماتیک است.

۷- سیستم مدیریت پایگاه داده ها (DBMS)

این سیستم یکی از نرم افزارهای واسط بین محیط فیزیکی ذخیره و بازیابی و محیط منطقی برنامه سازی مـی باشـد. DBMS به برنامه ساز امکان می دهد تا پایگاه داده های خود را تعریف کرده و در آن عملیات خود را انجام دهد.

عوامل موثر در دسته بندی DBMS ها عبارتند از:

قیمت، نوع کاربرد، سیستم فایل ، نوع و ماهیت DSL، نوع معماری، نوع ساختار داده‌ای، محیط سخت افزاری و محـیط سیستم عامل.

نرم افزار DBMS از نمای بیرونی از دو واحد تشکیل شده است:

۱- واحد پردازشگر پرسش‌ها و برنامه‌های کاربردی

۲- واحد ایجاد و مدیریت داده‌های ذخیره شده

نرم افزار DBMS از نمای درونی از سه لایه تشکیل شده است:

۱- هسته

۲- تسهیلات نرم افزاری

۳- مدیریت محیط پایگاه داده‌ها

در طراحی سطح ادراکی و سطح خارجی پایگاه داده ها، باید امکانات DBMS را در نظر گرفت.

DBMS می داند پایگاه داده ها از چه فایل هایی تشکیل شده، استراتژی دستیابی به رکوردهای فایل چیسـت و فرمـت هر نوع رکورد داخلی چگونه است.

دلایل ایجاد سربار(فزونکاری) در DBMS عبارتند از:

۱- لزوم اعمال قواعد جامعیت

۲- لزوم انجام تبدیلات بین سطوح

۳- لزوم اعمال ضوابط ایمنی

۸- مدیر پایگاه داده ها (DBA)

مدیر پایگاه داده ها فردی است متخصص در پایگاه داده ها، با مسئولیت علمی و فنی که همراه با یک تیم تخصصی کار می کند.

وظایف تیم DBA عبارت است از:

الف – مشارکت در :

۱- تفهیم نقش داده به مدیریت سازمان

۲- تفهیم مزایای تکنولوژی پایگاه داده ها

۳- تصمیم گیری در مورد استفاده یا عدم استفاده از تکنولوژی پایگاه داده ها

۴- انتخاب DBMS و پیکربندی سخت افزاری و نرم افزاری لازم

ب – تصمیم گیری در مورد:

۱- تعیین معماری سیستم پایگاه داده ها

۲- انتخاب و انتساب اعضاء تیمهای اجرایی

۳- زبان برنامه سازی مورد نیاز

۴- مشخصات ساختار سطح داخلی پایگاه داده ها و تعیین ساختار فایلهای مناسب

۵- نوشتن شمای داخلی (طراحی فیزیکی)

۶- چگونگی رشد دادن پایگاه داده ها

۷- چگونگی سازماندهی مجدد پایگاه داده ها

۸- چگونگی ترمیم پایگاه داده ها

ج – طراحی :

۱- سطح ادراکی پایگاه داده ها (طراحی منطقی)

۲- واسطهای کاربری

۳- برنامه های کاربردی

د- نظارت بر :

۱- تعیین دیدهای خارجی و نوشتن شماهای خارجی

۲- وارد کردن داده ها

۳- تهیه مستندات لازم

۴- عملیات انجام شونده در پایگاه داده ها

استقلال داده ای

وابسته نبودن برنامه های کاربردی به داده های ذخیره شده را استقلال داده ای می نامند که مهمترین اهداف تکنولوژی پایگاه داده ها می باشد. به عبارتی استقلال داده ای عبارت است از تاثیر ناپذیری برنامه های کاربردی در سطح خارجی در قبال رشد پایگاه داده ها و تغییر در ساختار داده های عملیاتی است.

استقلال داده ای بر دو نوع است:

۱- استقلال داده ای فیزیکی

مصونیت دیدهای کاربران و برنامه های کاربردی در قبال تغییرات در سطح داخلی – فیزیکی.

۲- استقلال داده ای منطقی

مصونیت دیدهای کاربران و برنامه های کاربردی در قبال تغییرات در سطح ادراکی.

- در DBMS های جدید،استقلال داده ای فیزیکی کاملا تامین است، ولی استقلال داده ای منطقی کاملا تامین نیست.
- تغییر در سطح ادراکی یعنی تغییر در طراحی منطقی پایگاه داده ها و تغییر در شمای ادراکی، کـه ایـن تغییـر دارای دو وجه است: رشد پایگاه در سطح ادراکی و سازماندهی مجدد پایگاه در سطح ادراکی.
- از مزایای مهم تکنولوژی پایگاه داده ها،استقلال داده ای است که لازمه تامین آن، معتبر ماندن شـمای خـارجی پـس از تغییرات در شمای ادراکی و شمای داخلی است.

کاتالوگ سیستم

کاتالوگ سیستم حاوی داده هایی است در مورد داده های ذخیره شده در پایگاه داده های کاربر. به این داده های ذخیره شده، متا داده می گویند.

- متا داده از دید کاربر خارجی پنهان است.
- شمای ادراکی و خارجی در کاتالوگ سیستم نگهداری می شود.

محتویات کاتالوگ در سیستم های مختلف یکسان نیست ولی بطور کلی شامل اطلاعات زیر می باشد:

۱- شماهای خارجی، ادراکی، داخلی

۲- ضوابط کنترل ایمنی داده ها

۳- مشخصات پیکربندی سخت افزاری سیستم

۴- شرح سازمان فیزیکی داده های ذخیره شده

۵- مشخصات کاربران و حقوق دستیابی آنها به داده ها

۶- مشخصات برنامه های کاربردی

۷- مشخصات پایانه های متصل به سیستم

۸- قواعد جامعیت

۹- ارتباط بین برنامه های کاربردی و داده های ذخیره شده

۱۰- توابع تعریف شده توسط کاربران

- تعریف جدول های مبنا، تعریف شاخص ها و تعریف جدول های مجازی در کاتالوگ ذخیره می شود.
- دیکشنری فعال توسط DBA مورد استفاده قرار می گیرد.
- از دستورات DDL برای ایجاد کاتالوگ استفاده می شود.
- برای تغییر محتوای کاتالوگ از دستورات DML (مانند Insert ، Delete ، Update) استفاده می شود.
- استفاده از کاتالوگ باعث افزایش استقلال از داده می شود.
- استفاده از کاتالوگ، تغییری در سرعت ذخیره یا بازیابی اطلاعات نمی گذارد.
- مقادیر داده های درون جداول، در کاتالوگ سیستم ذخیره نمی شود.

تراکنش (TRANSACTION)

تراکنش به برنامه ای گفته می شود که یک کاربر در محیط بانک اطلاعاتی اجرا می کند. پایان یک تراکنش یا موفق (commit) است و یا ناموفق (abort).

DBMS بر روی هر تراکنش کنترل هایی را انجام می دهد تا جامعیت بانک اطلاعاتی تضمین شود. این کنترلها به ACID معروف می باشند که به ترتیب معرف Atomicity وConsistency و Isolation وDurability می باشند.

۱- یکپارچگی (Atomicity)

به این معنی است که یا تمام دستورات یک تراکنش انجام می شود یا هیچکدام از دستورات اجرا نمی شوند. این خاصیت به همه یا هیچ موسوم است.(مثلا تراکنش انتقال مبلغی از یک حساب به حساب دیگر)

۲- همخوانی (Consistency)

یعنی هر تراکنش اگر به تنهایی اجرا شود بانک را از حالتی صحیح به حالتی صحیح دیگر منتقل می کند.

۳- انزوا (Isolation)

یعنی اثر تراکنش های همروند روی یکدیگر چنان باشد که ظاهرا هر کدام به طور مجزا و در انزوا انجام می شوند.

۴- پایائی (Durability)

به این معنی است که اثر تراکنش هایی که به مرحله انجام (commit) می رسند ماندنی است و به طور تصادفی از بین نمی رود. مثلا در تراکنش انتقال پول از حسابی به حساب دیگر ، بعد از واریز مبلغ تحت هیچ شرایطی (همچون آتش سوزی) اثر عمل انجام شده از بین نمی رود.

سیستم مدیریت پایگاه داده ها در فایل ثبت تراکنش ها، ،شناسه تراکنش،زمان انجام عمل و نوع عمل را ذخیره می کند.

تراکنش های هم روند

تراکنش ها می توانند اصل سریالیتی را رعایت نکنند و به طور هم روند اجرا شوند. به طور نمونه دو تراکنش A و B را در نظر بگیرید که A دو عمل a1 و a2 و B دو عمل b1 و b2 را انجام می دهند. این اعمال در اجرای هم روند می توانند به ترتیب زیر انجام گیرند:

زمان	تراکنش A	تراکنش B
t_1	a1	-
t_2	-	b1
t_3	a2	-
t_4		b2

تذکر: مبحث هم روندی در درس سیستم های عامل (بخش همزمانی) کتاب اینجانب بررسی شده است.

تذکر: برای جلوگیری از دخالت تراکنش ها در اجرای یکدیگر، می توان از قفل استفاده کرد. مبحث قفل در درس پایگاه داده پیشرفته در مقطع کارشناسی ارشد بررسی می شود.

معماری سیستم پایگاه داده‌ها

نحوه ترکیب اجزای سیستمی، شامل حداقل یک پایگاه داده‌ها، یک DBMS ، یک سیستم عامل و یک کامپیوتر با دستگاههای جانبی و تعدادی کاربر را، معماری سیستم پایگاه داده‌ها می‌گویند.

انواع معماری‌ها عبارتند از:

۱-معماری متمرکز

یک پایگاه داده‌ها روی یک سیستم کامپیوتری ایجاد می‌شود و به سیستم کامپیوتری دیگری ارتباط ندارد.

۲- معماری مشتری خدمتگزار

در معماری مشتری خدمتگزار (Client-Server) قسمتی از پردازش توسط یک ماشین و قسمتی دیگر توسط ماشین دیگر انجام می‌شود. یعنی مسئولیت ها بطور منطقی تقسیم شده است. داده‌ها در سایت client ذخیره می‌شوند و برنامه‌های کاربردی در سایت server اجرا می‌شوند.

انواع معماری مشتری خدمتگزار(C/S DB) :

الف- معماری چند مشتری/یک خدمتگزار (MC/S DB)

ب- معماری یک مشتری/چند خدمتگزار(C/MS DB)

ج- معماری چند مشتری/چند خدمتگزار(MC/MS DB)

مزایای معماری مشتری- خدمتگزار نسبت به معماری متمرکز

الف- تقسیم پردازش

ب- اشتراک داده‌ها

ج- کاهش ترافیک شبکه

د- استقلال ایستگاههای کاری

۳-معماری توزیع شده

این معماری از ترکیب دو تکنولوژی پایگاه داده‌ها و شبکه معماری توزیع شده حاصل می‌شود. پایگاه داده‌های توزیع شده (DDB) یعنی مجموعه‌ای از چند پایگاه دادهٔ بهم مرتبط و توزیع شده روی یک شبکه که از نظر کاربران، پایگاه واحدی است.

در معماری DDB ، یک شمای ادارکی سراسری وجود دارد.

در معماری DDB ، وجود یک طرح واحد توزیع داده ها،الزامی است.

تذکر: صرف وجود شبکه در یک محیط، به معنای وجود پایگاه توزیع شده در آن محیط نیست.

مزایای معماری توزیع شده	معایب معماری توزیع شده
۱- اشتراک داده‌ها ۲- کاهش هزینه ارتباطات ۳- دستیابی بهتر به داده‌ها ۴- تسهیل گسترش سیستم ۵- استفاده از پایگاه داده‌های از قبل موجود ۶- سازگاری با سازمانهای جدید	۱- هزینه بالا ۲- مصرف حافظه بیشتر ۳- پیچیدگی پیاده سازی ۴- پیچیدگی طراحی سیستم ۵ – وجود تهدیدهای بالقوه برای امنیت سیستم

۴-معماری با پردازش موازی

نوع گسترش یافته معماری توزیع شده است که برای دستیابی پذیری بالا طراحی می‌شود. سیستم‌های موازی قادر به اجرای موازی تعداد زیادی تراکنش در ثانیه می‌باشند.

مدل های ممکن برای معماری موازی

الف- معماری با حافظه اصلی مشترک

ب- معماری با دیسک مشترک

ج- معماری بی اجزا مشترک

مزایای معماری موازی با دیسک های مشترک

الف- کاهش مصرف حافظه جانبی

ب- تسهیل گسترش سیستم

ج- تسهیل تحمل خرابی

آزمون

۱- یکپارچگی(Integration) باعث کدام مزیت در سیستم بانک اطلاعات نسبت به سیستم پرونده ای است؟
(IT- دولتی ۸۹)

۱) کاهش تکرار اطلاعات (Redundancy)
۲) سهولت کار برای کاربر نهایی
۳) سرعت بیشتر در دسترسی به اطلاعات
۴) تغییر دیسک حاوی بانک اطلاعاتی

۱-۱) یکپارچگی در سیستم بانک اطلاعات، باعث کاهش تکرار اطلاعات می شود.

۲- کدام مورد مزیت سیستم پرونده ای بر سیستم بانک اطلاعاتی است؟ (IT – دولتی ۸۹)

۱) سهولت پاسخ به سئوالات پیش بینی نشده
۲) امنیت فیزیکی اطلاعات، بیشتر
۳) سهولت دسترسی همزمان به اطلاعات
۴) سازگاری اطلاعات، بیشتر

۲-۲) از مزیت های سیستم پرونده ای بر سیستم بانک اطلاعاتی، بیشتر بودن امنیت فیزیکی اطلاعات می باشد.

۳- کدام مورد زیر می تواند از معایب سیستم بانک اطلاعاتی نسبت به سیستم پرونده ای باشد؟
(IT – دولتی ۸۸)

۱) کاهش همزمانی عملیات
۲) کاهش امنیت اطلاعات
۳) افزایش زمان اجراء برنامه های کاربردی
۴) افزایش تکرار اطلاعات(Redundancy)

۳-۲) در سیستم بانک اطلاعاتی چون اطلاعات به اشتراک گذاشته می شوند، امنیت اطلاعات کاهش می یابد.

۴- کدام یک از امکانات زیر جزء عناصر تشکیل دهنده DBMS محسوب نمی شوند؟ (دولتی ۷۲)

۱) امکان پردازش زبان طبیعی برای کار با پایگاه

۲) امکان کار با داده ها به کمک یک DSL

۳) امکان تامین جامعیت و بی نقصی (Integrity) پایگاه

۴) امکان تامین ایمنی پایگاه

۴-۱) امکان پردازش زبان طبیعی برای کار با پایگاه، جزء عناصر تشکیل دهنده DBMS نمی باشد.

۵- کدام مورد باعث عدم برقراری استقلال داده منطقی است؟ (IT – دولتی ۸۹)

۱) تغییر محتوای داده ها در سطح فیزیکی ۲) تغییر محتوای داده ها در سطح منطقی

۳) تغییر ساختمان داده ها در سطح منطقی ۴) تغییر دیسک حاوی بانک اطلاعاتی

۵-۳) تغییر ساختمان داده ها در سطح منطقی، باعث عدم برقراری استقلال داده منطقی می باشد.

فصل ۲:
مدل ER

داده های ذخیره شده در پایگاه (داده های عملیاتی یا داده های پایا)، ابتدا باید در بالاترین سطح انتزاع، مدلسازی شوند. از روشهای مدلسازی معنایی می توان روش موجودیت- ارتباط (Entity Relationship):ER را نام برد.

مفاهیم اساسی در مدل ER

در مدل ER سه مفهوم اساسی، نوع موجودیت، صفت و نوع ارتباط وجود دارد.

نوع موجودیت (Entity type)

به مفهوم کلی شیء، نوع موجودیت می گویند. در واقع هر چیزی که بخواهیم در مورد آن اطلاع داشته باشیم اعم از اینکه وجود فیزیکی یا ذهنی داشته باشد. بطور نمونه انواع موجودیت ها در محیط عملیاتی دانشکده عبارتند از: دانشجو، درس، استاد، کارمند، گروه آموزشی و ... .

انواع موجودیت

۱- موجودیت قوی (مستقل)

موجودیتی که مستقل از هر نوع موجودیت دیگر، در یک محیط مشخص مطرح باشد. مانند موجودیت درس در محیط عملیاتی دانشگاه.

۲- موجودیت ضعیف (وابسته)

موجودیتی که وجودش وابسته به یک نوع موجودیت دیگر است، اما شناسه ندارد. مانند موجودیت عضو خانواده برای موجودیت کارمند یا موجودیت اثر منتشره برای موجودیت استاد.

می گوییم نوع موجودیت E2 زیر نوع موجودیت E1 است، اگر هر نمونه از E2 الزاماً نمونه ای از E1 باشد.

صفت

ویژگی یک نوع موجودیت را صفت موجودیت می گویند. هر موجودیت دارای مجموعه ای از صفات است.

تقسیم بندی صفات

صفت را می توان از چند نظر تقسیم بندی کرد:

۱- ساده (Single) یا مرکب (Composition)

صفتی که مقدار آن از لحاظ معنایی اتومیک یا تجزیه نشدنی باشد را صفت ساده می گویند(مانند صفت درس) و صفتی که از چند صفت ساده تشکیل شده باشد را مرکب می گویند.

تذکر: صفت مرکب تجزیه شدنی به صفات ساده است. مانند صفت آدرس که دارای اجزا نام شهر، نام خیابان و ... می باشد.

اگر نیازی به اجزاء تشکیل دهنده صفت مرکب نباشد آنرا به صورت یک صفت ساده در نظر می گیریم. یعنی مفهوم تجزیه پذیری نسبی است نه مطلق.

۲- تک مقداری یا چند مقداری

صفت تک مقداری صفتی است که به ازای یک نام صفت، حداکثر یک مقدار برای یک نمونه از موجودیت، داشته باشیم. مانند شماره دانشجویی که نمی تواند بیش از یک مقدار داشته باشد و اگر بتوان به ازاء یک نام صفت، چند مقدار برای یک نمونه از موجودیت داشته باشیم، آن صفت را چند مقداری می گوییم. مانند مدرک دانشگاهی برای موجودیت استاد که می تواند دارای مقادیری برای لیسانس، فوق لیسانس و دکتری باشد. یا شماره تلفن برای موجودیت دانشکده که می تواند چند مقدار داشته باشد.

صفت مرکبی که دارای چند مقدار است را صفت پیچیده (مختلط) می گویند. صفت مرکب آدرس- تلفن یک صفت پیچیده (complex) محسوب می شود. چون یک شخص می تواند چند آدرس داشته باشد و در هر آدرس بیش از یک شماره تلفن.

۳- ذخیره شده (Stored) یا مشتق (Derived)

صفتی که مقادیرش در پایگاه داده ها ذخیره شده باشد صفت ذخیره شده نام دارد. صفتی که مقادیرش در پایگاه ذخیره نشده باشد و از طریق محاسبه روی داده های ذخیره شده بدست آید را صفت مشتق می گویند.

صفت معدل دانشجو یک صفت مشتق است چون مستقیماً در پایگاه ذخیره نشده است و از طریق یکسری محاسبه ها روی داده های موجود در پایگاه داده ها، حاصل می شود.

۴- شناسه (کلید)

صفت شناسه موجودیت، صفتی است که دارای دو ویژگی یکتایی مقدار و کوتاه بودن طول مقادیر می باشد.

۵- مقدار هیچ پذیر

مقدار یک صفت برای برخی از نمونه های یک نوع موجودیت می تواند تعریف نشده (مقدار هیچ) باشد. مانند شماره تلفن یک نمونه کارمند که در پایگاه داده موجود نیست، یا نام استاد یک درس در ترم جاری که تا حالا اعلام نشده است، یا نمره دانشجویان که تا پایان ترم مقداری ندارد.

بعضی از جنبه های صفت، لزوما از میدان آن ناشی می شوند.مانند: نوع، معنا،صفت و محدودیت میدانی.

نوع ارتباط

نوع ارتباط یعنی تعامل (interaction) بین دو یا بیش از دو نوع موجودیت (و یا بین یک نوع موجودیت با خودش) که دارای یک معنای مشخص است و با یک نام بیان می شود. نوع ارتباط حالت خاصی از نوع موجودیت است.

به طور مثال، دو نوع ارتباط حذف و انتخاب بین موجودیتهای دانشجو و درس وجود دارد:

۱- دانشجو درس را حذف می کند.

۲- دانشجو درس را انتخاب می کند.

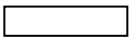
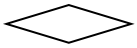
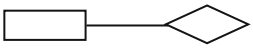
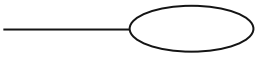
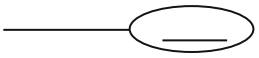
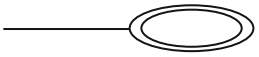
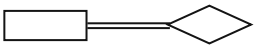
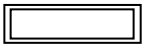
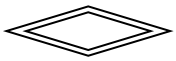
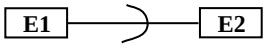
یک نوع ارتباط، حداقل یک شرکت کننده و همیشه یک معنا دارد.

یک نوع ارتباط می تواند صفت نداشته باشد. یعنی اگر N تعداد صفات نوع ارتباط R باشد،در این صورت : N>=0 .

اگر نوع موجودیت ضعیف را با صفت چند مقداری نمایش دهیم، انعطاف پذیری مدلسازی انجام شده، کاهش می یابد.

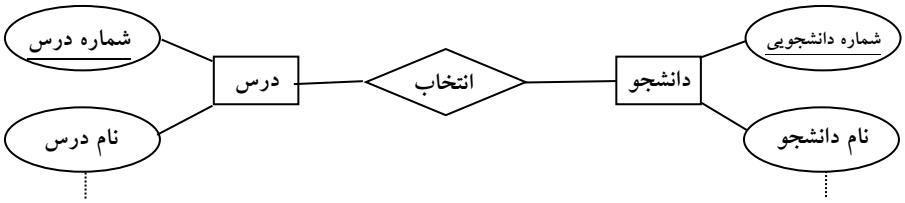
نمودار ER

این نمودار مدل کلی پایگاه داده ها در بالاترین سطح انتزاع است که سه مفهوم اساسی مدل ER (نوع موجودیت، صفت و نـوع ارتباط) توسط نمادهایی، نمایش داده می شود.

معنا	نماد
موجودیت	
نوع ارتباط	
مشارکت موجودیت در ارتباط	
صفت	
صفت شناسه	
صفت چند مقداری	
صفت مرکب	
صفت مشتق	
مشارکت الزامی	
موجودیت ضعیف(وابسته)	
نوع ارتباط با موجودیت ضعیف	
ارتباط " گونه ای اســت از ... " E2 IS-A E1	

مثال

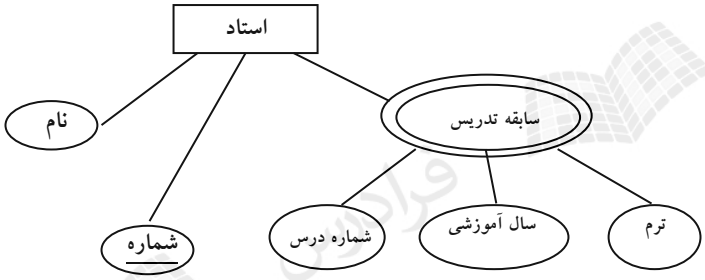
نمودار ER زیر نوع ارتباط انتخاب بین موجودیتهای دانشجو و درس را مشخص می کند:



نمودار بالا را چنین می خوانیم: دانشجو درس را انتخاب می کند. و یا درس توسط دانشجو انتخاب می شود.

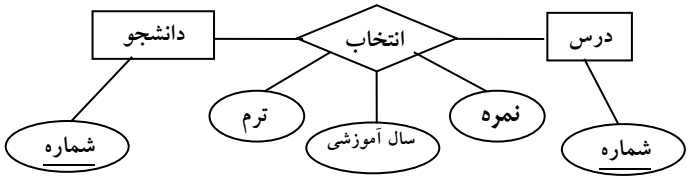
مثال

صفت سابقه تدریس برای استاد، چند مقداری است:



مثال

نوع ارتباط می تواند دارای صفت باشد. صفت نوع ارتباط ، می تواند تک مقداری یا چند مقداری باشد. در شکل زیـر، انتخـاب، صفت نوع ارتباط با سه صفت "سال آموزشی، ترم و نمره" می باشد.



انواع مشارکت

مشارکت بر دو نوع است: ۱- الزامی (کامل) ۲- غیر الزامی (ناکامل)

مشارکت یک موجودیت در یک ارتباط را الزامی گویند، اگر همه نمونه های آن موجودیت در آن ارتباط شرکت کنند، در غیر اینصورت مشارکت غیر الزامی است.

شکل زیر معرف مشارکت الزامی بین دانشجو و درس در رابطه انتخاب است:

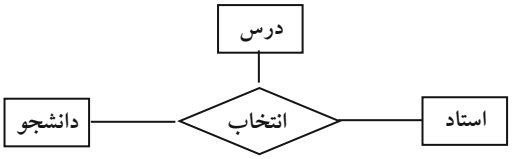
- مشارکت بین دانشجو و درس در رابطه حذف الزامی نمی باشد چون لزوماً همه دانشجویان درسی را حذف نمی کنند.
- نوع ارتباط نیز خود نوعی موجودیت است (موجودیت ضعیف)، بنابراین می تواند دارای صفاتی باشد که معمولاً فاقد صفت کلید است. مثلاً ارتباط انتخاب دارای صفاتی چون ترم، نمره و سال آموزشی می باشد.
- ارتباط یک نوع موجودیت ضعیف با یک نوع موجودیت قوی، دارای صفت نمی باشد.
- در مشارکت نوع موجودیت ضعیف در نوع ارتباط شناسا، مشارکت در نوع الزامی است.

درجه نوع ارتباط

تعداد شرکت کنندگان در یک ارتباط را درجه(چندی) آن ارتباط گویند.

مثال

ارتباط زیر از درجه سه گانی (Ternary) است:



معنای این ارتباط : دانشجوی st درس c را با استاد pr انتخاب می کند.

- چندی نوع ارتباط R ، همان نرخ کاردینالیتی R است.
- چندی نوع ارتباط در وضع مشارکت نوع موجودیت ها در ارتباط تاثیر ندارد.

انواع تناظر

انواع تناظر عبارتند از:

۱- یک به یک (1:1) ۲- یک به چند (1:N) ۳- چند به چند (N:M)

- در ارتباط یک به یک موجودیت R1 با R2 ، یک نمونه از R1 حداکثر با یک نمونه از R2 ارتباط دارد و برعکس.
- در ارتباط یک به چند R1 با R2 ، یک نمونه از R1 با تعدادی از نمونه های R2 ارتباط دارد ولی یک نمونه از R2 حداکثر با یک نمونه از R1 ارتباط دارد.
- در ارتباط چند به چند R1 با R2 ، یک نمونه از R1 با تعدادی از نمونه های R2 ارتباط دارد و برعکس.

مثال

تناظر بین موجودیت های زیر N : 1 است، یعنی یک دانشجو یک درس را حذف می کند ولی یک درس ممکن است توسط چند دانشجو حذف شود.

مثال

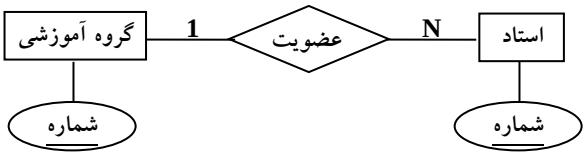
ارتباط بین موجودیت های زیر 1:1 است. یعنی یک استاد می تواند فقط مدیر یک گروه آموزشی باشد و هر گروه آموزشی فقط یک مدیر دارد.

مثال

ارتباط بین موجودیت های زیر N :M است، یعنی یک درس توسط چند دانشجو می تواند انتخاب شود و یک دانشجو می تواند چند درس را انتخاب کند.

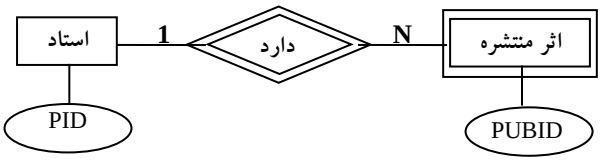
مثال

ارتباط بین گروه آموزشی و استاد 1:N است. یعنی در هر گروه اموزشی، N استاد عضویت دارند و هر استاد فقط در یـک گـروه آموزشی عضو می باشد:



مثال

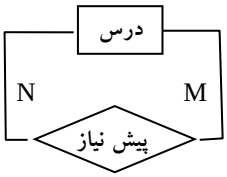
ارتباط بین استاد و اثر منتشره 1:N است. یعنی هر استاد می تواند N اثر داشته باشد و هر اثر متعلق به یک اسـتاد اسـت. اثـر منتشره یک موجودیت ضعیف است.



با فرض ضعیف بودن E نسبت به F ، چندی نوع ارتباط شناسا از سوی F به E باید 1:N باشد. همچنین صـفتی از F در مجموعه صفات E ، باید کلید باشد.

مثال

ارتباط پیشنیازی بین موجودیت درس N:M است :



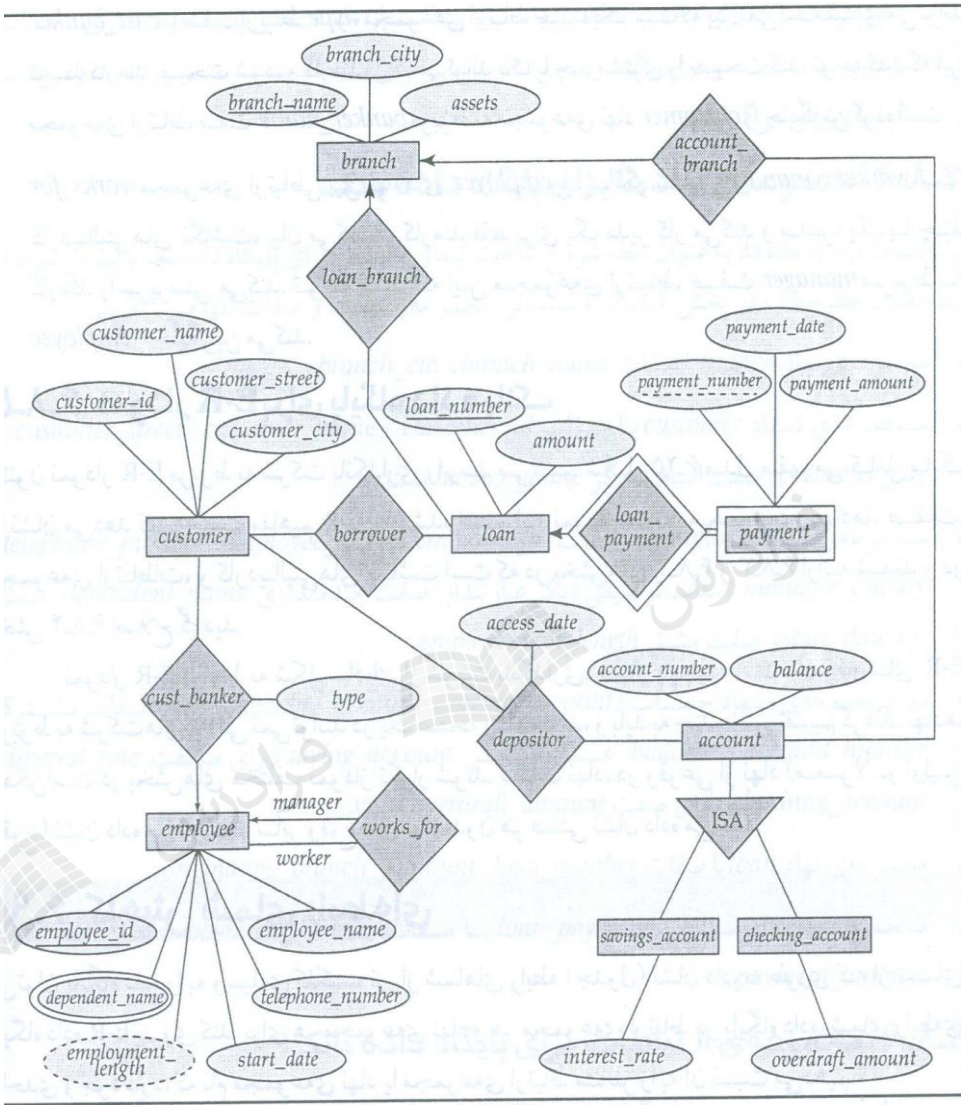
یک درس می تواند پیش نیاز چند درس باشد.

مثال

تناظر ارتباط زیر 1:N است:

مثال

نمودار ER مربوط به شرکت بانکداری



اگر E1 ’IS-A-PART-OF‘ E2 ، در این صورت E2 هیچ صفتی از E1 را به ارث نمی برد.

دام های پیوندی (Connection traps)

یکی از معایب مدلسازی به روش ER ، دامهای پیوندی است که از درک نادرست معنای ارتباطات ناشی می شود. دام های پیوندی رایج عبارتند از:

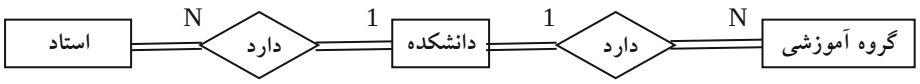
۱- دام یک چندی(چند شاخه) (Fan trap)

۲- دام شکاف(گسل) (Chasm trap)

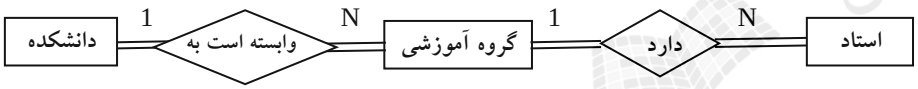
تذکر: یک دام به نام دام پیوندی حلقه ای وجود دارد که در این کتاب بررسی نشده است.

دام یک – چندی

این دام وقتی ایجاد می شود که ارتباطی بین چند نوع موجودیت وجود داشته باشد، اما مسیر ارتباطی مبهم باشد. در نمودار ER زیر یافتن پاسخ سئوال "در چه گروهی استاد شیرافکن عضویت دارد؟" ممکن نمی باشد.

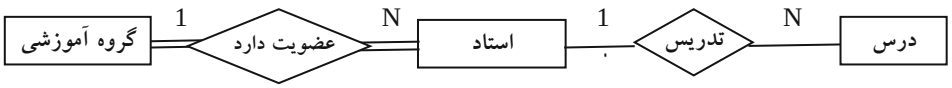


مدل سازی را به صورت زیر باید اصلاح کرد تا چنین دامی ایجاد نشود:

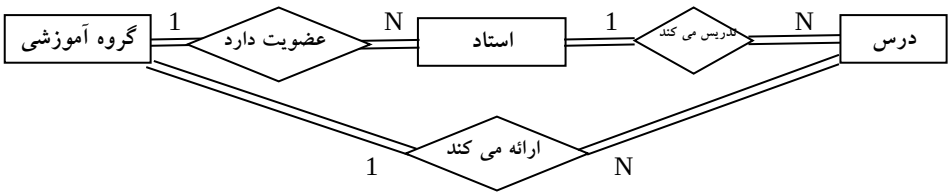


دام شکاف

این دام وقتی ایجاد می شود که مشارکت یک موجودیت در یک ارتباط الزامی نباشد. نمودار زیر معرف این دام است. در این حالت نمی توان به سئوال "درس پایگاه داده ها در کدام گروه آموزشی ارائه می شود" پاسخ داد. (با فرض اینکه استادی این درس را ارائه ندهد.)



برای رفع مشکل دام پیوندی، مدلسازی را به روش زیر انجام می دهیم.

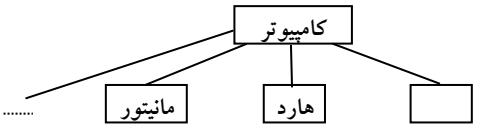


روش EER (Extended ER)

روش EER ، روشی است که در آن می توان از مفاهیم زیر که در مدلسازی شیء گرا وجود دارد، استفاده کرد.

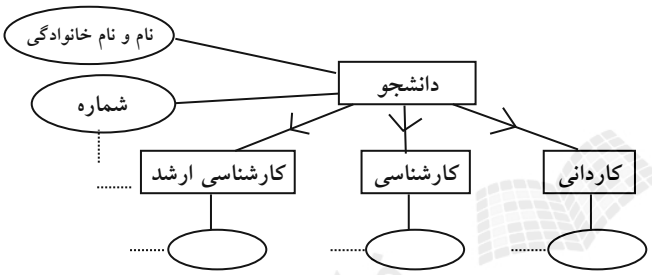
۱- تجزیه و ترکیب

جداسازی یک شییء کل به اجزاء تشکیل دهنده آن را تجزیه و عکس عمل تجزیه را ترکیب می نامند. شکل زیر مثالی از تجزیه و ترکیب است:



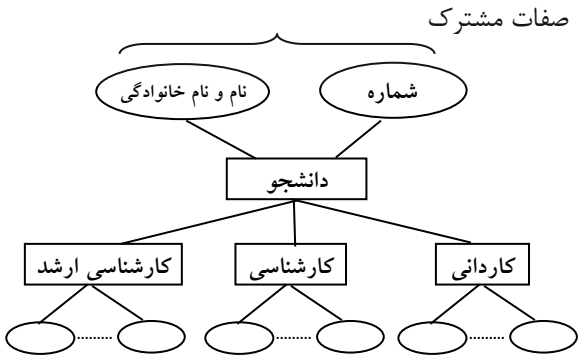
۲- تخصیص و تعمیم

تخصیص یعنی مشخص کردن انواع خاص یک موجودیت است. مثالی از تخصیص برای موجودیت دانشجو:



موجودیت دانشجو دارای سه زیر نوع "دانشجوی کاردانی، کارشناسی و کارشناسی ارشد" می باشد که صفاتی چون شماره و نام در همه آنها وجود دارد.(صفات مشترک)

تعمیم عکس عمل تخصیص است. تعمیم یعنی تشخیص حداقل صفت شناسه مشترک بین حداقل دو نوع موجودیت و انتساب آن صفت به یک نوع موجودیت جدید است. مثال زیر معرف تعمیم است.



تذکر: در تعمیم با داشتن زیر نوعهای خاص، صفات مشترک بین آنها را در یک مجموعه صفات برای نوع ارشد در نظر گرفته و آنها را برای هر زیر نوع تکرار نمی کنیم.

ارتباط ’IS-A‘ با تکنیک **تخصیص** و ارتباط ’IS-A-PART-OF‘ با تکنیک **تجزیه** نمایش داده می شود.

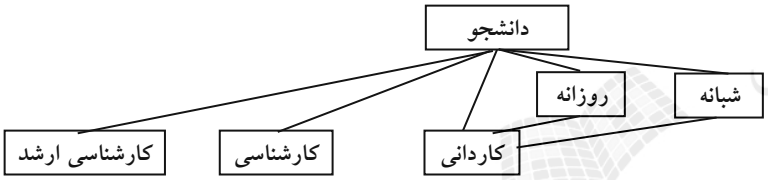
اگر نوع موجویت E حاصل تعمیم n نوع موجویت باشد، در این صورت n>=2 .

اگر N تعداد زیر نوع ها در تخصیص کامل باشد، در این صورت N>=2 .

اگر N تعداد صفات مشترک بین n نوع موجودیت باشد، در تعمیم این n نوع موجویت، n>=1 یک شرط لازم است.

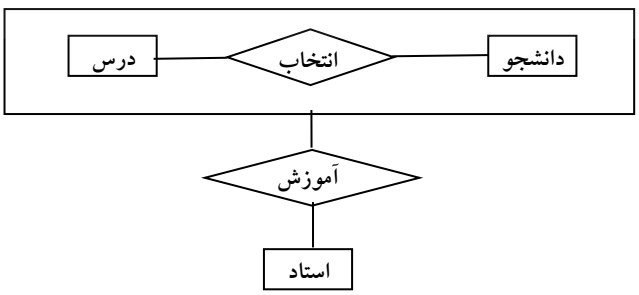
۳- وراثت چندگانه (Multiple inheritance)

یک زیر نوع می تواند، متعلق به چند موجودیت باشد. در مثال زیر، نوع دانشجوی کاردانی هم متعلق به موجودیت دانشجو می باشد و هم متعلق به دانشجوی روزانه و هم دانشجوی شبانه:



۴- تجمیع

می توان دو یا بیش از دو موجودیت مرتبط را به صورت یک موجودیت واحد در نظر گرفت. زمانی از تجمیع استفاده می کنیم که بخواهیم ارتباطی را بین ارتباط ها بیان کنیم. در مثال زیر تجمیع نمایش داده شده است.



از تکنیک تجمیع وقتی استفاده می کنیم که بخواهیم:

۱- درجه یک نوع ارتباط را کاهش دهیم.

۲- کارایی سیستم پایگاهی را برای برخی از برنامه ها افزایش دهیم.

۳- یک نوع ارتباط با یک نوع ارتباط دیگر را در مدلسازی منظور کنیم.

از تجمیع برای نمایش ارتباط بین حداقل یک نوع موجویت با یک نوع ارتباط دیگر استفاده می شود.

اگر E1 وE2 دو نوع موجودیت مجتمع شونده در تکنیک تجمیع باشند، می توانند متمایز نباشند.

اگر E1 وE2 دو نوع موجودیت مجتمع شونده در تکنیک تجمیع باشند، می توانند در تمام صفات مشترک باشند.

اگر E1 وE2 دو نوع موجودیت مجتمع شونده در تکنیک تجمیع باشند، در این صورت بین آنها باید حداقل یک نوع ارتباط وجود داشته باشد.

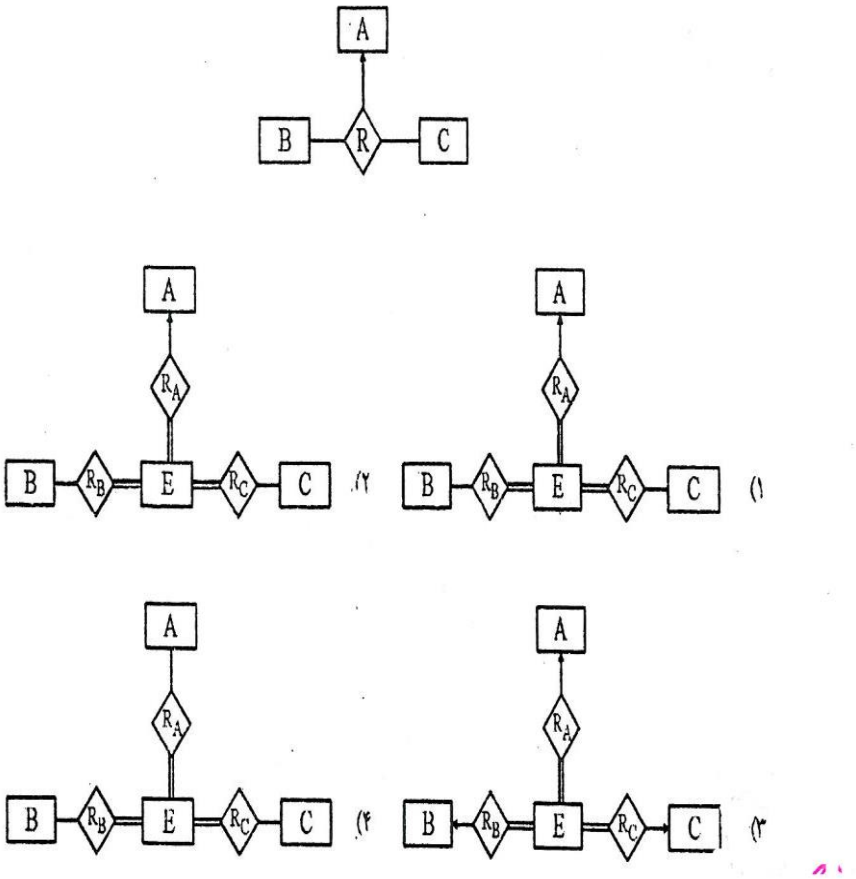
در تکنیک تجمیع ، اگر n تعداد نوع موجویت های مجتمع شونده باشد، آنگاه n>=1.

در روش ER گسترش یافته، وابستگی پیوندی قابل نمایش نیست.

در روش ER گسترش یافته، بعضی از محدودیت های خرد جهان واقع، قابل نمایش نیستند.

آزمون

۱- فرض کنید که رابطه سه تایی زیر بین موجودیت های A,B,C وجود دارد. حال اگر بخواهیم این رابطه سه تـایی را با رابطه های دودویی نمایش دهیم، کدام یک از نمودارهای موجودیت-رابطه (ERD) زیر دقیقا معادل بـا ایـن رابطه سه تایی می باشد؟ (IT- دولتی ۸۸)



۳-۱) اگر بخواهیم رابطه سه تایی بین موجودیت های A,B,C را با رابطه های دودویی نمـایش دهـیم، بایـد یـک موجودیـت جدید مانند E اضافه کنیم. برای این که رابطه سه تایی حفظ شود، موجودیت E باید:

الف- با موجودیت های قبلی ارتباط کامل و اجباری داشته باشد. (ارتباطی دو خطه =)

ب- موجودیت های A,B,C با موجودیت E یک ارتباط یک به چند داشته باشند.

فصل ۳

مدل رابطه ای

در سال ۱۹۷۰، کاد مدل رابطه ای را معرفی کرد. ساختار داده ای در مدل رابطه ای، مفهوم ریاضی رابطه است که موجب قوی شدن انتزاع در این مدل شده است. مفهوم مدل داده ای گسترده تر از مفهوم ساختار داده ای است. مدل داده ای تامین کننده محیط انتزاعی پایگاه داده ها می باشد و از ۳ بخش ساختاری، عملیاتی و جامعیتی تشکیل شده است.

تعریف رابطه

رابطه را از دو منظر تعریف می کنیم.

تعریف رابطه از نظر کاد

رابطه R تعریف شده روی n مجموعه S_1 تا S_n، زیر مجموعه ای از ضرب کارتزین آنها می باشد.

مثال

رابطه Student (S1,S2,S3) روی سه مجموعه زیر تعریف شده است:

S1 = { شماره } , S2 = { نام } , S3 = { نمره }

که نمایش جدولی آن به صورت زیر است:

S1	S2	S3
120	Ali	14
198	Reza	20

میدان

میدان، مجموعه ای است نامدار از مقادیر همنوع که یک یا بیش از یک صفت از آن مقدار می گیرند. از نظر کاد، مفهوم میدان،گسترش یافته مفهوم نوع داده است. مزایای میدان عبارتند از:

۱- امکانی برای کنترل مقداری پرسش ها

۲- امکانی برای کنترل معنایی پرسش ها

۳- امکانی برای تسریع پاسخدهی به برخی از پرسش ها

۴- امکانی برای ساده تر شدن شمای پایگاه داده ها.

در مدل رابطه ای هر دو صفت هم میدان، همنوع هستند.

تعریف رابطه از نظر دیت

با فرض وجود n میدان D1 , D2 , … , Dn نه لزوماً متمایز، رابطه از دو قسمت تشکیل شده است:

۱- عنوان (Heading) : مجموعه اسامی صفات خاصه

۲- بدنه (Body) : مجموعه ای از تاپل ها

- عنوان رابطه مجموعه ای است ثابت در زمان اما بدنه رابطه، مجموعه ای متغیر در زمان.
- درجه همان تعداد صفات و کاردینالیتی همان تعداد سطرها می باشد.
- تناظر بین مفاهیم رابطه ای و مفاهیم جدول:

  رابطه = جدول، تاپل = سطر ، صفت = ستون ، درجه = تعداد ستونها ، کاردینالیتی= تعداد سطرها

  تذکر: برای مفهوم کلید، متناظری نداریم.
- جدول باید حداقل یک ستون داشته باشد،اما رابطه می تواند اصلا صفت نداشته باشد.
- وجود هیچمقدار در جدول مجاز است و در رابطه مجاز نیست.
- رابطه، عنصر ساختاری اساسی در مدل رابطه ای است.
- کاردینالیتی عنوان و کاردینالیتی بدنه رابطه بزرگتر یا مساوی صفر است.

مثال

در رابطه R که در زیر نشان داده شده، مجموعه عنوان، مجموعه بدنه، درجه و کاردینالیتی رابطه را بیان کنید.

A	B	C
a1	b1	c1
a2	b2	c2

حل: رابطه R (A,B,C) دارای ۳ صفت خاصه و۲ سطر است، بنابراین درجه رابطه ۳ و کاردینالیتی آن ۲ است.

مجموعه عنوان رابطه R : HR = {A,B,C}

و مجموعه بدنه رابطه R : BR = {{ a1 , b1 , c1 } , {a2 , b2 , c2 } }

خواص رابطه

یک رابطه دارای خواص زیر است:

۱- تاپل تکراری ندارد.

یک مجموعه دارای عناصر تکراری نمی باشد و چون تاپلها، عناصر مجموعه پیکر هستند، تاپل تکراری در رابطه وجود ندارد.

۲- تاپلها نظم ندارند.

عناصر مجموعه دارای نظم نمی باشند و چون تاپلها، عناصر مجموعه پیکر هستند، نظم ندارند.

۳- صفات رابطه نظم ندارند (از چپ به راست).

عناصر مجموعه دارای نظم نمی باشند و چون صفات، عناصر مجموعه عنوان هستند، دارای نظم نمی باشند.

۴- مقادیر تمام صفات، تجزیه نشدنی(اتومیک) هستند.

صفت اتومیک، صفتی است که اگر آن را به اجزایی تجزیه کنیم، اجزای حاصل بی معنا باشند. البته تجزیه ناپذیری مفهومی نسبی است و بستگی به کاربردهای خاص دارد. مثلاً صفت تاریخ می تواند از سه جزء سال، ماه و روز تشکیل شود که این اجزاء ممکن است در یک کاربرد با معنا و در کاربردی دیگر بی معنا باشند.

مثال

صفت TY در رابطه R(ID,NAME,TY) تجزیه شدنی است. در واقع TY معرف ترم–سال است و مثلاً به صورت 1/90-91 نمایش داده می شود. که از دو قسمت ترم 1 و سال 90-91 تشکیل شده است.

صفات رابطه نظم ندارند، یعنی به صفات رابطه از طریق نام آنها دسترسی می شود، در حالیکه ستونهای جدول نظم دارند، یعنی به ستونهای جدول از طریق مکان آنها دسترسی می شود.

به دلیل خاصیت تک مقداری بودن صفات رابطه، مدل رابطه ای در نمایش داده های پیچیده مشکل دارد.

اگر در رابطه ای میدان ها مجزا باشند، در این صورت درجه یک رابطه با تعداد میدان های آن رابطه برابر است، و اگر میدان ها مجزا نباشند، درجه از تعداد میدان ها بزرگتر است.

مفاهیمی مانند "نظم، نشانی و اشاره گر" از مفاهیم فایلینگ، در مدل رابطه ای مطرح نیست.

انواع کلید

انواع کلید عبارتند از:

کلید کاندید(C.K)	هر زیر مجموعه از مجموعه عنوان که دارای خاصیت یکتایی مقدار و کاهش ناپذیری باشد.
کلید اصلی(P.K)	یکی از کلیدهای کاندید که توسط طراح پایگاه داده انتخاب می شود.
کلید بدیل(A.K)	هر کلید کاندید غیر از کلید اصلی.
کلید خارجی (F.K)	با فرض وجود دو رابطه R1 , R2 ، هر زیر مجموعه از صفات R2 که در R1 کلید کاندید باشد، کلید خارجی R2 است. (R1 و R2 لزوما متمایز نیستند)
سوپر کلید(S.K)	هر ترکیبی از اسامی صفات رابطه که در هیچ دو تاپل، مقدار یکسانی نداشته باشد.

(C.K : Condidate Key) (P.K : Primary Key) (A.K : Alternate Key)
(F.K : Foreign Key) (S.K : Super Key)

تذکر: اگر R1 فقط یک کلید کاندید داشته باشد، در تعریف کلید خارجی به جای کلید کاندید از کلید اصلی نیز می توان استفاده کرد.

نکاتی در رابطه با کلید کاندید

۱- کلید کاندید می تواند ساده یا مرکب باشد.

۲- رابطه ممکن است بیش از یک کلید کاندید داشته باشد.

۳- کلید کاندید امکانی است برای ارجاع به یک تاپل در رابطه .

۴- کلیدهای کاندید یک رابطه ممکن است صفت مشترک داشته باشند.

۵- رابطه ای که کلید کاندید آن از ترکیب تمام صفات رابطه حاصل می شود، تمام کلید (ALL KEY) نام دارد.

۶- در شمای ادراکی، کلیدهای کاندید باید معرفی شوند.

۷- کلید کاندید می توان هیچمقدار داشته باشد.

۸- هر رابطه ای حتماً کلید اصلی دارد، چون هر رابطه حداقل یک کلید کاندید دارد.

۹- اگر تعداد کلیدهای کاندید رابطه R برابر N باشد، تعداد کلیدهای بدیل برابر N-1 است.

۱۰- حداکثر تعداد کلیدهای کاندید ساده یک رابطه درجه n برابر n است.

۱۱- حداکثر تعداد کلیدهای کاندید یک رابطه درجه n برابر $C^{n}_{\left\lceil \frac{n}{2} \right\rceil}$ است.

۱۲- یک رابطه با درجه n ، حداکثر دارای $\frac{n!}{m!\times(n-m)!}$ کلید کاندید m صفتی است. $(m \geq 1)$

۱۳- حداکثر تعداد کلیدهای کاندید N صفتی دوبدو و ناهمپوشای یک رابطه درجه n برابر $\left\lfloor \frac{n}{N} \right\rfloor$ است.

۱۴- کلید کاندید کاهش ناپذیر است. یعنی اگر یکی از عناصر کلید حذف شود، باقیمانده لزوماً کلید کاندید نیست.

نکاتی در رابطه با کلید خارجی

۱- کلید خارجی می تواند مقدار تکراری داشته باشد.

۲- کلید خارجی می تواند مقدار تهی (Null) داشته باشد.

۳- کلید خارجی برای نمایش ارتباطات بین انواع موجودیت ها بکار می رود.

۴- کلید خارجی یک رابطه، می تواند با نام دیگر، کلید کاندید در همان رابطه باشد.

۵- کلید خارجی یک رابطه، می تواند با نام دیگر، کلید کاندید در رابطه ای غیر از آن رابطه باشد.

۶- کلید خارجی یک رابطه، می تواند با هر نام، کلید کاندید در هر تعداد رابطه باشد.

۷- تنها امکان نمایش ارتباط بین دو موجودیت، کلید خارجی نیست بلکه یک صفت مشترک نیز می تواند یک ارتباط ایجاد کند.

۸- از معایب کلید خارجی می توان بروز افزونگی و فزونکاری سیستم به خاطر کنترل جامعیت را نام برد.

۹- در کلید خارجی، با افزایش افزونگی، کار لازم برای کنترل جامعیت افزایش می یابد.

۱۰- کلید خارجی و کلید بدیل ممکن است در رابطه ای وجود نداشته باشند.

۱۱- در رابطه نشان دهنده نوع ارتباط با چندی 1:N ،کلید خارجی لزوما جزء تشکیل دهنده کلید کاندید نیست.

۱۲- در یک رابطه با درجه n و با یک کلید کاندید ساده، حداکثر n کلید خارجی ساده می تواند وجود داشته باشد.

۱۳- تعداد کلید خارجی یک رابطه می تواند صفر باشد.

۱۴- رابطه با درجه n، حداکثر 2^{n-1} کلید خارجی دارد.

نکاتی در رابطه با سوپرکلید

۱- هر کلید کاندید، یک سوپر کلید است.

۲- هر سوپر کلید(ابر کلید)، شامل حداقل یک کلید کاندید است.

۳- سوپر کلید دارای خاصیت یکتایی مقدار است.

۴- همه صفات با سوپر کلید وابستگی تابعی دارند.

۵- سوپر کلید، کاهش پذیر است.

۶- سوپر کلید می توان هیچمقدار داشته باشد.

۷- اگر H_R عنوان رابطه R ، $G \subset H_R$ و t_i و t_j دو تاپل دلخواه متمایز از R باشند و $t_i(G) \neq t_j(G)$ ، در اینصورت G سوپر کلید R است.

۸- رابطه با درجه n، حداکثر $2^n - 1$ سوپر کلید دارد.

۹- رابطه با درجه n، با دو کلید کاندید ساده، دارای $3 \times 2^{n-2}$ سوپر کلید است.

۱۰- یک رابطه با درجه n و تعداد k کلید کاندید ساده،دارای $2^n - 2^{n-k}$ سوپر کلید دارد.

مثال

در رابطه R(A,B,C,D,E,F,G) ، صفات A و (B,D) کلیدهای کاندید هستند. چند سوپر کلید را نام ببرید.

حل:

CBDFG , BDEF , ABCD

■

میدان اصلی، میدانی است که مقادیرش حداقل در یک رابطه، کلید اصلی باشند.

مثال

کلید های کاندید و خارجی را در رابطه های زیر مشخص کنید.

ST (شماره گروه آموزشی، رشته تحصیلی،سطوح دوره تحصیلی ، نام دانشجو ، شماره دانشجویی)

CT (شماره گروه آموزشی ارائه کننده درس ، نوع درس ، تعداد واحد ، عنوان درس ، شماره درس)

SCT(..... شماره درس ، شماره دانشجویی)

حل: **کلید کاندید**:

شماره دانشجویی در ST و شماره درس در CT و ترکیب شماره درس و شماره دانشجو در SCT.

تذکر: اگر عنوان هیچ دو درس یکسان نباشد آنگاه می توان عنوان درس را نیز در CT کلید کاندید در نظر گرفت.

کلید خارجی:

شماره دانشجویی در SCT کلید خارجی است، چون همین صفت در ST کلید اصلی است.

شماره درس درSCT کلید خارجی است، چون همین صفت در CT کلید اصلی است.

■

مثال

رابطه پیشنیازی تمام کلید است و در این رابطه دو کلید خارجی وجود دارد.

PT (شماره درس پیشنیاز , شماره درس)

■

مثال

بانک تهیه کننده- قطعه با سه جدول s,p,sp مفروض است:

S (s# , sname , status , city)
P (p# , pname , color , weight , city)
SP(s# , p# , qty)

در رابطه اول اطلاعات تهیه کننده، در رابطه دوم اطلاعات قطعه و در رابطه سوم اطلاعات میزان تولید از هر قطعه توسط تهیه کنندگان نگهداری می شود.

حل:

کلید اصلی : صفت s# در رابطه s و صفت p# در رابطه p و ترکیب صفات s#,p# در رابطه sp.

کلید خارجی: صفت s# در رابطه sp کلید خارجی است چون در رابطه s کلید اصلی است.

صفت p# در رابطه sp کلید خارجی است، چون در رابطه p کلید اصلی است. ■

مثال

در دو رابطه زیر کلیدهای کاندید و خارجی را تعیین کنید.

DEPT(شماره دپارتمان , نام دپارتمان , تلفن , شماره مدیر دپارتمان)

PROF (شماره مدرس , نام مدرس , شماره دپارتمان مدرس)

حل:

کلید های کاندید: شماره دپارتمان درDEPT و شماره مدرس در PROF .

کلیدهای خارجی: شماره مدیر دپارتمان درDEPT کلید خارجی است چون درPROF کلید اصلی است.

شماره دپارتمان مدرس درPROF کلید خارجی است چون درDEPT کلید اصلی است.

لازم به ذکر است که مدیر دپارتمان خود یک مدرس است.

■

مثال

صفت شماره دپارتمان باعث ایجاد ارتباط بین دو رابطه زیر می باشد، در حالیکه کلید خارجی محسوب نمی شود:

CT(شماره دپارتمان و ... و نام درس و شماره درس)

PT(شماره دپارتمان و ... و نام مدرس و شماره مدرس)

■

قواعد جامعیت

جامعیت پایگاه داده ها یعنی صحت، دقت و سازگاری داده های ذخیره شده در پایگاه در تمام لحظات. بروز عواملی چون اشتباه در ورود اطلاعات، اشتباه در برنامه های کاربردی، وجود افزونگی کنترل نشده و خرابی های سخت افزار و نرم افزاری موجب نقض جامعیت می شوند. **قواعد جامعیت بر ۳ نوع است:**

۱- قاعده میدانی

قاعده مشخص کننده مقادیر مجاز یک میدان (مثلاً مقادیر میدان نمره اعداد از ۰ تا ۲۰ است)

۲- قواعد خاص

قواعد جامعیت خاص(کاربردی)، قواعدی هستند که توسط کاربر، مجاز تعریف می شوند. DBMS به کاربر امکان تعریف این قواعد جامعیت را می دهد. مجموعه قواعد خاص یک محیط عملیاتی، باید مورد تایید مدیر داده ها(DA) برسد و سپس DBA آنها را در طراحی و پیاده سازی منظور نماید.

قواعد خاص بر سه نوع می باشند:

الف- قاعده صفتی : قاعده بیان کننده نوع صفت

مثلاً صفت نام دانشجویی از نوع کاراکتر است.

ب- قاعده رابطه ای : قاعده بیان کننده مقادیر مجاز یک متغیر رابطه ای

مثلاً در رابطه درس، درس عملی از گروه آموزشی خاصی نمی تواند بیشتر از دو واحد داشته باشد.

ج- قاعده پایگاهی : قاعده ناظر به چند متغیر رابطه ای مرتبط با هم.

مثلاً در رابطه های دانشجو،درس، مدرس و دانشجو– درس– مدرس این محدودیت وجود داشته باشد که "مدرس با مرتبه فوق لیسانس به بالا نباید درسی از دوره کاردانی را تدریس کند."

تذکر: در سیستم های موجود برای معرفی این قواعد از مکانیسم اظهار (Assertion) استفاده می شود.

۳- قواعد عام

قواعد عام (متا قواعد)، قواعدی که توسط هر سیستم رابطه ای در پایگاه رابطه ای اعمال می شوند و بـه داده هـای خـاص وابسته نیستند و بر دو نوع می باشند:

الف- **قاعده جامعیت موجودیتی:** هیچ جزء تشکیل دهنده کلید اصلی نباید تهی باشد.

ب- **قاعده جامعیت ارجاعی** : مقدار کلید خارجی یک رابطه نمی تواند در رابطه مرجع وجود نداشته باشد.

- مقدار هیچ، بلانک یا صفر نیست و می تواند به عنوان عملوند در عملیات محاسباتی استفاده شود.
- از مفهوم مقدار هیچ در مدل رابطه ای باید اجتناب کرد.
- بر طبق قاعده جامعیت ارجاعی، اگر صفتی در R2 کلید خارجی باشد، در اینصورت یا می تواند مقدار هیچ داشته باشد یا باید حتماً مقداری داشته باشد که در رابطه مرجع (R1) وجود داشته باشد.
- محدودیت های میدانی، صفتی، موجودیتی و ارجاعی را محدودیت های ساختاری می گویند.
- قاعده جامعیت موجودیتی در مدل رابطه ای را می توان ذاتی مدل دانست.
- در مدل رابطه ای، معرفی کلید خارجی برای کنترل جامعیت پایگاه داده ها لازم است.

راههای اعمال قواعد جامعیت

۱- معرفی کلید اصلی

۲- معرفی کلید خارجی

۳- معرفی میدان و مقادیر آن

۴- معرفی وابستگی های تابعی

۵- اعلام هیچ مقدار ناپذیری صفت

۶- اعلان محدودیت ها در شمای پایگاه

- رابطه R1 با درجه N1 و رابطه R2 با درجه N2 مفروضند. اگر K1 صفت از R1 وK2 صفت از R2 ،از یـک میـدان مقـدار بگیرند، مجموع میدان های متمایز که این دو رابطه روی آنها تعریف شده اند، برابر است با: N1+N2-(K1+K2)+1

آزمون

۱- کدام یک از گزاره های زیر در مدل رابطه ای صحیح است؟ (دولتی۸۶)

۱) ترتیب چند تایی(Tuple) های یک رابطه مهم است.

۲) ترتیب خصیصه های(Attributes) یک رابطه مهم است.

۳) هر رابطه دارای کلید خارجی(Foreign key) است.

۴) هر رابطه حداقل دارای یک نامزد کلیدی(Candidate key) است.

۴-۱) هر رابطه حداقل دارای یک نامزد کلیدی(Candidate key) است.

می دانیم که:

الف- ترتیب چند تایی های یک رابطه مهم نمی باشد.

ب- ترتیب خصیصه های یک رابطه مهم نمی باشد.

ج- هر رابطه دارای کلید خارجی نمی باشد.

۲- برای تعیین کلید اصلی یک رابطه توجه به کدام یک از موارد زیر ضروری است؟ (دولتی ۹۰)

۱) مقادیر ابر کلیدهای رابطه

۲) عملیات مورد نیاز روی رابطه

۳) معنی خصیصه های رابطه در دنیای خارج

۴) مقادیر خصیصه های رابطه در زمان بار کردن رابطه

۳-۲) برای تعیین کلید اصلی یک رابطه، توجه به معنی خصیصه های رابطه در دنیای خارج ضروری است.

۳- رابطه های r و s را در نظر بگیرید. کدام زیر مجموعه از صفات r می تواند کلید خارجی باشد؟

(دولتی ۹۰)

$r(\underline{R_1} : D_1, R_2 : D_2, R_3 : D_3)$

$s(\underline{R_2} : D_2, R_3 : D_3, R_4 : D_4)$

۱) R_2 ۲) R_3 ۳) R_1, R_2 ۴) R_2, R_3

۴-۳) صفت R_2 می تواند کلید خارجی رابطه r باشد، چون در S کلید اصلی است.

۴- کدام عبارت در مورد کلید خارجی (Foreign Key) در مدل رابطه ای صحیح است؟ (دولتی ۸۸)

۱) کلید خارجی یکی از (Candidate Key) های همان رابطه است.

۲) کلید خارجی بایستی کلید اصلی رابطه دیگری باشد. (نباید Alternate Key باشد.)

۳) کلید خارجی بایستی خصیصه ساده (Simple Attribute) باشد.

۴) کلید خارجی یک رابطه می تواند متناظر با مقادیر Candidate Key همان رابطه باشد.

۴-۴) کلید خارجی یک رابطه می تواند متناظر با مقادیر کلید کاندید همان رابطه باشد.

فصل ۴:
جبر رابطه ای

کاربر برای اینکه بتواند عملیات خود را روی پایگاه داده ها انجام دهد می تواند از امکانات جبر رابطه ای و یا حساب رابطه ای که جزئی از مدل رابطه ای هستند، استفاده کند. از موارد کاربرد جبر رابطه ای می توان تعریف حیطه اعمال ضوابط ایمنی پایگاه داده ها ، تعریف حیطه عملیات کنترل همروندی تراکنش ها و تعریف ضوابط جامعیت پایگاه داده ها را نام برد. جبر رابطه ای، از نظر رابطه ای کامل است، چون می توان هر رابطه معتبر از مجموعه رابطه های ممکن را به کمک یک عبارت جبر رابطه ای بیان کرد. جبر رابطه ای بسته است،چون حاصل عملکرد هر یک از عملگرهای جبر رابطه ای، یک رابطه است.

عملگرهای جبر رابطه ای

نام عملگر	علامت	نوع استفاده	هدف
گزینش	σ	$\sigma_{COND}(R)$	رکوردهایی از رابطه R را بر می گرداند که در شرط COND صدق می کنند.
پرتو	Π	$\Pi_{Attribute-list}(R)$	رابطه جدیدی شامل صفات رابطه R (با حذف رکوردهای تکراری)
اجتماع (UNION)	$\cup$	$A \cup B$	شامل رکوردهایی که در A یا B یا هر دو وجود دارند.
اشتراک (INTERSECT)	$\cap$	$A \cap B$	شامل رکوردهایی که در A و در B وجود دارند.
تفاضل (MINUS)	-	$A - B$	شامل رکوردهایی که در A وجود دارند ولی در B وجود ندارند.
تقسیم (DIVIDE)	$\div$	$A \div B$	شامل رکوردهایی از A که شامل همه رکوردهای B باشد.
ضرب کارتزین (TIMES)	$\times$	$A \times B$	کلیه ترکیب های ممکن دو رابطه A و B
پیوند (JOIN)	$\bowtie$ یا ∞	$A \bowtie B$	رابطه ای شامل همه فیلدهای دو رابطه با شرط برابری فیلد مشترک آنها

عملگرهای گزینش و پرتو

برای آشنایی با نحوه عملکرد عملگرهای "گزینش و پرتو" ، چند مثال می زنیم.

مثال

با توجه رابطه A که در زیر آورده شده است، مطلوب است، حاصل $\sigma_{sname=sn2}(A)$.

S#	SNAME	CITY
S1	SN1	C1
S2	SN2	C2
S3	SN3	C3

حل:

حاصل گزینش سطرهایی که صفت sname آنها برابر sn2 است برابر است با:

S#	SNAME	CITY
S2	SN2	C2

مثال

با توجه رابطه A که در زیر آورده شده است، مطلوب است، حاصل $\Pi_{sname}(A)$:

S#	SNAME	CITY
110	ALI	HAMEDAN
120	SARA	TEHRAN
130	ALI	KERMAN

حل:

حاصل پرتو روی ستون sname برابر است با:

SNAME
ALI
SARA

تذکر: در نتیجه عملگر پرتو، سطر تکراری حذف می شود.

مثال

اگر L_1 و L_2، دو مجموعه از صفات رابطه R(H) باشند، در چه صورت داریم:

$$\Pi_{<L1>}(\Pi_{<L2>}(R)) = \Pi_{<L1>}(R)$$

حل: در صورتی که : $L_1 \subseteq L_2$.

مثال

رابطه R(H) مفروض است. اگر $A \subset H_R$ مجموعه ای از صفات باشد که در مسند عملگر گزینش R وجود داشته باشد و $B \subset H_R$ مجموعه صفات پرتوی از R باشد. در چه صورت داریم:

$$\Pi_{<B>}(\sigma_A(R)) = \sigma_A(\Pi_{<B>}(R))$$

حل: در صورتی که : $A \in B$.

■

با فرض اینکه p و q ، مسند گزینش از رابطه R باشد، داریم:

$$\sigma_{p\wedge q}(R) = \sigma_p(\sigma_q(R))$$
$$\sigma_p(\sigma_q(R)) = \sigma_q(\sigma_p(R))$$
$$\sigma_{p\wedge q\wedge r}(R) = \sigma_p(\sigma_q(\sigma_r(R)))$$

اگر C_R کاردینالیتی رابطه R باشد،آنگاه : $C_{\Pi(R)} \le C_R$. البته اگر در صفات پرتو، کلید کاندید رابطه R وجود داشته باشد، آنگاه : $C_{\Pi(R)} = C_R$.

عملگرهای UNION ، INTERSECT و MINUS

دو رابطه ای که در عملگرهای اجتماع، اشتراک و تفاضل استفاده می شود، باید از نظر نوع سازگار **(Type Compatible)** باشند، یعنی درجه دو رابطه یکسان بوده و همچنین میدان هایی که روی آنها تعریف شده اند،یکسان باشند.

عمل درج تاپل با عملگر UNION و عمل حذف تاپل با عملگر MINUS انجام می شود.

مثال

دو رابطه A و B مفروض هستند. مطلوب است: $A \cup B$ ، $A \cap B$ و $A - B$

A

S#	SNAME	CITY
S1	SN1	C1
S2	SN2	C2

B

S#	SNAME	CITY
S1	SN1	C1
S3	SN3	C3

حل:

A - B

S#	SNAME	CITY
S2	SN2	C2

A ∩ B

S#	SNAME	CITY
S1	SN1	C1

A ∪ B

S#	SNAME	CITY
S1	SN1	C1
S2	SN2	C2
S3	SN3	C3

■

تساوی مقابل برقرار است: $(R \cap S) - T = R \cap (S - T)$

تساوی مقابل برقرار است: $R \cap (S \cup T) = (R \cap S) \cup (R \cap T)$

تساوی مقابل برقرار است:

$\sigma_{C1\ OR\ C2}(R) = \sigma_{C1}(R) \cup \sigma_{C2}(R)$

که می تواند به صورت زیر نیز بیان شود:

$(R\ \text{WHERE}\ C_1\ OR\ C_2) \equiv (R\ \text{WHERE}\ C_1)\ UNION\ (R\ \text{WHERE}\ C_2)$

تساوی های زیر برقرار هستند:

$\sigma_{C1\ AND\ C2}(R) = \sigma_{C1}(R) \cap \sigma_{C2}(R)$

$\sigma_{X=a}(R_1) \cup \sigma_{X=a}(R_2) = \sigma_{X=a}(R_1 \cup R_2)$

$\sigma_{X=a}(R_1) \cap \sigma_{X=a}(R_2) = \sigma_{X=a}(R_1 \cap R_2)$

$\sigma_{X=a}(R_1) - \sigma_{X=a}(R_2) = \sigma_{X=a}(R_1 - R_2)$

$\sigma_{NOT\ C}(R) = R - (\sigma_C(R))$

عملگرهای DIVIDE و TIMES

برای ضرب دو رابطه R1 و R2 ، هر تاپل R1 با تمام تاپلهای R2 ترکیب می شود.

مثال

مطلوب است حاصل ضرب R2 و R1 .

R1

A
A1
A2
A3

R2

B
B1
B2

حل: هر تاپل R1 با تمام تاپلهای R2 ترکیب می شود. $R_1 \times R_2$ برابر است با :

A	B
A1	B1
A1	B2
A2	B1
A2	B2
A3	B1
A3	B2

مثال

مطلوب است حاصل تقسیم R بر هر یک از رابطه های A,B,C .

R

S#	P#
S1	P1
S1	P2
S1	P3
S2	P1
S2	P2
S3	P2
S4	P2
S4	P3

A

P#
P1
P2
P3

B

P#
P2
P3

C

P#
P1

حل:

$R \div A$

S#
S1

$R \div B$

S#
S1
S4

$R \div C$

S#
S1
S2

حاصل در تمام تقسیم های بالا، جدولی است با ستون S# ، که شامل شماره تهیه کنندگانی است که همه P# را شامل می شوند. به طور نمونه جواب $R \div A$ ، برابر S1 است، چون تنها S1 است که همه قطعات P1,P2,P3 را تهیه کرده است.

مثال

مطلوب حاصل $R_1 \div R_2$.

R1

A	B	C
A1	B1	C1
A1	B1	C3
A2	B2	C2
A3	B1	C1
A3	B1	C3

R2

C
C1
C3

حل:

A	B
A1	B1
A3	B1

زوج (A1,B1) و (A3,B1) همه فیلدهای C یعنی C1 و C3 را شامل می شوند.

مثال

مطلوب است حاصل $R_1 \div R_2$.

R1

A	B	C
A1	B1	C1
A1	B2	C2
A1	B2	C3
A2	B1	C1
A2	B3	C2

R2

B	C
B1	C1
B2	C2

حل: حاصل تقسیم جدولی است با ستون A :

A
A1

■

مثال

مطلوب است حاصل $R_1 \div R_2$.

R1

A	B	C
1	5	9
2	6	1
3	5	2
1	6	9

R2

B
5
6

حل: حاصل تقسیم جدولی است با ستونهای A و C :

A	C
1	9

■

شبیه سازی عمل $\div$ با عملگرهای $\times$و $-$ و Π :

$$A(X,Y) \div B(Y) = \Pi_{<x>}(A) - (\Pi_{<x>}(\Pi_{<x>}(A) \times B - A))$$

مثال

معادلی برای مجموعه عبارات زیر پیدا کنید.

$$T_1 \leftarrow \Pi_C(A)$$
$$T_2 \leftarrow \Pi_C((B \times T_1) - A)$$
$$T \leftarrow T_1 - T_2$$

حل: $A \div B$

■

$(A \ \ DIVIDE \ \ B) \ \ TIMES \ \ B \subseteq A$

اگر $(A \ \ TIMES \ \ B)DIVIDE \ \ B = X$ و $(A \ \ DIVIDE \ \ B)TIMES \ \ B = Y$ ،

در این صورت: $X \supseteq Y$

اگر $R = \Pi_{<HA>}(A \times B)$ ، آنگاه $R = A$.

عملگر پیوند

برای محاسبه join دو رابطه، ابتدا دو رابطه را ضرب کرده و سپس سطرهایی که دارای شرط join هستند را گزینش کرده و در نهایت ستونهای تکراری را توسط عملگر پرتو حذف می کنیم. یعنی پیوند($\triangleright\triangleleft$)، عملگری است که از سه عملگر $\Pi, \sigma, \times$ تشکیل شده است.

مثال

مطلوب است حاصل پیوند دو رابطه S و SP .

S

S#	SNAME
1	ALI
2	SARA
3	OMID

SP

S#	P#	QTY
1	20	100
1	30	250
2	20	50
3	30	80

حل: ابتدا حاصل ضرب دو رابطه را بدست می آوریم:

S.S#	SNAME	SP.S#	P#	QTY
1	ALI	1	20	100
1	ALI	1	30	250
1	ALI	2	20	50
1	ALI	3	30	80
2	SARA	1	20	100
2	SARA	1	30	250
2	SARA	2	20	50
2	SARA	3	30	80
3	OMID	1	20	100
3	OMID	1	30	250
3	OMID	2	20	50
3	OMID	3	30	80

حال سطرهایی که S# آنها با یکدیگر برابر نمی باشد را حذف می کنیم:

S.S#	SNAME	SP.S#	P#	QTY
1	ALI	1	20	100
1	ALI	1	30	250
2	SARA	2	20	50
3	OMID	3	30	80

و در نهایت یکی از ستون های S# را حذف می کنیم:

S#	SNAME	P#	QTY
1	ALI	20	100
1	ALI	30	250
2	SARA	20	50
3	OMID	30	80

مثال

با توجه به پایگاه "تهیه کننده – قطعه"، معادل دستورات زیر را بنویسید.

$$T \leftarrow \sigma_{S.S\#=SP.S\#}(S \times SP)$$

$$\prod_{S.S\#\ ,SNAME,\ P\#,\ QTY}(T)$$

حل:

$$S \quad JOIN_{S.S\#=SP.S\#} \quad SP$$

مثال

اگر R1 و R2 حاوی اطلاعات زیر باشند، R1 $\triangleright\!\triangleleft$ R2 چند سطر خواهد داشت؟

R1

A	B
1	2
2	3
3	4
4	5

R2

A	C
1	3
3	6
3	5

حل: ستون مشترک دو رابطه یعنی A ، باید دارای مقدار یکسان باشند:

A	B	C
1	2	3
3	4	6
3	4	5

مثال

اگر R1 و R2 حاوی اطلاعات زیر باشند، R1 ⋪ R2 چند سطر خواهد داشت؟

R1

A	B	C
1	2	3
2	3	4
3	4	5
4	5	6

R2

A	B	D
2	3	7
2	3	8
4	5	9

حل: ستون های مشترک دو رابطه یعنی A و B ، باید دارای مقدار یکسان باشند:

A	B	C	D
2	3	4	7
2	3	4	8
4	5	6	9

تساوی های زیر برقرار است:

$A \quad JOIN \quad B = B \quad JOIN \quad A$

$(A \quad JOIN \quad B) JOIN \quad C = A \quad JOIN \quad (B \quad JOIN \quad C)$

اگر مجموعه عنوان دو رابطه A و B ، اشتراکی نداشته باشند ($H_A \bigcap H_B = \phi$)، آنگاه:

$(A \quad JOIN \quad B) \quad DIVIDE \quad B = A$

اگر $H_A \bigcap H_B = \phi$ ، آنگاه: $A \quad JOIN \quad B = A \quad TIMES \quad B$

اگر $H_A = H_B$ ، آنگاه: $A \quad JOIN \quad B = A \quad INTERSECT \quad B$

اگر رابطه های A وB ، نوع- سازگار باشند، اگر $A \subseteq B$ آنگاه : $A \quad JOIN \quad B = A$

اگر R_i (i=1,2,…n)، n پرتو از رابطه R باشند و با این فرض که صفت پیوند هیچمقدار ندارند:

$R \subseteq R_1 \quad JOIN \quad R_2 \quad JOIN \quad \ldots \quad JOIN \quad R_n$

اگر فرض بالا نباشد، می توان به جای ⊆ از " ⊂ یا ⊃ یا =" نیز استفاده کرد.

فرض می کنیم U وV دو زیر مجموعه از عنوان رابطه R باشند به نحوی که $U \cup V = H_R$. در این صورت: $R \subseteq R[U] \quad JOIN \quad R[V]$

مثال

در چه صورت، تساوی زیر برقرار است؟

$$\Pi_{<L1 \cup L2>}(R \quad JOIN_C \quad S) = \Pi_{<L1>}(R) \quad JOIN_C \quad \Pi_{<L2>}(S)$$

حل: در شرط پیوند C ، فقط زیر مجموعه ای از $L1 \cup L2$ قید شده باشد.

■

مثال

با توجه به سه رابطه زیر، پیوند $R_1 \quad JOIN \quad R_2 \quad JOIN \quad R_3$ روی چه صفاتی انجام می شود؟

(در زیر کلید هر رابطه خط کشیده شده است.)

$$R_1(\underline{A}, C, D) \quad , \quad R_2(\underline{A, B}, E) \quad , \quad R_3(\underline{B}, F, D, G)$$

حل: پیوند روی صفات A,B,D می باشد.

■

مجموعه عملگرهای مبنایی یک مجموعه کامل می باشند و هر عملگر دیگر را می توان برحسب آنها بیان کرد:

$$\{\sigma, \Pi, \cup, -, \times\}$$

مثال

عملگر اشتراک را به کمک عملگر مبنایی بیان کنید.

حل:

$$A \cap B = A - (A - B)$$

البته می توان عملگر اشتراک را به کمک عملگر تفریق و اجتماع نیز بیان کرد:

$$A \cap B = (A \cup B) - [(A - B) \cup (B - A)]$$

■

مثال

عملگر تقسیم را به کمک عملگرهای مبنایی بیان کنید.

حل:

$$A(Y, X) \div B(X) = A[Y] - ((A[Y] \times B) - A)[Y]$$

■

پایگاه داده دانشجو و درس

این پایگاه داده از سه رابطه تشکیل شده است:

1- S (SID , SNAME , SDEG , SMJR , SDEID)

جدول دانشجو(شماره دانشجو،نام دانشجو،سطح دوره تحصیلی،رشته تحصیلی ،شماره گروه آموزشی دانشجو)

2- C(CID , CTITLE , CREDIT , CTYPE , CDEID)

جدول درس(شماره درس ، عنوان درس ، تعداد واحد ، نوع درس،شماره گروه آموزشی ارائه کننده درس)

3- SC(SID,CID ,TR , GRADE)

جدول دانشجو- درس (شماره دانشجو ، شماره درس، ترم ،نمره)

حال با توجه به این سه جدول، به پرس و جوهای زیر پاسخ می دهیم:

۱- شماره درسهایی که دانشجو با شماره 123 آنها را انتخاب کرده است.

$$\Pi_{CID}[\sigma_{SID='123'}(S \infty SC)]$$

۲- نام درسهایی که دانشجو با شماره 123 آنها را انتخاب نکرده است.

$$\Pi_{CTITLE}[[\Pi_{CID}(C)-\Pi_{CID}(\sigma_{SID='123'}SC)] \infty C]$$

۳- نام دانشجویانی که حداقل تمام درسهای انتخاب شده توسط دانشجوی شماره 123 را ، انتخاب کرده باشند.

$$\Pi_{SNAME}[\ [\Pi_{SID,CID}(SC)\div\Pi_{CID}(\sigma_{SID='123'}(SC))] \infty S]$$

۴- نام دانشجویان و نام درسهایی که متعلق به یک گروه آموزش هستند.

$$\Pi_{SNAME,CTITLE}(S \infty C)$$

۵- شماره دانشجویان و شماره درس ها به صورت جفت به طوریکه در هر جفت، دانشجو، درس را انتخاب نکرده باشد.

$$\Pi_{SID,CID}(S\times C)-\Pi_{SID,CID}(SC)$$

۶- نام درسهایی که تمام دانشجویان انتخاب کرده باشند.(وجود کلمه "تمام" ، متناظر با استفاده از عملگر تقسیم می باشد.)

$$\Pi_{CTITLE}[(\Pi_{SID,CID}(SC)\div\Pi_{SID}(S))\infty C]$$

پایگاه داده تهیه کننده و قطعه

پایگاه داده ای تهیه کننده و قطعه دارای سه جدول است:

S (S# , SNAME , STATUS , CITY)
P (P# , PNAME , COLOR , WEIGHT , CITY)
SP (S# , P# , QTY)

در جدول S مشخصات تهیه کنندگان و در جدول P مشخصات قطعات و در جدول SP میـزان تولیـد قطعـات توسـط تهیـه کننده گان نگهداری می شود. مثالهای زیر با توجه به این پایگاه داده حل شده است.

۱- مشخصات تهیه کنندگانی که نام آنها مقدار null باشد.

$$\sigma_{sname=''}(S)$$

۲- شماره قطعاتی که در شهر همدان تولید شده اند.

$$\Pi_{P\#}(\sigma_{CITY='HAMEDAN'}(P))$$

۳- مشخصات قطعاتی که وزن آنها 20 یا 30 باشد.

$$\sigma_{weight=20 \vee weight=30}(P)$$

۴- شماره تهیه کنندگان و قطعات تولیدی با تولید کمتر از ۲۰ عدد.

$$\Pi_{S\#,P\#}(\sigma_{QTY<20}(SP))$$

۵- لیست شهرهای تهیه کنندگان و قطعات

$$\Pi_{CITY}(S) \cup \Pi_{CITY}(P)$$

۶- شهرهایی که هم تهیه کنندگان و هم قطعات در آن ها وجود دارند

$$\Pi_{CITY}(S) \cap \Pi_{CITY}(P)$$

۷- شماره قطعاتی که وزن آنها از 20 کیلو بیشتر است یا توسط عرضه کننده S1 عرضه می شوند.

$$\Pi_{P\#}(\sigma_{WEIGHT>20}(P)) \cup \Pi_{P\#}(\sigma_{S\#='S1'}(SP))$$

۸- شماره تهیه کنندگانی که در شهر تهیهS1 ساکن هستند.

$$\Pi_{s\#}(\Pi_{city,s\#} \cap \Pi_{city,s\#}(\sigma_{s\#='s1'}(S)))$$

۹- نام تهیه کنندگانی که قطعه P1 را تهیه کرده اند.

$$\Pi_{SNAME}(\sigma_{P\#='P1'}(S \infty SP))$$

۱۰- نام عرضه کنندگان قطعه P1 یا P3 .

$$\Pi_{sname}(\sigma_{p\#='p1' \vee p\#='p3'}(S \infty SP))$$

۱۱- نام تهیه کنندگانی که قطعه P2 را تهیه نمی کنند.

$$\Pi_{SNAME}(\Pi_{S\#}(S) - \Pi_{S\#}(\sigma_{P\#='P2'}(SP))) \infty S)$$

۱۲- نام تهیه کنندگانی که حداقل یک قطعه قرمز رنگ تولید می کنند.

$$\Pi_{SNAME}(\sigma_{P.COLOR='RED'}(S \infty P \infty SP))$$

۱۳- نام تهیه کنندگانی که تمام قطعات را تهیه می کنند.

$$\Pi_{SNAME}(S \infty (\Pi_{S\#,P\#}(SP) \div \Pi_{P\#}(P)))$$

عملگر نیم پیوند (SEMIJOIN)

عملگر نیم پیوند(⋉) نوع خاصی از عملگر پیوند طبیعی است که فقط تاپلهای پیوند شدنی از رابطه سمت چپ در رابطه جواب وارد می شوند. کاربرد این عملگر در رابطه های توزیع شده می باشد.

مثال

حاصل SP **SEMIJOIN** S را بدست آورید.

S#	P#	QTY
S1	P1	100
S1	P2	200
S2	P1	100
S3	P2	100

S#	SNAME
S1	Sn1
S2	Sn2
S3	Sn3

حل:

S#	SNAME
S1	SN1
S2	SN2
S3	SN3

■

مثال

عبارت زیر چه چیزی را مشخص می کند.

$$S \ltimes (\prod_{S\#,P\#}(SP) \div \prod_{P\#}(P))$$

حل: مشخصات تهیه کنندگانی که تمام قطعات را عرضه کرده اند.

■

مثال

عبارت زیر چه چیزی را مشخص می کند.

$$S \ltimes (\prod_{S\#}(S) - \prod_{S\#}(\sigma_{P\#='P1'}(SP)))$$

حل: مشخصات تهیه کنندگانی که قطعه P1 را تهیه نکرده اند.

■

مثال

عبارت زیر چه چیزی را مشخص می کند.

$$\sigma_{CITY='HAMEDAN'}(P) \ltimes (\sigma_{QTY<20}(SP))$$

حل: مشخصات قطعاتی که در همدان تهیه شده اند و تعداد آنها از 20 عدد کمتر است.

■

رابطه زیر در صورتی که $A = B$ باشد، برقرار است:

$$A \ \ SEMIJOIN \ \ B = B \ \ SEMIJOIN \ \ A$$

تساوی های زیر برقرار است: (پیوند و نیم پیوند بر روی شرط یکسانی هستند)

$$A \ \ JOIN \ \ B = A \ \ JOIN \ \ (B \ \ SEMIJOIN \ \ A)$$

$$A \ \ JOIN \ \ B = B \ \ JOIN \ \ (A \ \ SEMIJOIN \ \ B)$$

$$A \ \ JOIN \ \ B = (A \ \ SEMIJOIN \ \ B) \ \ JOIN \ \ (B \ \ SEMIJOIN \ \ A)$$

عملگر نیم تفاضل (SEMIMINUS)

این عملگر به صورت زیر تعریف می شود:

A SEMIMINUS B = A MINUS (A SEMIJOIN B)

عملگرهای فرا پیوند (OUTER JOIN)

عملگرهای فرا پیوند عبارتند از:

۱- فرا پیوند چپ (Left Outer Join)

۲- فرا پیوند راست (Right Outer Join)

۳- فرا پیوند کامل (Full Outer Join)

عملگر فرا پیوند چپ معمولاً به شرط تساوی است و علاوه بر تاپلهای پیوند شدنی از دو رابطه، تاپلهای پیوند نشدنی از رابطه چپ هم با مقدار هیچ (null) پیوند می شوند. در حالت فراپیوند راست، تاپلهای پیوند نشدنی از رابطه راست، با مقدار هیچ پیوند می شوند و در حالت فرا پیوند کامل، تاپلهای پیوند نشدنی از رابطه چپ و هم از رابطه راست با مقدار هیچ پیوند می شوند. از علامت ⟗ برای نمایش فراپیوند کامل استفاده می شود.

مثال

دو رابطه R1 و R2 به صورت زیر مفروض است:

R1

A	B
A1	B1
A2	B1
A3	B2
A4	B3
A5	B4

R2

C	D	A
C1	D1	A1
C2	D2	A1
C3	D3	A2
C4	D4	A3
C5	D3	A2
C6	D5	A6

الف- فرا پیوند چپ دو رابطه R1 و R2 .

A	B	C	D
A1	B1	C1	D1
A1	B1	C2	D2
A2	B1	C3	D3
A2	B1	C5	D3
A3	B2	C4	D4
A4	B3	null	Null
A5	B4	null	Null

ب- فرا پیوند راست دو رابطه R1 و R2 .

A	B	C	D
A1	B1	C1	D1
A1	B1	C2	D2
A2	B1	C3	D3
A2	B1	C5	D3
A3	B2	C4	D4
A6	null	C6	D5

ج - فرا پیوند کامل دو رابطه R1 و R2 .

A	B	C	D
A1	B1	C1	D1
A1	B1	C2	D2
A2	B1	C3	D3
A2	B1	C5	D3
A3	B2	C4	D4
A4	B3	null	Null
A5	B4	null	Null
A6	null	C6	D5

عملگر فرا اجتماع (OUTER UNION) : در عملگر اجتماع باید مجموعه عنوان دو رابطه یکسان باشند، اما در فرا اجتماع ، دیگر نیاز به برقراری این شرط نیست و عنوان رابطه حاصل، اجتماع دو عنوان است و از مقدار هیچ (null) بجای مقادیر صفات ناموجود در تاپلها استفاده می شود.

اگر دو رابطه صفت مشترکی نداشته باشند،حاصل FULL OUTER JOIN آنها با OUTER UNION آنها برابر است.

توسط عملگر ρ می توان یک نام جدید برای یک رابطه بدست آورد. به طور نمونه دستور $\rho_x(S)$ برای رابطه S یک نام دیگر به نام x می گذاریم. در دستور زیر جفت شماره تهیه کنندگانی که از یک شهر هستند، مشخص می شود:

$$\Pi_{s.s\#,k.s\#}(\sigma_{s.city=k.city}(\rho_k(S)\infty S))$$

حاصل تقسیم، کاردینالیتی رابطه حاصل از پیوند دو رابطه بر کاردینالیتی رابطه حاصل از ضرب کارتزین دو رابطه را، ضریب گزینش عملگر پیوند گویند: $jsf = \frac{C_{join}}{C_{cart}}$.

(اگر پیوند شرطی نباشد، این ضریب برابر یک است.)

عملگر گروه بندی (SUMMARIZE)

عملگر گروه بندی(γ)، عملگری تک عملوندی برای گروه بندی تاپلهای یک رابطه برحسب مقادیر یک یا بیش از یک صفت و سپس انجام محاسبه روی یکی از صفات توسط یکی از توابع جمعی است.

توابع MUX (حداکثر)، MIN (حداقل)، AVG (میانگین)، SUM(جمع) و COUNT (شمارشگر تاپلها) را توابع جمعی می گویند.

بعد از گروه بندی می توان یک یا چند تابع جمعی را در هر گروه اعمال کرد.

مثال

حاصل دستور زیر بر روی جدول R را بدست آورید.

SUMMARIZE R **BY**(X) **ADD** AVG(Y) **AS** K;

حل:

R

X	Y
x1	5
x1	3
x2	6
x3	1
x3	9

$\Rightarrow$

X	K
x1	4
x2	6
x3	5

مثال

در مثال قبل اگر به جای تابع AVG از تابع MIN استفاده شده بود آنگاه حاصل به صورت زیر خواهد بود:

X	K
x1	3
x2	6
x3	1

مثال

حاصل دستور زیر بر روی جدول R را بدست آورید.

SUMMARIZE R BY(B) ADD SUM(A) AS T;

حل:

R

A	B
10	2
15	3
20	1
35	2
40	3

$\Rightarrow$

B	T
1	20
2	45
3	55

مثال

با فرض وجود رابطه R(A,B,C,D) ، نتیجه اجرای عبارت جبر رابطه ای زیر چیست؟

SUMMARIZE R PER R[ ] ADD SUM (D) AS SD

حل: رابطه ای با یک صفت SD و یک تاپل و با مقدار حاصل جمع مقادیر D .

عملگر SUMMARIZE را به کمک عملگرهای" EXTEND و RENAME و PROJECT " می توان شبیه سازی کرد.

عملگرهای SUMMARIZE ، EXTEND و RESTRICT ، تک عملوندی هستند.

کلید کاندید

رابطه های A و B مفروضند. کلید کاندید رابطه حاصل از اعمال یک عملگر به صورت زیر مشخص می شود.

۱- کلید کاندید $A \quad MINUS \quad B$ ، کلید کاندید A است.

۲- کلید کاندید $A \quad SEMIJOIN \quad B$ ، کلید کاندید A است.

۳- کلید کاندید $A \quad SEMIMINUS \quad B$ ، کلید کاندید A است.

۴- کلید کاندید $A \quad UNION \quad B$ ، عنوان A یا B است.

۵- کلید کاندید $A \quad INTERSECT \quad B$ ، کلید کاندید در A یا B است.

۶- کلید کاندید $A \quad TIMES \quad B$ ، ترکیب کلیدهای کاندید در A و B است.

۷- کلید کاندید $A \quad JOIN \quad B$ ، اجتماع کلیدهای کاندید در A و B است.(صفت پیوند،کلید کاندید در A است)

۸- کلید کاندید $SUMMARIZE \quad A \quad PER(B)$، کلید کاندید B است.

۹- کلید کاندید $\sigma_p(A)$، کلیدهای کاندید A یا زیر مجموعه ای از آن است. به عبارتی اگر K ، کلید کاندید رابطه A و K' کلید کاندید رابطه $\sigma_p(A)$ باشد، در این صورت: $K' \subseteq K$

۱۰- کلید رابطه حاصله از عملگرهای فراپیوند چپ، فراپیوند راست و فرا اجتماع، مشخص نیست.

مثال

رابطه $R(\underline{A, B, C}, D)$ مفروض است. اگر $R_1 = \sigma_{C='1'}(R)$ ،در این صورت کلید کاندید R_1 چیست؟

حل: کلید کاندید $\{A, B\}$ است.

مثال

رابطه $R(\underline{A,B,C},D)$ مفروض است. اگر $R_1 = \sigma_{<A=a \wedge B=b>}(R)$ ،در این صورت کلید کاندید R_1 چیست؟

حل: کلید کاندید C است.

مثال

رابطه R مفروض است. اگر $R_1 = \sigma_{P1}(R)$ و $R_2 = \sigma_{P2}(R)$ ، آنگاه کلید کاندید $R_1 \bigcup R_2$ چیست؟

حل: $H_{R1} \bigcup H_{R2}$

حساب رابطه ای

حساب رابطه ای یکی از روش های اعمال پرس و جو در مدل رابطه ای می باشد که با جبر رابطه ای منطقاً معادل است. یعنی برای هر عبارت جبر رابطه ای، یک عبارت معادل در حساب رابطه ای وجود دارد و برعکس. حساب رابطه ای به زبان طبیعی و جبر رابطه ای به یک زبان برنامه سازی نزدیک است. حساب رابطه ای بر دو نوع تاپلی و میدانی(دامنه ای) می باشد.

در حساب رابطه ای داریم:

EXISTS x(p) ≡ COUNT(x where p=1)

FORALL x(p) ≡ COUNT(x WHERE p)

همچنین داریم:

COUNT (x WHERE p) ≡COUNT (x)

FORALL T(f) ≡NOT EXISTS T (NOT F)

EXISTS T(f) ≡ NOT(FORALL T (NOT f))

NOT EXISTS T(f) ≡ NOT FORALL T(f)

FORALL T((f)) OR (g)) ≡ NOT EXISTS T(NOT (f) AND NOT(g))

مثال

پایگاه داده های "تهیه کننده- قطعه"، به صورت زیر است:

S(S# , SNAME , STATUS, CITY)
P(P# , PNAME , ….)
SP(S# , P# , QTY)

عبارت حساب رابطه ای زیر، پاسخ چه پرسشی است؟

```
S.SX.S#  WHERE  FORALL  SPY(IF  SPY.S#='S3'
                THEN EXISTS SPZ(SPZ.S# =SX.S#
                                AND  SPZ.P# = SPY.P#))
```

(SPY و SPZ دو متغیر تاپلی تعریف شده روی SP هستند.)

حل: شماره تهیه کنندگانی را بدهید که حداقل تمام قطعات تهیه شده توسط 'S3' را تهیه کرده اند. البته اگر تهیه کننده S3 ، قطعه ای تهیه نکرده باشد، آنگاه شماره تمام تهیه کنندگان موجود در رابطه S از جمله خود 'S3' داده می شود.

مثال

رابطه R1 با کلید A ،رابطه R2 با کلید B و رابطه R3 با کلید (A,B) مفروضند. با فرض اینکه W,U,V سه متغیر تاپلی تعریف شده به ترتیب روی R1 وR2 وR3 باشند، معادل عبارت جبر رابطه ای زیر، در حساب رابطه ای را بنویسید.

(R1 TIMES R2) [A,B] MINUS R3[A,B]

حل:

(V.A , V.B) WHERE NOTEXISTS W(W.A=V.A AND W.B=U.B)

در فصل بعد خواهید دید که معادل SQL آن به صورت زیر است:

```
SELECT  R1.A ,R2.B  FROM R1,R2
EXCEPT
SELECT  R3.A,R3.B  FROM  R3;
```

مثال

عبارت زیر به صورت حساب رابطه ای است. معادل آن را در جبر رابطه ای بنویسید.

SUM (V WHERE V.A='a1' AND V.B='b1' , C) AS SC

حل:

(SUMMARIZE R PER R{TUPLE<'a1' , 'b1'>} ADD SUM(C) AS SC) {SC}

در فصل بعد خواهید دید که معادل SQL آن به صورت زیر است:

SELECT SUM(R.C) AS SC FROM R WHERE R.A='a1' AND R.B='b1'

حساب رابطه ای، نارویه ای تر از جبر رابطه ای است.

رابطه مبنا با یک عبارت جبر رابطه ای تعریف نمی شود.

آزمون

۱- رابطه های R(A,B) و S(A,C,D) زیر مفروض اند. S ⟗ R چند تاپل خواهد داشت؟(دولتی ۹۰)

(⟗ : فراپیوند طبیعی کامل--Natural full outer join)

R

A	B
1	2
3	4

S

A	C	D
1	2	3
1	3	4
2	4	5

۱) 2 ۲) 3 ۳) 4 ۴) 5

۳-۱) در حالت فرا پیوند کامل، علاوه بر تاپلهای پیوند شدنی از دو رابطه، تاپلهای پیوند نشدنی هم از رابطه چپ و هم از رابطه راست با مقدار هیچ پیوند می شوند.

A	B	C	D
1	2	2	3
1	2	3	4
3	4	null	null
2	null	4	5

۲- رابطه R(a,b,c) و دو عبارت جبری زیر را در نظر بگیرید. کدام عبارت همواره تعریف شده است؟(دولتی ۸۹)

$Q_1 : \pi_{b,c}(\sigma_{b=c}(R))$

$Q_2 : \pi_{a,b}(\sigma_{a=b}(R))$

۱) $Q_1 \div Q_2$ ۲) $Q_1 \rhd\lhd Q_2$ ۳) $Q_1 \cap Q_2$ ۴) موارد ۱و۲

۲-۲) نتیجه پرسش Q_1، جدولی است با دو ستون b و c و نتیجه پرسش Q_2، جدولی است با دو ستون a و b. بنابراین چون ستونها دو جدول هم نام نمی باشند، اشتراک آنها تعریف نشده است. همچنین چون ستونهای Q_2 زیر مجموعه Q_1 نمی باشند، $Q_1 \div Q_2$ تعریف نشده است.

۳- اگر A و B دو رابطه دارای خصیصه های(Attributes) یکسان باشند، حاصل الحاق A و B کدام یک از عبارات زیر است؟ (دولتی ۸۷)

۱) $A \times B$ ۲) $A \cap B$ ۳) $A \cup B$ ۴) هیچ کدام

۲-۳) چون تمام خصیصه های دو رابطه یکسان است، حاصل الحاق آنها با اشتراک آنها برابر خواهد بود.

منتخبی از عناوین آموزشی منتشر شده بر روی فرادرس

برنامه نویسی	
عنوان آموزش	مدت زمان تقریبی
آموزش اصول و مبانی برنامه نویسی – کلیک کنید (+)	۲ ساعت
آموزش برنامه نویسی جاوا – کلیک کنید (+)	۲۳ ساعت
آموزش برنامه نویسی PHP – کلیک کنید (+)	۲۹ ساعت
آموزش برنامه نویسی C++ – کلیک کنید (+)	۲۰ ساعت
آموزش پیشرفته C++ (شی گرایی در سی پلاس پلاس) – کلیک کنید (+)	۸ ساعت
آموزش مقدماتی برنامه نویسی سی شارپ (C#) – کلیک کنید (+)	۸ ساعت
مجموعه آموزش های کاربردی برنامه نویسی C# (سی شارپ) – کلیک کنید (+)	۱۴ ساعت
آموزش شی گرایی در سی شارپ (C#) – کلیک کنید (+)	۱۴ ساعت
آموزش برنامه نویسی پایتون – مقدماتی – کلیک کنید (+)	۱۸ ساعت
آموزش تکمیلی برنامه نویسی پایتون – کلیک کنید (+)	۵ ساعت
آموزش برنامه نویسی C – کلیک کنید (+)	۱۳ ساعت
آموزش SQL Server – مقدماتی – کلیک کنید (+)	۱۱ ساعت
آموزش SQL Server – تکمیلی – کلیک کنید (+)	۶ ساعت
آموزش کار با دستورات پایگاه داده در SQL Server – کلیک کنید (+)	۳ ساعت
مجموعه آموزش های برنامه نویسی متلب (MATLAB) – کلیک کنید (+)	۹ ساعت
مجموعه آموزش های برنامه نویسی متلب پیشرفته – کلیک کنید (+)	۷ ساعت
آموزش برنامه نویسی اندروید (Android) – مقدماتی – کلیک کنید (+)	۱۶ ساعت
آموزش برنامه نویسی اندروید (Android) – تکمیلی – کلیک کنید (+)	۲۰ ساعت
آموزش برنامه نویسی اندروید (Android) – پیشرفته – کلیک کنید (+)	۲۶ ساعت
آموزش پروژه محور برنامه نویسی دلفی (Delphi) – کلیک کنید (+)	۳ ساعت
آموزش زبان برنامه نویسی فرترن (Fortran) – مقدماتی – کلیک کنید (+)	۴ ساعت
آموزش مروری زبان برنامه نویسی FORTRAN با مثال های کاربردی – کلیک کنید (+)	۲ ساعت
آموزش پروژه محور اینترنت اشیا (IoT) - کنترل لوازم منزل با شبکه و پیامک و ماژول هـای SIM و ESP8266 – کلیک کنید (+)	۱۰ ساعت
آموزش برنامه نویسی Swift (سوئیفت) برای برنامه نویسی iOS – کلیک کنید (+)	۵ ساعت
آموزش برنامه نویسی پاسکال (Pascal) – کلیک کنید (+)	۶ ساعت
آموزش برنامه نویسی برای دانش آموزان با اسمال بیسیک (Small Basic) – کلیک کنید (+)	۱۶ ساعت
آموزش برنامه نویسی تصویری به کودکان با زبان اسکرچ (Scratch) – کلیک کنید (+)	۶ ساعت
آموزش محافظت از کدهای نرم افزاری با SmartAssembly (اسمارت اسمبلی) – کلیک کنید (+)	۱ ساعت

برنامه نویسی (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش مبانی برنامه نویسی (الگوریتم و فلوچارت) با رویکرد حل مسأله – کلیک کنید (+)	۱۰ ساعت
آموزش پروژه محور Python (پایتون) - ساخت نرم افزار برای Windows و Linux – کلیک کنید (+)	۱۰ ساعت
آموزش برنامه نویسی آردوینو (Arduino) با پایتون (Python) – کلیک کنید (+)	۵ ساعت
آموزش ساخت اپلیکیشن اندروید (Android) با PHP و MySQL – کلیک کنید (+)	۲ ساعت
آموزش مقدماتی زبان برنامه نویسی کاتلین (Kotlin) برای توسعه اندروید (Android) – کلیک کنید (+)	۳ ساعت
آموزش داکر (Docker) برای توسعه دهندگان – کلیک کنید (+)	۱ ساعت
آموزش توسعه وب با زبان برنامه نویسی Go – کلیک کنید (+)	۱ ساعت
آموزش ویژوال بیسیک دات نت (Visual Basic.NET) – مقدماتی – کلیک کنید (+)	۱۳ ساعت
آموزش ویژوال بیسیک دات نت (Visual Basic.NET) – تکمیلی – کلیک کنید (+)	۱۶ ساعت
آموزش تاریخچه گیت (Git) و مقدمات آن – کلیک کنید (+)	۱ ساعت
آموزش گیت (Git) برای مدیریت نسخه توزیع شده – کلیک کنید (+)	۱ ساعت
آموزش زبان برنامه نویسی AWK در لینوکس - پردازش و آنالیز فایل های متنی – کلیک کنید (+)	۵ ساعت
آموزش برنامه نویسی R و نرم افزار RStudio – کلیک کنید (+)	۷ ساعت
آموزش تکمیلی برنامه نویسی R و نرم افزار RStudio – کلیک کنید (+)	۵ ساعت
آموزش مقدماتی ساخت ربات تلگرام با PHP – کلیک کنید (+)	۱ ساعت
آموزش کتابخانه قالب استاندارد (STL) در C++ برای ساده نویسی برنامه ها – کلیک کنید (+)	۹ ساعت
آموزش برنامه نویسی با روش سه لایه به زبان VB.Net – کلیک کنید (+)	۴ ساعت
آموزش بازی سازی در پایتون با کتابخانه Pygame – کلیک کنید (+)	۴ ساعت
آموزش پایتون گرافیکی (رابط های گرافیکی پایتون) – کلیک کنید (+)	۵ ساعت
آموزش ساخت ربات تلگرام با پایتون (Python) – کلیک کنید (+)	۱ ساعت
مجموعه آموزش های طراحی رابط های گرافیکی (GUI) در متلب	۷ ساعت
آموزش تکمیلی طراحی رابط گرافیکی کاربر (GUI) با متلب (MATLAB) – کلیک کنید (+)	۱۴ ساعت
آموزش ساخت ربات تلگرام با متلب (MATLAB) – کلیک کنید (+)	۱ ساعت
آموزش ساخت ربات تلگرام با جاوا (Java) – کلیک کنید (+)	۱ ساعت
آموزش پایگاه داده ها در جاوا – کلیک کنید (+)	۲ ساعت
آموزش کار با فایل ها در جاوا با پکیج Java/IO (به همراه پروژه دفترچه یادداشت) – کلیک کنید (+)	۳ ساعت
آموزش پروژه محور جاوا - تولید QR code و ایجاد و اسکن بارکد (Barcode) – کلیک کنید (+)	۱ ساعت

برنامه نویسی (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش پروژه محور جاوا (Java) - طراحی و ساخت شبکه اجتماعی – کلیک کنید (+)	۶ ساعت
آموزش طراحی رابط کاربری مدرن در .NET با کامپوننت های جانوس (Janus) – کلیک کنید (+)	۴ ساعت
آموزش ساخت بازی ساده در ویژوال بیسیک – کلیک کنید (+)	۲ ساعت
آموزش پیاده سازی سیستم انبارداری و فروش با ویژوال بیسیک دات نت (VB.NET) – مقدماتی – کلیک کنید (+)	۱۰ ساعت
آموزش پیاده سازی سیستم انبارداری و فروش با ویژوال بیسیک دات نت (VB.NET) – تکمیلی – کلیک کنید (+)	۱۲ ساعت
آموزش مقدماتی پیاده سازی مسائل بهینه سازی در پایتون (Python) – کلیک کنید (+)	۴ ساعت
آموزش الگوهای طراحی (Design Patterns) در پایتون (Python) – کلیک کنید (+)	۱۲ ساعت
آموزش رایگان نکات و ترفندهای متلب – کلیک کنید (+)	۲ ساعت
مجموعه آموزش های برنامه نویسی متلب برای علوم و مهندسی – کلیک کنید (+)	۱۴ ساعت
آموزش فریم ورک Spring در جاوا – کلیک کنید (+)	۶ ساعت
آموزش ORM هایبرنیت (Hibernate) جاوا (Java) – کلیک کنید (+)	۱۸ ساعت
آموزش آشنایی با LINQ to SQL در C# – کلیک کنید (+)	۲ ساعت
آموزش برنامه نویسی تحت شبکه با سی شارپ در قالب پروژه – کلیک کنید (+)	۱ ساعت
آموزش برنامه نویسی با روش سه لایه به زبان سی شارپ – کلیک کنید (+)	۴ ساعت
آموزش Cryptography در دات نت – کلیک کنید (+)	۳ ساعت
آموزش قفل نرم افزاری در C# از طریق رجیستری – کلیک کنید (+)	۴ ساعت
آموزش پروژه محور سی شارپ (C#) – سیستم مدیریت آموزشگاه – کلیک کنید (+)	۱۹ ساعت
آموزش پروژه محور سی شارپ (C#) – سیستم مدیریت حسابداری و انبارداری – کلیک کنید (+)	۱۴ ساعت
آموزش برنامه نویسی سورس کدهای الگوریتمی در سی شارپ – کلیک کنید (+)	۷ ساعت
آموزش پروژه محور سی شارپ (C#) – پیاده سازی سیستم هتلداری – کلیک کنید (+)	۱۰ ساعت
آموزش ساخت سیستم آزمون گیر در سی شارپ – کلیک کنید (+)	۴ ساعت
آموزش ارسال و دریافت پیامک با مودم در سی شارپ (C#) – کلیک کنید (+)	۳ ساعت
آموزش پروژه محور سی شارپ (C#) – پیاده سازی سیستم مدیریت آتلیه – کلیک کنید (+)	۱۱ ساعت
آموزش پروژه محور سی شارپ (C#) – پیاده سازی سیستم نوبت دهی – کلیک کنید (+)	۷ ساعت
آموزش پروژه محور سی شارپ (C#) – پیاده سازی سیستم صدور بلیط – کلیک کنید (+)	۹ ساعت
آموزش رسم نمودار در سی شارپ (C#) – کلیک کنید (+)	۳ ساعت
آموزش پروژه محور سی شارپ (C#) – سیستم مدیریت بانک – کلیک کنید (+)	۷ ساعت
آموزش ساخت بازی با سی شارپ (C#) – کلیک کنید (+)	۵ ساعت
آموزش Metaprogramming (برنامه نویسی متا) در پایتون – کلیک کنید (+)	۲ ساعت

برنامه نویسی (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش کاربردی Entity Framework در سی شارپ (C#) – کلیک کنید (+)	۱۲ ساعت
آموزش پروژه محور سی شارپ (C#) – پیاده سازی سیستم مدیریت کارمندان – کلیک کنید (+)	۶ ساعت
آموزش پروژه محور سی شارپ (C#) – پیاده سازی سیستم مدیریت چک های بانکی – کلیک کنید (+)	۶ ساعت
آموزش پروژه محور سی شارپ (C#) – شبیه سازی دستگاه عابر بانک – کلیک کنید (+)	۴ ساعت
آموزش گزارش گیری با کریستال ریپورت و استیمول سافت (Stimulsoft) در سی شارپ (C#) – کلیک کنید (+)	۲ ساعت
آموزش پایگاه داده اس کیو لایت (SQLite) در سی شارپ (C#) – کلیک کنید (+)	۲ ساعت
آموزش پروژه محور سی شارپ (C#) - پیاده سازی نرم افزار حسابداری شخصی – کلیک کنید (+)	۶ ساعت
آموزش پروژه محور سی شارپ (C#) – پیاده سازی سیستم رستوران آنلاین – کلیک کنید (+)	۱۵ ساعت
آموزش کار با بانک اطلاعاتی اکسس (Access) در سی شارپ (C#) – کلیک کنید (+)	۳ ساعت
آموزش کار با فایل ها و دایرکتوری در C# (سی شارپ) – کلیک کنید (+)	۲ ساعت
آموزش پروژه محور سی شارپ C# – پیاده سازی نرم افزار FeedReader (فیدریدر) – کلیک کنید (+)	۳ ساعت
آموزش کار با دستورات گرافیکی در سی شارپ (C#) – کلیک کنید (+)	۴ ساعت
آموزش مقدماتی WPF در سی شارپ (C#) برای ساخت و اجرای اینترفیس – کلیک کنید (+)	۳ ساعت
آموزش WPF در سی شارپ برای ساخت و اجرای اینترفیس – پیشرفته – کلیک کنید (+)	۴ ساعت
آموزش WPF در سی شارپ (C#) برای ساخت و اجرای اینترفیس – تکمیلی – کلیک کنید (+)	۲ ساعت
آموزش ساخت ربات تلگرام با سی شارپ (C#) – کلیک کنید (+)	۱ ساعت
آموزش پروژه محور پایگاه داده LocalDB در سی شارپ (C#) - سیستم مدیریت کارمندان – کلیک کنید (+)	۱۲ ساعت
آموزش کتابخانه Spire.Doc در سی شارپ (C#) برای ایجاد و ویرایش فایل Word – کلیک کنید (+)	۵ ساعت
آموزش پروژه محور سی شارپ (C#) - ساخت برنامه دانلودگر – کلیک کنید (+)	۱ ساعت

علوم کامپیوتر	
عنوان آموزش	مدت زمان تقریبی
آموزش ساختمان داده ها – کلیک کنید (+)	۱۰ ساعت
آموزش پیشرفته ساختمان داده (همراه با حل نمونه سوالات کنکور ارشد و دکتری) – کلیک کنید (+)	۸ ساعت
آموزش ساختمان داده ها (مرور – تست کنکور ارشد) – کلیک کنید (+)	۲۰ ساعت
آموزش سیستم های عامل – کلیک کنید (+)	۱۱ ساعت
آموزش سیستم عامل (مرور - تست کنکور ارشد) – کلیک کنید (+)	۱۲ ساعت
آموزش پایگاه داده ها – کلیک کنید (+)	۸ ساعت
آموزش پایگاه داده ها (مرور – تست کنکور ارشد) – کلیک کنید (+)	۸ ساعت
آموزش طراحی و پیاده سازی زبان های برنامه سازی – کلیک کنید (+)	۱۰ ساعت
آموزش طراحی و پیاده سازی زبان های برنامه سازی (مرور – تست کنکور ارشد) – کلیک کنید (+)	۹ ساعت
آموزش طراحی الگوریتم – کلیک کنید (+)	۱۵ ساعت
آموزش طراحی الگوریتم به همراه حل مثال های عملی – کلیک کنید (+)	۸ ساعت
آموزش روش های حل روابط بازگشتی – کلیک کنید (+)	۴ ساعت
آموزش رابطه های بازگشتی در طراحی الگوریتم و ساختمان گسسته (مرور - تست کنکور ارشد) – کلیک کنید (+)	۲ ساعت
آموزش طراحی الگوریتم (مرور – تست کنکور ارشد) – کلیک کنید (+)	۱۵ ساعت
آموزش روش تقسیم و حل در طراحی الگوریتم (مرور - تست کنکور کارشناسی ارشد) – کلیک کنید (+)	۲ ساعت
آموزش مبانی منطق و نظریه مجموعه ها – کلیک کنید (+)	۱۸ ساعت
آموزش مدار منطقی (مرور – تست کنکور ارشد) – کلیک کنید (+)	۱۵ ساعت
آموزش مبانی الکترونیک دیجیتال - بخش اول – کلیک کنید (+)	۱۲ ساعت
آموزش مبانی الکترونیک دیجیتال - بخش دوم – کلیک کنید (+)	۱۱ ساعت
آموزش معماری کامپیوتر با رویکرد حل مساله - مقدماتی– کلیک کنید (+)	۲۰ ساعت
آموزش معماری کامپیوتر با رویکرد حل مساله – پیشرفته – کلیک کنید (+)	۱۴ ساعت
آموزش نظریه زبان ها و ماشین ها – کلیک کنید (+)	۹ ساعت
آموزش نظریه زبان ها و ماشین (مرور - تست کنکور ارشد) – کلیک کنید (+)	۸ ساعت
آموزش شبیه سازی و تست نظریه زبان ها و ماشین در JFLAP – کلیک کنید (+)	۳ ساعت
آموزش یادگیری عمیق در پزشکی – کلیک کنید (+)	۴ ساعت
آموزش ساختمان گسسته با رویکرد حل مساله – کلیک کنید (+)	۱۶ ساعت
آموزش ساختمان گسسته (مرور و حل تست های کنکور کارشناسی ارشد) – کلیک کنید (+)	۱۲ ساعت
آموزش آمار و احتمال مهندسی – کلیک کنید (+)	۱۳ ساعت
آموزش آمار و احتمال مهندسی (حل تمرین و تست کنکور ارشد) – کلیک کنید (+)	۱۴ ساعت

علوم کامپیوتر (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش شبکه های کامپیوتری ۱ – کلیک کنید (+)	۱۹ ساعت
آموزش شبکه های کامپیوتری ۲ - همراه با حل مسأله – کلیک کنید (+)	۲۳ ساعت
آموزش شبیه سازی شبکه های مبتنی بر سیسکو (Cisco) در نرم افزار Boson NetSim – کلیک کنید (+)	۱ ساعت
آموزش طراحی کامپایلر – کلیک کنید (+)	۱۵ ساعت
آموزش طراحی کامپایلر (مرور و حل تست های کنکور کارشناسی ارشد) – کلیک کنید (+)	۱۵ ساعت
آموزش ذخیره و بازیابی اطلاعات – کلیک کنید (+)	۸ ساعت
آموزش ساختمان داده ها همراه با پیاده سازی در C++ – کلیک کنید (+)	۲۳ ساعت
آموزش انتقال داده ها در شبکه های کامپیوتری و اینترنت به همراه حل مسائل – کلیک کنید (+)	۹ ساعت
آموزش مقدماتی Hadoop (هدوپ) برای تجزیه و تحلیل کلان داده – کلیک کنید (+)	۵ ساعت
آموزش نظریه صف (Queueing theory) – کلیک کنید (+)	۲۲ ساعت
آموزش گرافیک کامپیوتری – کلیک کنید (+)	۴ ساعت
آموزش گرافیک کامپیوتری با OpenGL – کلیک کنید (+)	۱۶ ساعت
آموزش مهندسی نرم افزار ۱ – کلیک کنید (+)	۱۲ ساعت
آموزش زبان تخصصی مهندسی کامپیوتر – کلیک کنید (+)	۱۰ ساعت
آموزش مروری بر پیچیدگی محاسبات (Computational Complexity) – کلیک کنید (+)	۱ ساعت
آموزش تبدیل فلوچارت به کد با Flowgorithm – کلیک کنید (+)	۱ ساعت
آموزش نرم افزار RAPTOR برای ترسیم فلوچارت – کلیک کنید (+)	۱ ساعت
آموزش حل سوالات کنکور کارشناسی ارشد کامپیوتر – ۹۴ – کلیک کنید (+)	۲ ساعت
آموزش مرور و حل تست های کنکور کارشناسی ارشد معماری کامپیوتر – کلیک کنید (+)	۱۷ ساعت
آموزش برنامه نویسی اسمبلی – کلیک کنید (+)	۱۰ ساعت
آموزش نظریه گراف و کاربردها – کلیک کنید (+)	۱۵ ساعت
آموزش ترکیبیات و کاربردها – کلیک کنید (+)	۱۵ ساعت
آموزش نظریه بازی ها و کاربرد آن در اقتصاد – کلیک کنید (+)	۸ ساعت
آموزش مدل سازی UML با نرم افزار Rational Rose – کلیک کنید (+)	۴ ساعت
آموزش الگوریتم موازی و پردازش موازی – کلیک کنید (+)	۱۳ ساعت
آموزش ترکیب اطلاعات (Information Fusion) – کلیک کنید (+)	۲۰ ساعت
آموزش نهان نگاری دیجیتال (Digital watermarking) – کلیک کنید (+)	۳ ساعت
آموزش نهان نگاری و پیاده سازی آن در سی شارپ (C#) – کلیک کنید (+)	۳ ساعت
آموزش واترمارکینگ در نهان نگاری با سی شارپ (C#) – کلیک کنید (+)	۱ ساعت
آموزش شبکه های پیچیده پویا (Complex Dynamical Networks) – کلیک کنید (+)	۱ ساعت

علوم کامپیوتر (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش تخمین تلاش لازم برای توسعه نرم افزارها با متلب – کلیک کنید (+)	۳ ساعت
آموزش توسعه نرم افزار با متد ICONIX و زبان مدل سازی UML– کلیک کنید (+)	۴ ساعت
آموزش نگارش پروپوزال، پایان نامه و ارائه - ویژه پردازش تصویر و بینایی کامپیوتر – کلیک کنید (+)	۱ ساعت

هوش مصنوعی	
عنوان آموزش	مدت زمان تقریبی
آموزش هوش مصنوعی – کلیک کنید (+)	۴ ساعت
آموزش هوش مصنوعی – تکمیلی – کلیک کنید (+)	۱۹ ساعت
آموزش هوش مصنوعی توزیع شده – کلیک کنید (+)	۲۴ ساعت
آموزش هوش مصنوعی (مرور و حل تست کنکور ارشد) – کلیک کنید (+)	۱۵ ساعت
مجموعه آموزش های شبکه های عصبی مصنوعی در متلب – کلیک کنید (+)	۲۸ ساعت
مجموعه آموزش های کاربردی شبکه های عصبی مصنوعی – کلیک کنید (+)	۱۰ ساعت
آموزش طراحی سیستم های فازی عصبی یا ANFIS با استفاده از الگوریتم های فرابتکاری و تکاملی – کلیک کنید (+)	۲ ساعت
آموزش کاربردی مکاترونیک - محاسبات پایه طراحی درب های خودکار – کلیک کنید (+)	۵ ساعت
آموزش شبکه عصبی GMDH به همراه پیاده سازی عملی در متلب – کلیک کنید (+)	۴ ساعت
آموزش شبکه های عصبی گازی به همراه پیاده سازی عملی در متلب – کلیک کنید (+)	۳ ساعت
آموزش طبقه بندی و بازشناسی الگو با شبکه های عصبی LVQ در متلب – کلیک کنید (+)	۳ ساعت
آموزش یادگیری عمیق (Deep learning) – کلیک کنید (+)	۳ ساعت
آموزش برنامه نویسی یادگیری عمیق با پایتون (TensorFlow و Keras) – کلیک کنید (+)	۳ ساعت
مجموعه آموزش های سیستم های فازی در متلب – کلیک کنید (+)	۲۱ ساعت
مجموعه آموزش های تئوری و عملی الگوریتم ژنتیک – کلیک کنید (+)	۱۴ ساعت
آموزش پیاده سازی ترکیب الگوریتم ژنتیک و PSO در متلب – کلیک کنید (+)	۱ ساعت
مجموعه آموزش های بهینه سازی چند هدفه در متلب – کلیک کنید (+)	۱۹ ساعت
آموزش حل مساله فروشنده دوره گرد با استفاده از الگوریتم ژنتیک – کلیک کنید (+)	۲ ساعت
آموزش انتخاب ویژگی یا Feature Selection – کلیک کنید (+)	۲ ساعت
آموزش انتخاب ویژگی با استفاده از الگوریتم های فرابتکاری و تکاملی – کلیک کنید (+)	۴ ساعت
آموزش تخمین خطای طبقه بندی یا Classifier Error Estimation – کلیک کنید (+)	۲ ساعت
آموزش کاهش تعداد رنگ تصاویر با استفاده از روش های خوشه بندی هوشمند – کلیک کنید (+)	۱ ساعت

مهندسی برق	
عنوان آموزش	مدت زمان تقریبی
آموزش الکترونیک ۱ – کلیک کنید (+)	۱۱ ساعت
آموزش حل تمرین الکترونیک ۱ – کلیک کنید (+)	۲۵ ساعت
آموزش الکترونیک ۱ (مرور و حل تست) – کلیک کنید (+)	۱۱ ساعت
آموزش الکترونیک ۲ – کلیک کنید (+)	۸ ساعت
آموزش حل تمرین الکترونیک ۲ – کلیک کنید (+)	۲۵ ساعت
آموزش الکترونیک ۲ (مرور و حل تست) – کلیک کنید (+)	۱۲ ساعت
آموزش الکترونیک ۳ – کلیک کنید (+)	۱۵ ساعت
آموزش الکترونیک ۳ (مرور و حل مساله) – کلیک کنید (+)	۱۲ ساعت
آموزش مبانی مهندسی برق ۱ – کلیک کنید (+)	۱۴ ساعت
آموزش مبانی مهندسی برق ۱ (مرور و حل مساله) – کلیک کنید (+)	۱۴ ساعت
آموزش مبانی مهندسی برق ۲ – کلیک کنید (+)	۱۶ ساعت
آموزش مدارهای الکتریکی ۱ – کلیک کنید (+)	۲۴ ساعت
آموزش حل تمرین مدارهای الکتریکی ۱ – کلیک کنید (+)	۱۷ ساعت
آموزش مدارهای الکتریکی ۱ (مرور و حل تست) – کلیک کنید (+)	۱۴ ساعت
آموزش مدارهای الکتریکی ۲ – کلیک کنید (+)	۱۱ ساعت
آموزش مدارهای الکتریکی ۲ (حل تمرین) – کلیک کنید (+)	۱۸ ساعت
آموزش مدارهای الکتریکی ۲ (مرور و حل تست) – کلیک کنید (+)	۱۳ ساعت
آموزش هندسه فینسلر (Finsler Geometry) – کلیک کنید (+)	۹ ساعت
آموزش زبان تخصصی مهندسی برق (همراه با حل نمونه سوالات کنکور ارشد) – کلیک کنید (+)	۱۶ ساعت
آموزش مروری ماشین های الکتریکی ۱ – کلیک کنید (+)	۲ ساعت
آموزش ماشین های الکتریکی ۱ – کلیک کنید (+)	۱۴ ساعت
آموزش ماشین های الکتریکی ۱ (مرور و حل تست) – کلیک کنید (+)	۱۶ ساعت
آموزش ماشین های الکتریکی ۲ – کلیک کنید (+)	۱۷ ساعت
آموزش ماشین های الکتریکی ۲ (مرور و حل تست کنکور ارشد) – کلیک کنید (+)	۱۳ ساعت
آموزش ماشین های الکتریکی ۳ – کلیک کنید (+)	۱۱ ساعت
آموزش مدارهای مخابراتی – کلیک کنید (+)	۹ ساعت
آموزش ماشین های الکتریکی مخصوص – کلیک کنید (+)	۹ ساعت
آموزش کنترل موتورهای الکتریکی صنعتی ۱ – کلیک کنید (+)	۱۴ ساعت
آموزش کنترل موتورهای الکتریکی صنعتی ۲ – کلیک کنید (+)	۱۷ ساعت
آموزش کنترل صنعتی - حل مساله – کلیک کنید (+)	۱۵ ساعت

مهندسی برق (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
مجموعه آموزش های مدارهای منطقی (طراحی دیجیتال) – کلیک کنید (+)	۹ ساعت
آموزش مبانی الکترونیک دیجیتال - بخش اول – کلیک کنید (+)	۱۲ ساعت
آموزش مبانی الکترونیک دیجیتال - بخش دوم – کلیک کنید (+)	۱۱ ساعت
آموزش الکترومغناطیس مهندسی – کلیک کنید (+)	۲۰ ساعت
آموزش الکترومغناطیس مهندسی (مرور و حل تست کنکور ارشد) – کلیک کنید (+)	۱۸ ساعت
آموزش الکترومغناطیس ۱ – کلیک کنید (+)	۹ ساعت
آموزش مقدماتی شبکه های هوشمند (Smart grid) انرژی الکتریکی – کلیک کنید (+)	۳ ساعت
آموزش کاربردی مکاترونیک - محاسبات پایه طراحی درب های خودکار – کلیک کنید (+)	۵ ساعت
آموزش کامسول مولتی فیزیکس - COMSOL Multiphysics (مباحث منتخب) – کلیک کنید (+)	۲ ساعت
آموزش نرم افزار COMSOL Multiphysics برای پدیده های الکترومغناطیس – کلیک کنید (+)	۴ ساعت
آموزش شبیه سازی مدولاسیون دیجیتال در متلب (MATLAB) – کلیک کنید (+)	۸ ساعت
آموزش نرم افزار سیلواکو (SILVACO) برای شبیه سازی افزاره های نیمه رسانا – کلیک کنید (+)	۴ ساعت

ابزار مدیریت وبسایت	
عنوان آموزش	مدت زمان تقریبی
آموزش کار با cPanel – کلیک کنید (+)	۱ ساعت
آموزش کار با کنترل پنل plesk برای ارائه خدمات وب – کلیک کنید (+)	۱ ساعت
آموزش مدیریت هاست با دایرکت ادمین (DirectAdmin) – کلیک کنید (+)	۱ ساعت
آموزش جوملا (Joomla) برای راه اندازی و مدیریت وب سایت – کلیک کنید (+)	۳ ساعت
آموزش نرم افزار Mobirise برای ساخت وب سایت بدون کدنویسی – کلیک کنید (+)	۱ ساعت
آموزش مقدماتی Oracle APEX برای تولید و توسعه نرم افزارهای تحت وب – کلیک کنید (+)	۸ ساعت
آموزش وردپرس (WordPress) – مقدماتی – کلیک کنید (+)	۱۱ ساعت
مجموعه آموزش های راه اندازی سایت و کار با وردپرس – تکمیلی – کلیک کنید (+)	۶ ساعت
آموزش گوگل آنالیتیکس (Google Analytics) برای تحلیل آمار وب سایت – کلیک کنید (+)	۳ ساعت
آموزش آنالیز و بهینه سازی سایت با GTmetrix – کلیک کنید (+)	۲ ساعت
آموزش تست نفوذ در وب و راه های مقابله با آن – کلیک کنید (+)	۱۰ ساعت

طراحی و گرافیک	
عنوان آموزش	مدت زمان تقریبی
آموزش فتوشاپ (Photoshop) – مقدماتی – کلیک کنید (+)	۱۲ ساعت
مجموعه آموزش های کاربردی طراحی و گرافیک با نرم افزار کورل (CorelDRAW) – کلیک کنید (+)	۱۴ ساعت
آموزش کورل پیشرفته – کلیک کنید (+)	۱۰ ساعت
آموزش نرم افزار طراحی گرافیکی Adobe Illustrator – کلیک کنید (+)	۱۰ ساعت
آموزش نرم افزار طراحی گرافیکی Adobe Illustrator – تکمیلی – کلیک کنید (+)	۵ ساعت
آموزش مقایسه نرم افزارهای گرافیکی Photoshop , CorelDRAW , Illustrator – کلیک کنید (+)	۱ ساعت
آموزش مقدماتی طراحی کاراکتر در Adobe Illustrator (کاراکتر پلیس) – کلیک کنید (+)	۲ ساعت
آموزش نرم افزار Blender برای طراحی و مدل سازی سه بعدی – کلیک کنید (+)	۸ ساعت
آموزش مدل سازی کاراکتر در بلندر - ایده پردازی تا ریگ – کلیک کنید (+)	۱۴ ساعت
آموزش طراحی فلت (Flat design) با کورل (CorelDRAW) – کلیک کنید (+)	۱ ساعت
آموزش فتوشاپ برای استفاده در صنعت چاپ و تبلیغات – کلیک کنید (+)	۳ ساعت
آموزش تایپوگرافی (Typography) - سایه زدن حروف در فتوشاپ (Photoshop) – کلیک کنید (+)	۳ ساعت
آموزش پروژه محور فتوشاپ (Photoshop) - ساخت ماسک (Mask) فانتزی – کلیک کنید (+)	۱ ساعت
آموزش آشنایی با پنل Adjustments در فتوشاپ – کلیک کنید (+)	۲ ساعت
آموزش ساخت جلوه های ویژه در فتوشاپ – کلیک کنید (+)	۳ ساعت
آموزش Adobe Lightroom (ادوبی لایت روم) برای ویرایش و روتوش تصاویر – کلیک کنید (+)	۶ ساعت
آموزش Adobe Dimension برای ایجاد تصاویر سه بعدی محصولات و تبلیغات – کلیک کنید (+)	۲ ساعت
آموزش Adobe Bridge (ادوبی بریج) برای مدیریت فایل های چندرسانه ای – کلیک کنید (+)	۲ ساعت
آموزش ایندیزاین (InDesign) برای چاپ و صفحه آرایی کتاب و مجلات – کلیک کنید (+)	۴ ساعت
آموزش نرم افزار SketchBook Pro (اسکچ بوک) برای طراحی آناتومی حیوانات – کلیک کنید (+)	۲۲ ساعت
آموزش طراحی مفهومی (Conceptual) خودرو با SketchBook – کلیک کنید (+)	۳ ساعت
آموزش نقاشی سه بعدی با نرم افزار Paint 3D ویندوز – کلیک کنید (+)	۳ ساعت
آموزش مبانی هنرهای تجسمی ۱ – کلیک کنید (+)	۳ ساعت
آموزش اصول طراحی نشانه (Sign Design) – کلیک کنید (+)	۱ ساعت
آموزش نقاشی دیجیتال با Photoshop و Mischief – کلیک کنید (+)	۹ ساعت
آموزش طراحی لوگو با Adobe Illustrator – کلیک کنید (+)	۲ ساعت
آموزش طراحی محیط بازی های دو بعدی با نرم افزار Adobe Illustrator و Photoshop – کلیک کنید (+)	۳ ساعت

طراحی و گرافیک (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش طراحی لوگو - آشنایی با مفاهیم و ایده پردازی (همراه با مثال عملی) – کلیک کنید (+)	۱ ساعت
آموزش طراحی برچسب و کاور CD و DVD – کلیک کنید (+)	۲ ساعت
آموزش ابزار سه بعدی در فتوشاپ (Photoshop) - کلیک کنید (+)	۵ ساعت
آموزش فتوشاپ سه بعدی - طراحی و متحرک سازی - کلیک کنید (+)	۲ ساعت
آموزش موتور رندر آرنولد (Arnold Renderer) برای ایجاد جلوه های ویژه در DS Max۳ - کلیک کنید (+)	۲ ساعت
آموزش کاربرد فتوشاپ در معماری - کلیک کنید (+)	۵ ساعت

روباتیک	
عنوان آموزش	مدت زمان تقریبی
آموزش رباتیک و ربات های سری – کلیک کنید (+)	۲ ساعت
آموزش برنامه نویسی آردوینو (Arduino) با محوریت پروژه های رباتیک – مقدماتی – کلیک کنید (+)	۹ ساعت
آموزش برنامه نویسی آردوینو (Arduino) – تکمیلی – کلیک کنید (+)	۷ ساعت
آموزش سینماتیک مستقیم و معکوس روبات ها – کلیک کنید (+)	۳ ساعت
آموزش مبانی ربات های برنامه پذیر – کلیک کنید (+)	۵ ساعت
آموزش شبیه سازی و کنترل ربات با اندروید (Android) – کلیک کنید (+)	۱۶ ساعت
آموزش مکاترونیک کاربردی ۱ – کلیک کنید (+)	۱۳ ساعت
آموزش مکاترونیک کاربردی ۲ – کلیک کنید (+)	۷ ساعت
آموزش کاربردی مکاترونیک - محاسبات پایه طراحی درب های خودکار – کلیک کنید (+)	۵ ساعت
آموزش برد آردوینو (Arduino) با انجام پروژه های عملی – کلیک کنید (+)	۱۹ ساعت
آموزش پروژه محور آردوینو با سی شارپ و اندروید – کنترل موتور پله ای – کلیک کنید (+)	۵ ساعت
آموزش پیاده سازی سیستم های کنترلی با آردوینو – کلیک کنید (+)	۵ ساعت

فصل ۵:
زبان رابطه ای SQL

در این فصل به بررسی زبان رابطه ای SQL می پردازیم.

زبان رابطه ای SQL

اصولاً هر RDBMS دارای نسخه SQL خاص خود است. ویژگی های SQL عبارت است از:

۱- زبانی غیر رویه ای

۲- دارای عملگرهای بسیار قوی

۳- تامین کننده استقلال داده ای

۴- دارای اکمال ساختاری

۵- شامل تمام داده های استاندارد

۶- قابل استفاده هم به صورت مستقل و هم به صورت ادغام شده

انواع داده ها در SQL

انواع داده ها که در SQL می توان از آنها استفاده کرد عبارتند از:

نوع	نام	تعداد بایت	محدوده قابل پذیرش
	tinyint(n)	N	از 0 تا 255
	smallint	2	از 32768- تا 32767+
	int	4	از 2- میلیارد تا 2+ میلیارد (حدودا)
	intbig	8	از -2^{63} تا $2^{63}-1$
	float	8	از $-1.79E^{28}$ تا $1.79E^{28}$
	real	4	از $-3.40E^{28}$ تا $3.40E^{28}$
	decimal	حداکثر 17	از $-10^{28}+1$ تا $10^{28}-1$
	numeric	حداکثر 17	از $-10^{28}+1$ تا $10^{28}-1$
کاراکتری	**char(n)**	n	حداکثر 8000 کاراکتر

و انواع دیگر مانند:

bit , binary , image , money , currency , datetime , sysname , timestamp , varchar , varbinary , XML

انواع عملگرها در SQL

توسط عملگرها، عمل خاصی را انجام می دهیم. انواع عملگرها عبارتند از:

۱- محاسباتی (+,-,*,/,%)

۲- رابطه ای (=!, =,>,<,<=,>=)

۳- منطقی(AND,OR,NOT)

۴- بیتی (~,&,|,^)

۵- انتساب(=)

۶- ویژه (IN,BETWEEN,ALL,ANY,LIKE)

دستورهای SQL

دستورات در SQL را می توان به سه دسته زیر تقسیم بندی کرد:

۱- **DML** : (SELECT , INSERT , UPDATE , DELETE)

۲- **DDL** : (CREATE , ALTER , DROP)

۳- **DCL** : (GRANT , REVOKE)

ایجاد جدول (CREATE TABLE)

برای ایجاد بانک اطلاعاتی می توان از دستور CREATE TABLE استفاده کرد.

مثال

ایجاد جدول دانشجو (S) با فیلد های شماره دانشجویی(کلید) و نام دانشجو:

```
CREATE  TABLE  S
 (
   ID      Char (10)   PRIMARY  KEY ,
   Name   Char (20),
);
```

به جای PRIMARY KEY می توان از UNIQUE نیز استفاده کرد.

با ایجاد جدول، مشخصات جدول در کاتالوگ سیستم وارد می شود.

حذف جدول (DROP TABLE)

برای حذف یک جدول از دستور DROP TABLE ، استفاده می شود. وقتی جدولی حذف شود، فایل متناظر جدول حذف شده و تعریف جدول از کاتالوگ خارج می شود. برای حذف جدول S ، از دستور زیر استفاده می کنیم:

```
DROP   TABLE    S;
```

در دستور DROP TABLE…..option به جای option می توان CASCADE یا RESTRICT را نوشت.

در دستور زیر اگر گزینه RESTRICT را بنویسیم،

DROP TABLE base-table {RESTRICT| CASCADE}

در این صورت اگر روی جدول،دید تعریف شده باشد، جدول حذف نمی شود. همچنین اگر روی ستونی از جدول شاخص تعریف شده باشد، جدول حذف نمی شود.

درج (INSERT)

از این دستور برای اضافه کردن رکورد به جداول بانک اطلاعاتی استفاده می شود.

مثال

درج اطلاعات یک دانشجو به جدول STUDENT:

```
INSERT   INTO    S (ID,Name)
VALUES ('110' , 'Ali' );
```

اگر فیلدی NOTNULL تعریف شده باشد، مقادیر تهی برای آن نمی توان وارد کرد.

بهنگام سازی (UPDATE)

برای ویرایش رکوردهای جدول از دستور UPDATE استفاده می کنیم.

مثال

شماره دانشجویی 100 را به 200 تغییر دهید.

```
UPDATE   S
SET        ID=200
WHERE    ID=100;
```

حذف رکورد (DELETE)

دستور delete برای حذف سطرهای جدول به کار می رود.

مثال

حذف دانشجو به شماره 300 :

```
DELETE   FROM   STUDENT
WHERE   ID=300;
```

بازیابی (SELECT)

عملکرد این دستور ترکیب عملکرد دو عملگر گزینش و پرتو در جبر رابطه ای می باشد.

مثال

بازیابی شماره و نام دانشجویان از جدول S :

```
SELECT  ID, Name
FROM    S;
```

مثال

بازیابی شماره دانشجویانی که نام آنها ALI باشد از جدول S :

```
SELECT  ID
FROM    S
WHERE   Name='ALI';
```

مثال

بازیابی شماره و نام دانشجویانی که نام آنها ALI باشد:

```
SELECT  *
FROM    S
WHERE   Name='ALI';
```

مثال

بازیابی نام دانشجویان بدون نمایش نام های تکراری:

```
SELECT DISTINCT SNAME
FROM S;
```

با دستور SELECT می توان عملکرد فقط گزینش یا فقط پرتو در جبر رابطه ای را پیاده کرد. اگر از * SELECT استفاده شود، مشخصات همه ستونها داده می شود و مانند عملکرد گزینش عمل می کند.

مثال

با توجه به دو رابطه R(a,b) و S(c,d) ، معادل جبر رابطه ای هر یک از دستورات SQL داده شده در مقابل آن آورده شده است:

SELECT a **FROM** R	$\prod_{a}(R)$
SELECT * **FROM** R WHERE a=2	$\sigma_{a=2}(R)$
SELECT a **FROM** R WHERE b=5	$\prod_{a}(\sigma_{b=5}(R))$
SELECT * **FROM** R,S	$R \times S$
SELECT b,d **FROM** R,S WHERE a=c	$\prod_{b,d}(\sigma_{a=c}(R \times S))$

توابع جمعی (Aggregate Functions)

توابع جمعی در قسمت جبر رابطه ای توضیح داده شد. این توابع عبارتند از:

COUNT , MAX , MIN , SUM , AVG

مثال

پرس و جویی که تعداد دانشجویان را بر می گرداند:

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

برای شمارش تعداد سطرهای یک جدول از COUNT (*) استفاده می شود.

استفاده از DISTINCT در توابع MIN , MAX بی اثر است.

مرتب سازی رکوردها

توسط امکان ORDER BY می توان جدول جواب را برحسب یک یا بیش از یک ستون به صورت صعودی (ASC) یا نزولی (DESC) مرتب کرد. (ASC پیش فرض است).

```
SELECT *
FROM  S
ORDER BY  ID DESC;
```

البته می توان به جای ID از شماره مکانی آن در جدول یعنی عدد یک استفاده کرد.

تست مقدار تهی فیلد

توسط این امکان می توان وجود مقدار هیچ در یک ستون را بررسی کرد.

مثال

شماره دانشجویانی را بدهید که نمره آنها در درس C1 هنوز اعلام نشده است.

```
SELECT  SID
FROM    SC
WHERE  CID = 'C1'  AND GRADE  IS  NULL;
```

عملگر LIKE

توسط این امکان در SQL می توان عمل بازیابی را بر اساس نشانوند جستجوی کاراکتر با شرایط مورد نظر انجام داد. به عبارتی مشخص می کند که آیا رشته ای در قسمتی از فیلد قرار دارد یا خیر.

تذکر: به جای کاراکتر – می تواند یک کاراکترو به جای کاراکتر % می تواند تعدادی کاراکتر قرار گیرد.

مثال

مشخصات دانشجویانی را بدهید که نام آنها ۴ حرفی است و حرف دوم در نام آنها A باشد، مانند SARA.

```
SELECT *
FROM   ST
WHERE  SNAME  LIKE  '-A--';
```

■

مثال

مشخصات دانشجویانی را بدهید که نام آنها با کاراکتر A شروع شود.

```
SELECT *
FROM   ST
WHERE  SNAME  LIKE   'A%';
```

تذکر: در این مثال، اگر از '%A' استفاده شده بود، نام هایی که به A ختم می شدند مشخص می شد و اگر از '%A%' استفاده شده بود، نام هایی که در آنها A داشت مشخص می شدند.

■

مثال

مشخصات دانشجویانی را بدهید که نام آنها ۳ حرفی است و حرف اول، یکی از کاراکترهای A,B,C,D باشد.

```
SELECT *
FROM   ST
WHERE  SNAME  LIKE  '[A-D]--';
```

تذکر: به جای کاراکتر $[a_1 - a_2]$می تواند یک کاراکتر که بین a_1 تا a_2 باشد، قرار گیرد.

■

مثال

مشخصات دانشجویانی را بدهید که نام آنها ۳ حرفی است و حرف اول یکی از کاراکترهای A,B,C,D نباشد:

```
SELECT *
FROM   ST
WHERE  SNAME  LIKE  '[^A-D]--';
```

تذکر: به جای کاراکتر $[\hat{}a_1 - a_2]$می تواند یک کاراکتر که بین a_1 تا a_2 نباشد، قرار گیرد.

■

عملگر UNION

توسط این امکان می توان عملکرد عملگر اجتماع در جبر رابطه ای را پیاده سازی کرد.

مثال

با فرض اینکه اطلاعات دانشجویان در بانک S و اطلاعات اساتید در بانک T قرار دارد، برای مشخص کردن نام دانشجویان و اساتید از دستور زیر استفاده می کنیم:

```
(SELECT NAME   FROM S )
UNION
(SELECT TNAME  FROM T );
```

■

تذکر: عملگرهای UNION و INTERSECT و EXCEPT ، سطرهای تکراری را حذف می کنند مگر اینکه از UNION ALL ، INTERSECT ALL و EXCEPT ALL استفاده شود.

عملگر BETWEEN

توسط این امکان می توان وجود یک مقدار را در یک محدوده بررسی کرد.

مثال

شماره دانشجویانی را بدهید که نمره آنها در درسC1 بین ۱۰ تا ۱۲ باشد.

```
SELECT SC.SID
FROM   SC
WHERE CID='C1' AND GRADE BETWEEN 10 AND 12;
```

مثال

شرط X BETWEEN Y AND Z معادل است با:

$$X \le Y \quad AND \quad Y \le Z$$

گروه بندی اطلاعات

توسط امکان گروه بندی، می توان سطرهای یک جدول را برحسب مقادیر یک ستون گروه بندی کرد به نحوی که در هر گروه مقدار آن ستون یکسان باشد.

مثال

گروه بندی جدول R بر حسب ستون Y :

```
SELECT *
FROM   R
GROUP BY Y;
```

X	Y	Z
X1	Y1	Z1
X1	Y2	Z2
X2	Y1	Z2
X3	Y1	Z3
X3	Y3	Z4

⇒

X	Y	Z
X1	Y1	Z1
X2	Y1	Z2
X3	Y1	Z3
X1	Y2	Z2
X3	Y3	Z4

در دستور SELECT نوشتن کلازهای زیر اختیاری است:

ORDER BY, WHERE , GROUP BY

در مقابل SELECT نمی توان نام ستونی را قید کرد که جزء ستونهای گروه بندی نباشد، مگر اینکه در تابع جمعی استفاده شده باشد.

مثال

گروه بندی جدول دانشجو – درس برحسب مقادیر ستون شماره CID :

```
SELECT  SID , AVG(GRADE)
FROM    SC
GROUP BY CID;
```

برای اعمال شرط در GROUP BY از عبارت HAVING باید استفاده کرد.

استفاده از HAVING در دستور SELECT

اگر بخواهیم از توابع جمعی در شرط WHERE استفاده کنیم، پیام خطا صادر می شود و باید از HAVING به جای WHERE استفاده کرد.

بنابراین دستور زیر نادرست است:

```
SELECT   S# , AVG(QTY)
FROM     SP
WHERE    AVG(QTY) > 100
GROUP    BY  S#;
```

دستور صحیح به صورت زیر می باشد:

```
SELECT  S#,AVG(QTY)
FROM    SP
GROUP  BY  S#
HAVING  AVG(QTY) > 100;
```

در دستور SELECT ، کلاز GROUP BY می تواند بدون HAVING بکار رود.

HAVING همیشه با GROUP BY می آید.

پرس و جوهای مرکب از چندین پرس و جو تشکیل می شوند و برای ایجاد آنها از عملگرهای UNION، INTERSECT و EXCEPT که به ترتیب معادل $\cup$ و $\cap$ و $-$ در جبر رابطه ای می باشند، استفاده می شود.

پیوند رابطه ها

برای پاسخ به بعضی از پرسش ها نیاز است که به بیش از یک جدول مراجعه شود. در این حالت از پیوند باید استفاده شود.

مثال

نام دانشجویانی را بدهید که درس شماره C3 را انتخاب کرده اند.

حل:

می دانیم که شماره دانشجویان در جدول ST است و اطلاعات در رابطه با دروس ثبت نام شده در جدول SC وجود دارد.

بنابراین دو جدول ST,SC را روی صفت مشترک آنها یعنی SID پیوند می زنیم.

```
SELECT  ST.SNAME
FROM    ST ,  SC
WHERE   ST.SID = SC.SID    AND    SC.CID = ' C3  ';
```

می توان یک جدول را با خودش پیوند کرد.

مثال

شماره جفت دانشجویان از یک گروه آموزشی را بدهید.

حل:

```
SELECT  A.SID , B.SID
FROM    ST  A , ST   B
WHERE  A. SDEID = B.SDEID  AND   A.SID < B.SID;
```

در مثال بالاA وB نامهای دیگر برای جدول ST می باشند. (وقتی یک جدول را با خودش پیوند می زنیم باید این عمل صورت گیرد.)

مثال

با فرض وجود سه جدول T1,T2,T3 از یک پایگاه داده ها:

۱- پیوند ضربدری(Cross join) دو جدول T1 و T2:

```
SELECT  *
FROM    T1,T2;
```

۲- پیوند ضربدری دو جدول T1 و T2 با شرط θ:

```
SELECT   *
FROM     T1,T2
WHERE    θ ;
```

۳- پیوند دو جدول T1 و T2 روی فیلد مشترک F :

```
SELECT   *
FROM     T1,T2
WHERE    T1.F = T2.F;
```

که می توان به صوت زیر نیز نوشت:

```
SELECT   A.*
FROM     T1  A,T2  B
WHERE    A.F = B.F;
```

۴- پیوند سه جدول T1 و T2 و T3 .

ارتباط بین دو جدول T1 و T2 از طریق فیلد مشترک F1 و ارتباط بین دو جدول T2 و T3 از طریق فیلد مشترک F2 برقرار می شود. (X,Y,Z,W فیلدهای موجود در سه جدول هستند.) (θ شرط معمولی است.)

```
SELECT   X,Y,Z,W
FROM     T1  A,T2  B, T3  C
WHERE    A.F1=B.F1  AND  B.F2 = C.F2  AND  θ ;
```

پیوند درونی و بیرونی

۱- پیوند درونی

در مواقعی که تعداد شرط ها زیاد می شود، برای ساده شدن تشخیص شرط پیوند با شرط معمولی ، از پیوند درونی (INNER JOIN) استفاده می شود. در این نوع پیوند، شرط معمولی در قسمت WHERE و شرط الحاق در قسمت INNER JOIN می ایند.

```
SELECT  X,Y,Z,W   FROM  T1  A
INNER   JOIN  T2  B  ON   A.F1= B.F1
INNER   JOIN  T3  C  ON   B.F2 = C.F2
WHERE  شرط معمولی ;
```

۲- پیوند بیرونی

در این نوع پیوند، کلیه رکوردهای موجود در یک جدول، حتی رکوردهایی که در جدول دیگر وجود ندارند، ارزیابی شده و در خروجی نمایش داده می شود. پیوند بیرونی دارای سه نوع "چپ، راست و کامل" می باشد.

پیوند بیرونی چپ دو جدول T1 و T2 : کلیه رکوردهای T1 و T2 که در شرط ON صدق کنند را نمایش می دهد. همچنین رکوردهایی در T1 (جدول سمت چپ پیوند) که در شرط ON صدق نکنند، را نیز نمایش می دهد و به جای فیلدهای T2 ، مقدار null نمایش داده می شود.

```
SELECT  *
FROM   T1
LEFT  OUTER  JOIN   T2
ON     شرط پیوند ;
```

در پیوند بیرونی راست (RIGHT OUTER JOIN) دو جدول T1 و T2 ، کلیه رکوردهای T1 و T2 که در شرط ON صدق کنند را نمایش می دهد. همچنین رکوردهایی در T2 (جدول سمت راست پیوند) که در شرط ON صدق نکنند، را نیز نمایش می دهد و به جای فیلدهای T1 ، null نمایش داده می شود.

در پیوند بیرونی کامل (FULL OUTER JOIN) دو جدول T1 و T2، کلیه رکوردهای T1 و T2 با هم الحاق شده و به جای فیلدهایی از هر دو جدول که در شرط صدق نمی کنند، null نمایش داده می شود.

مثال

دو جدول M و N مفروض هستند. حاصل اجرای دستور داده شده را بدست آورید؟

```
SELECT  *
FROM  N
LEFT  OUTER  JOIN  M
ON  N.B=M.B;
```

N

A	B	C
a1	b1	c1
a2	b4	c2
a3	b2	c3

M

B	D
b1	d1
b2	d2
b2	d5

حل: حاصل پیوند بیرونی راست دو جدول داده شده بر روی فیلد B، به صورت زیر است:

A	B	C	D
a1	b1	c1	d1
a3	b2	c3	d2
a3	b2	c3	d5
a2	b4	c2	null

پرسش های تودرتو (Nested Query)

پرسش هایی که تعدادی پرسش فرعی (درونی) در درون خود دارند را پرسش تودرتو می نامند. در واقع یک دستور SELECT که درون یک دستور SELECT دیگر نوشته شود. در هنگام اجرا ابتدا پرسش داخلی اجرا می شود.

مثال

نام دانشجویانی را بدهید که درس C2 را گرفته اند.

حل:

```
SELECT  SNAME
FROM    ST
WHERE  SID  IN  ( SELECT  SID
                  FROM    SC
                  WHERE   CID='C2' );
```

در این مثال SELECT داخلی جدولی شامل شماره دانشجویانی را می دهد که درس C2 را گرفته انـد. سـپس SELECT دوم نام دانشجویانی را می دهد که شماره آنها در جدول حاصل از SELECT داخلی وجود داشته باشد.

عملکرد =ANY ، معادل عملکرد عملگر IN است.

اگر مجموعه جواب پرسش داخلی، تک عنصری باشد آنگاه می توان قبل از SELECT داخلی، مستقیماً از عملگرهای $=, <, >, \leq, \geq$ استفاده کرد.

مثال

نام دانشجویان هم رشته با دانشجوی شماره 110 را بدهید.

حل:

```
SELECT   SNAME
FROM     ST
WHERE    SMJR = ( SELECT    SMJR
                  FROM      ST
                  WHERE     SID = ' 110 ' );
```

مثال

نام افرادی را بدهید که وزن آنها از بیشترین وزن موجود در پایگاه کمتر باشد.(جدول T با دو فیلد NAME,WEIGHT مشخصات افراد شامل نام و وزن آنها را نگهداری می کند.)

```
SELECT   T
WHERE    WEIGHT < (  SELECT   MAX(WEIGHT)
                     FROM   T  );
```

می توان از سور وجودی EXISTS درSQL استفاده کرد.

مثال

نام دانشجویانی را بدهید که درسC5 را انتخاب نکرده اند.

حل:

```
SELECT   SNAME
FROM     ST
WHERE   NOT EXISTS( SELECT  *
                    FROM     SC
                      WHERE   SC.SID=S.SID  AND   SC.CID='C5'
                    );
```

پایگاه داده "تهیه کننده – قطعه"

پایگاه داده ای تهیه کننده و قطعه دارای سه جدول است:

S (S# , SNAME , STAUAS , CITY)
P (P# , PNAME , COLOR , WEIGHT , CITY)
SP (S# , P# , QTY)

در جدول S مشخصات تهیه کنندگان و در جدول P مشخصات قطعات و در جدول SP میـزان تولیـد قطعـات توسـط تهیـه کننده گان نگهداری می شود. جدول ها را با مقادیر فرضی پر می کنیم:

S

S#	SNAME	STATUS	CITY
S1	Sn1	20	C1
S2	Sn2	10	C2
S3	Sn3	30	C2
S4	Sn4	20	C1
S5	Sn5	30	C3

P

P#	PNAME	COLOR	WEIGHT	CITY
P1	Pn1	RED	12	C1
P2	Pn2	YELLOW	17	C2
P3	Pn3	BLUE	17	C4
P4	Pn3	GREEN	14	C1
P5	Pn5	BLUE	12	C2
P6	Pn6	BLACK	19	C1

SP

S#	P#	QTY
S1	P1	100
S1	P4	200
S2	P1	300
S2	P2	400
S3	P6	500
S4	P3	500
S5	P2	800
S5	P4	700
S5	P6	200

حال با توجه به این رابطه ها به هر یک از سئوالات زیر پاسخ دهید:

۱- نام تهیه کنندگانی را بیابید که قطعه P2 را تهیه می کنند.

```
SELECT   SNAME
FROM    S
WHERE    S#  IN ( SELECT   S#
                 FROM    SP
                 WHERE    P# = 'P2'
                    );
```

با توجه به رابطه ها، نتیجه SELECT داخلی برابر است با:

S#
S2
S5

و نتیجه SELECT خارجی برابر است با:

SNAME
Sn2
Sn5

به جای IN از ANY = نیز می توان استفاده کرد.

۲- پرس و جوی قبلی را به کمک EXEISTS بنویسید.

```
SELECT   SNAME
FROM  S
WHERE   EXISTS( SELECT  *
                FROM   SP
                  WHERE    SP.S# = S.S #   AND  SP.P# = 'P2');
```

۳- نام تهیه کنندگانی را بیابید که اقلاً یک قطعه به رنگ RED تهیه می کنند.

```
SELECT   SNAME
FROM     S
WHERE     S#  IN  ( SELECT   S#
                    FROM      SP
                  WHERE    P#   IN ( SELECT   P#
                                    FROM      P
                                    WHERE   COLOR = 'RED'
                                   )
                );
```

برای پاسخ به این سئوال به سهSELECT تودرتو نیاز است. خروجیSELECT داخلی با توجه بـه داده هـای فرضـیP1 اسـت. خروجSELECT میانیS1 و S2 است و خروجیSELECT خارجیSn1 , Sn2 است.

۴- شماره قطعاتی را بیابید که توسط بیش از یک تهیه کننده، تهیه شده باشد.

```
SELECT  P#
FROM   SP
GROUP   BY  P#
HAVING   COUNT(*) > 1;
```

رابطه SP بعد از گروه بندی روی P# :

S#	P#	QTY
S1	P1	100
S2	P1	300
S2	P2	400
S5	P2	800
S4	P3	500
S1	P4	200
S5	P4	700
S3	P6	500

و حاصل این پرس و جو برابر است با : P4 , P2 , P1

۵- حداکثر مقدار تهیه شده از هر قطعه را بیابید.

```
SELECT   P# , MAX(QTY)
FROM   SP
GROUP BY   P#;
```

جدول جواب به صورت زیر است:

P#	
P1	300
P2	800
P3	500
P4	700
P6	500

۶- شماره قطعاتی را بیابید که یا وزن آنها بیشتر از 16 باشد یا توسط S2 تهیه شده است یا هر دو شرط را دارد.

```
SELECT  P#  FROM  P  WHERE  WEIGHT > 16
UNION
SELECT  P#  FROM  SP  WHERE  S# = 'S2' ;
```

نتیجه برابر است با :

P#
P1
P2
P3

یادآوری: عملگرUNION موجب حذف عناصر تکراری می شود.

پایگاه داده "تهیه کننده – قطعه – پروژه"

پایگاه داده ایSPJ با جدول های تهیه کننده، قطعه و پروژه به صورت زیر مفروض است:

S (S# , SNAME , STATUS , CITY)
P (P# , PNAME , COLOR , WEIGHT , CITY)
J (J# , JNAME , CITY)
SPJ (S# , P# , J# , QTY)

با توجه به این پایگاه به سئوالات زیر پاسخ دهید:

۱- تمام جفت هایی از نام شهرها را مشخص کنید که عرضه کننـــده ای در شـــهر اول، پـــروژه ای در شـــهر دوم را پشتیبانی کند.

```
SELECT   DISTINCT   S.CITY  AS  A , J.CITY  AS  B
FROM     S , J
WHERE   EXISTS ( SELECT  *
                 FROM     SPJ
                 WHERE   SPJ.S# = S.S#    AND   SPJ.J#  =J.J# );
```

۲- شماره قطعاتی را بیابید که به پروژه ای اختصاص می یابد و میانگین آن بیش از ۳۰۰ است.

```
SELECT   DISTINCT    SPJ.P#
FROM   SPJ
GROUP BY   SPJ.P# ,  SPJ.J#
HAVING      AVG ( SPJ.QTY > 300 );
```

۳- شماره پروژه هایی را مشخص کنید که شهر آن پروژه، اولین شهر به ترتیب الفبا باشد.

```
SELECT  J.J#
FROM     J
WHERE   J.CITY = ( SELECT   MIN(J.CITY)
                   FROM   J  );
```

۴- شماره پروژه هایی را مشخص کنیدکه به طور کامل توسط S1 پشتیبانی می شوند.

```
SELECT     J.J#
FROM       J
WHERE     NOT   EXISTS ( SELECT    *
                          FROM     SPJ
                          WHERE     SPJ.J#  = J.J#   AND  NOT ( SPJ.S# = 'S1' )   );
```

۵- جفت هایی از شماره قطعه/ شماره عرضه کننده را مشخص کنید که عرضه کننده خاص قطعه مشخص شده را عرضه نمی کند.

```
SELECT   S.S# , P.P#   FROM  S  CROSS JOIN   P
EXCEPT
SELECT   SPJ.S# , SPJ.P#    FROM     SPJ;
```

مثلا اگر جواب به صورت (S1,P5) باشد، یعنی تهیه کننده S1 قطعه P5 را تولید نمی کند.

پایگاه داده "بانک"

پایگاه داده bank با شش جدول زیر مفروض است:

1- **customer** (**name** : نام مشتری , **city** : شهر مشتری , **addr** : آدرس مشتری)

2- **account** (**acc** : شماره حساب , **bname**: نام شعبه , balance : موجودی)

3- **depositor** (**name**: نام مشتری , **acc** : شماره حساب)

4- **borrower** (**name**: نام مشتری , **Ln** : شماره وام)

5- **loan** (**ln** : شماره وام , **bname** : نام شعبه , **amout** : مبلغ وام)

6- **branch** (**bname** : نام شعبه , **cityb** : شهر شعبه , **d** : دارایی)

با توجه به این پایگاه به سئوالات زیر پاسخ دهید:

۱- نام مشتریانی که حساب، وام یا هر دو را دارند:

```
SELECT name FROM customer
UNION
SELECT name FROM borrower
```

۲- نام مشتریانی که هم حساب و هم وام دارند:

```
SELECT name FROM customer
INTERSECT
SELECT name FROM borrower
```

۳- بازیابی نام تمام شعبه ها در جدول loan :

```
SELECT bname FROM loan
```

۴- شماره وام های شعبه تجارت مرکزی با مبلغ وام بیشتر از دو میلیون

```
SELECT ln
FROM loan
WHERE bname ='تجارت مرکزی' AND amount > 2000000
```

۵- نام شعبه ها و میانگین موجودی آن ها به شرط آنکه میانگین موجودی انها از دو میلیارد تومان بیشتر باشد.

```
SELECT bname, AVG(balance)
FROM account
GROUP BY bname
Having AVG(balance)>2000000000
```

۶- نام مشتریانی که حداکثر یک حساب در شعبه تجارت مرکزی دارند.

```
SELECT distinct name
FROM despositor AS x
Where unique( SELECT x.name
              FROM account y , despositor as z
              WHERE x.name=z.name AND y.acc=z.acc AND y.bname='تجارت مرکزی' )
```

۷- نام مشتری، شماره وام و میزان وام، وام گیرندگان از بانک

```
SELECT  name , ln , amount
FROM   borrower  b
Inner join   loan
ON   b.ln=b.ln
```

۸- نام مشتری، شماره وام و میزان وام کسانی که از بانک تجارت مرکزی وام گرفته اند

```
SELECT  name , b.ln , amount
FROM    borrower   b
INNER JOIN  loan  k
ON  b.ln=k.ln
WHERE  bname='تجارت مرکزی'
```

مجوز در SQL

سرپرست پایگاه داده ها می تواند به کاربران SQL ، مجوزهایی مانند "درج رکورد، حذف رکورد، تغییر رکورد ، ایجاد جدول، حذف جدول ، اضافه کردن فیلد، حذف فیلد و ..." را اعطا کند یا از آنها پس بگیرد. برای اعطا از دستور GRANT و برای پس گیری از دستور REVOKE استفاده می شود.

مثال

اعطای مجوز درج در جدول ST به کاربر ALI :

```
GRANT   INSERT
ON   ST
TO   ALI;
```

تذکر:

با استفاده از عبارت WITH GRANT OPTION در انتهای دستور بالا، این امکان به کاربر ALI داده می شود که بتواند مجوز خودش را به کاربران دیگر نیز بدهد.

مثال

پس گرفتن مجوز درج در جدول ST از کاربر ALI :

```
REVOKE   INSERT
ON   ST
FROM   ALI;
```

تذکر:

اگر از عبارت Cascade در انتهای دستور بالا استفاده شود، لغو مجوز به صورت آبشاری خواهد بود. یعنی اگر به طور نمونه در گراف مجوز ali پدر کاربری مانند omid باشد، آنگاه با لغو مجوز ali، مجوز omid نیز لغو می شود.

مثال

دستور زیر، اجازه انتخاب ستون ها و نیز به روز در آوردن فیلدهای SNAME,SDEG جدول ST را به کاربر sara می دهد.

```
GRANT   SELECT,UPDATE(SNAME,SDEG)
ON   ST
TO   sara;
```

لیست مجوزها در جدول زیر آورده شده است:

مجوز	هدف
SELECT	بازیابی داده
INSERT	اضافه کردن رکورد
UPDATE	تغییر رکورد
DELETE	حذف رکورد
DROP	حذف جدول
ALTERATION	حذف یا اضافه فیلد
INDEX	ایجاد یا حذف شاخص
EXECUTE	اجرای تابع ذخیره شده
USAGE	استفاده از دامنه خاص
REFERENCES	تعیین کلید خارجی در هنگام تعریف جدول
ALLPRIVILEGE	کلیه مجوزها

مجوز حذف ستون را نمی توان به کاربر داد.

با GRANT امتیاز انجام یک عمل روی جدول مبنا و جدول مجازی(دید) را می توان به کار برد. ولی روی جدول موقتی نمی توان به کار برد.

دید

دید نوعی رابطه نامدار، مشتق و مجازی است که توسط دستورCREATE VIEW ایجاد و توسط دستور DROP VIEW حذف می شود.

انواع رابطه ها عبارتند از:

نامدار	رابطه ای که با یک نام به سیستم معرفی می شود.
مبنا	نوعی رابطه نامدار که از رابطه های دیگر مشتق نشده است و دارای داده های ذخیره شده متناظر است.
مشتق	رابطه ای که به کمک یک عبارت رابطه ای برحسب رابطه های مبنا ،تعریف می شود.
دید VIEW	نوعی رابطه نامدار که از رابطه های دیگر مشتق شده است و ماهیتاً رابطه مجازی است و داده های ذخیره شده خاص خود را ندارد.
لحظه ای SNAPSHOT	نوعی رابطه نامدار و مشتق مانند دید است اما واقعی که تاپلهای آن می توانند تغییر کنند . این رابطه در کاربردهایی چون حسابداری و تهیه گزارشات آماری پریودیک ،کاربرد دارد. فرق آن با VIEW در این است که اولاً واقعی است و ثانیاً مقادیر آن در پایگاه ذخیره می شود.
عبارتی	نوعی رابطه که از عبارت رابطه ای بدست می آید.
نتیجه پرسش	رابطه ای بی نام و مشتق که حاصل اجرای یک پرسش مشخص است.
بینابینی	رابطه ای بی نام و مشتق که حاصل ارزیابی یک عبارت رابطه ای است که درون یک عبارت بزرگتر جای دارد.

مثال

تعریف یک دید به نامV1 روی جدول S :

```
CREATE  VIEW  V1 (A,B,C )
AS  SELECT  S # , STATUS , CITY
FROM  S
WHERE  STATUSE > 15 ;
```

S

S	Sname	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

⇒

V1

A	B	C
S1	20	London
S3	30	Paris
S4	20	London
S5	30	Athens

مثال

تعریف یک دید روی دید V1 :

```
CREATE VIEW   V2 (M,N,P)
AS SELECT  A,B,C
FROM     V1
WHERE  P < > 'London' ;
```

V2

M	N	P
S3	30	Paris
S5	30	Athens

دستور DROP VIEW V1 CASCADE موجب حذف دید V1 و V2 خواهد شد. یعنی با حذف جدول پایه، تمام دیدهای تعریف شده روی آن نیز حذف می شوند.

مثال

تعریف دید PQ روی جدول SP :

```
CREATE VIEW   PQ ( P# , TQ)
AS SELECT       P# , SUM (QTY)
FROM    SP
GROUP BY  P#;
```

SP

S #	P#	QTY
S1	P1	10
S1	P2	30
S2	P2	15
S3	P1	30
S3	P3	5
S4	P2	5

⇒

PQ

P #	TQ
P1	40
P2	50
P3	5

بازیابی از دید

مثال

دید V1 را به صورت زیر تعریف می کنیم:

```
CREATE  VIEW  V1
AS  SELECT  S # , STATUS , CITY
FROM     S
WHERE  STATUS > 15 ;
```

حال اگر کاربری، حکم بازیابی زیر را صادر کند:

```
SELECT  *
FROM     V1
WHERE   CITY = 'PARIS';
```

این حکم بعد از تبدیل به صورت زیر در می آید:

```
SELECT  S # , STATUS , CITY
FROM     S
WHERE  CITY = 'PARIS'  AND  STATUS > 15;
```

گاهی اوقات مشکلاتی در بازیابی از دید برای کاربر رخ می دهد.

مثال

دید PQ به صورت زیر مفروض است:

```
CRETATE  VIEW   PQ ( P# , TQ )
AS  SELECT  P# , SUM (QTY)
FROM  SP
GROUP  BY  P#;
```

اگر دستور بازیابی زیر توسط کاربر صادر شود:

```
SELECT  *
FROM    PQ
WHERE  TQ > 300;
```

به صورت زیر تبدیل خواهد شد:

```
SELECT  P# , SUM (QTY)
FROM   SP
WHERE   SUM(QTY) > 300
GROUP   BY  P#;
```

این حکم غیرمجاز است چون نمی توان از تابع جمعی در برابرWHERE استفاده کرد.

حذف از دید

مثال

دیدV1 شامل مشخصات کامل دانشجویان مقطع کارشناسی ارشد است:

```
CREATE VIEW V1
AS SELECT *
FROM S
WHERE SDEG = 'ms';
```

حال حذف از این دید را بدون هیچ مشکلی می توان انجام داد :

```
DELETE
FROM V1
WHERE SID = '123';
```

این دستور تبدیل به حکم زیر خواهد شد :

```
DELETE
FROM S
WHERE SID = '123';
```

درج در دید

مثال

عمل درج در دیدV1 مثال قبل به صورت زیر انجام می شود:

```
INSERT INTO V1
VALUES ( '300' , 'Ali' , 'ms' , 'Comp' , 'D65' );
```

اگر سعی به درج دانشجویی با شماره دانشجویی تکراری شود، آنگاه در صورت استفاده از عبارت WITH CHECK OPTTION در هنگام تعریف دید، سیستم از این عمل جلوگیری می کند.

شرایط قابل بهنگام بودن دیدها

۱- عدم وجود DISTINCT

۲- عدم وجود GROUP BY

۳- عدم وجود HAVING

۴- عدم استفاده از توابع جمعی در مقابلSELECT

۵- عدم وجود بیشتر از نام یک جدول در مقابل FROM

۶- جدول مقابل FROM باید یک جدول مبنا یا یک دید قابل بهنگام سازی باشد.

یکی از معایب مدل رابطه ای این است که بعضی از دیدها، عملیات بهنگام سازی را نمی پذیرند.

مزایا و معایب دید

مزایا:

۱- تامین کننده پویایی بالا در تعریف پایگاه

۲- تامین کننده محیط انتزاعی برای کاربران سطح خارجی

۳- تامین کننده اشتراک داده ها

۴- تامین کننده استقلال داده ای

۵- امکان تعریف Data – Object با اندازه های مختلف

۶- امکانی برای کوتاه نویسی پرسش ها

۷- تسهیل کننده واسط کاربر برنامه ساز با پایگاه

۸- تامین کننده مکانیسم خودکار امنیت داده ها

معایب:

۱- ایجاد اضافه کاری در سیستم برای انجام تبدیل خارجی/ ادراکی

۲- عدم امکان انجام عملیات ذخیره سازی در بسیاری از دیدها و در نتیجه ایجاد محدودیت برای کاربر.

انواع دیدها از نظر پذیرش عملیات ذخیره سازی

۱- دیدهای پذیرا : دیدهایی که عملیات ذخیره سازی را می‌پذیرند.

۲- دیدهای ناپذیرا : دیدهایی که عملیات ذخیره سازی را نمی‌پذیرند.

در دیدهای پذیرا می‌توان عملیات ذخیره سازی را انجام داد، اما گاهی به مشکلاتی برمی‌خوریم.

انواع دیدهای پذیرا

۱- دید گزینش : حاصل عملکرد گزینش در یک رابطه

۲- دید پیوندی CK – CK: حاصل پیوند دو رابطه روی کلید کاندید مشترک آنها که در عملیات ذخیره سازی مشکلی ندارد.

۳- دید پیوندی CK - FK : حاصل پیوند روی کلیه کاندید یک رابطه و کلید خارجی رابطه دیگر که در حذف مشکل دارد

۴- دید حاصل از اجتماع، اشتراک و تفاضل دو رابطه : بدون مشکل در عملیات ذخیره سازی

۵- دید گزینش- پرتوی دارای کلید رابطه مبنا

انواع دیدهای ناپذیرا

۱- دید پرتوی یا گزینش – پرتوی فاقد کلید

۲- دید پیوندی FK - FK

۳- دید پیوندیNK - NK

۴- دید حاصل تقسیم

۵- دید حاوی صفت مجازی

دید پیوندیNK - NK ، حاصل پیوند روی صفت مشترک غیرکلید دو رابطه می باشد.

در دید حاصل تقسیم، انجام عملیات ذخیره سازی ناممکن است

متغیرها- ساختارهای تصمیم – رویه های ذخیره شده- توضیحات

برای تعریف متغیر در SQL از دستور DECLARE و برای مقدار دهی متغیر از SET استفاده می شود.

مثال

تعریف متغیر x از نوع int و مقداردهی آن با 10 :

```
DECLARE  @x  int
SET  @x=10
```

به کمک دستور CREATE TYPE می توان یک نوع داده را تعریف کرد. برای تعریف نوع داده مجرد(انتزاعی)،از دستور زیر استفاده می کنیم:

```
CREATE TYPE name WITH OID;
```

تذکر: نوع داده POINTER را نمی توان تعریف کرد.

تذکر: متغیرهای سیستمی SQL با دو کاراکتر@@ شروع می شوند. مانند متغیر @@rowcount که تعداد سطرهای جدول را نگهداری می کند.

ساختارهای تصمیم

برخی از ساختارهای تصمیم در SQL عبارتند از:

دستور CASE	دستور IF
CASE عبارت **WHEN** شرط۱ **THEN** نتیجه۱ **WHEN** شرط۲ **THEN** نتیجه۲ ... **WHEN** شرطn **THEN** نتیجه n **ELSE** نتیجه **k** **END**	**IF** شرط دستور۱ **ELSE** دستور۲

رویه های ذخیره شده

می توان مجموعه ای از دستورات SQL را در یک مجموعه قرار داد و کامپایل کرد و با یک دستور اجرا کـرد. رویه های ذخیره شده برنامه نویسی ماژولی را ممکن می سازد. همچنین چون رویه ها بـه صـورت کامپایـل شده در حافظه نهان بانک اطلاعاتی نگهداری می شوند، سرعت اجرا افزایش می یابد.

دستورات کار با رویه به صورت زیر است:

۱- ایجاد رویه : CREATE PROCEDURE

۲- اجرای رویه: EXEC

۳- تغییر رویه : ALTER PROCEDURE

۴- حذف رویه : DROP PROCEDURE

توضیحات Comments

به دو روش می توان توضیحات را در SQL تعریف کرد:

۱- توضیحات یک سطری : علامت -- و سپس توضیحات.

۲- توضیحات چند سطری : علامت */ در ابتدا و علامت /* در انتهای توضیحات.

آزمون

۱- رابطه R(a,b,c) را در نظر بگیرید: (دولتی ۸۹)

Q_1 : **SELECT Distinct** a,b **FROM** R

Q_2 : **SELECT** a,b **FROM** R **GROUP BY** a,b

۱) پاسخ Q_1 زیر مجموعه ای از پاسخ Q_2 است.

۲) Q_1 و Q_2 پاسخ های یکسان تولید می کنند.

۳) پاسخ Q_2 زیر محموعه ای از پاسخ Q_1 است.

۴) Q_1 و Q_2 پاسخ های متفاوت تولید می کنند.

۲-۱) هر دو دستور داده شده معادل هستند. نتیجه Q_1 ستونهای a و b از رابطه R(a,b,c) با حذف تکراری ها می باشد. در Q_2 ابتدا جدول بر اساس ستون a وb دسته بندی شده و سپس برای هر گروه، a و b چاپ می شود و خروجی آن مانند Q_1 است.

۲- با اجرای دستور SQL زیر روی بانک اطلاعاتی تولید کنندگان قطعات، کدام یک از گزاره های زیر صحیح است؟ (دولتی ۸۷)

```
INSERT INTO S(S# , SNAME,CITY)
VALUES ( 'S10' , 'Smith ', 'New York' )
```

۱) دستور با خطای اجرا مواجه می شود.

۲) دستور با خطای ترجمه مواجه می شود.

۳) اعتبار(STATUS) تولید کننده Smith مقدار Null خواهد شد.

۴) اعتبار(STATUS) تولید کننده Smith مقدار قبلی خود را حفظ می کند.

۲-۴) با اجرای دستور SQL زیر بر روی S(s#,sname,city,status) ، چون به status مقداری اختصاص داده نشده است، مقدار قبلی خود را حفظ خواهد کرد:

```
INSERT INTO S(S# , SNAME,CITY)
VALUES ( 'S10' , 'Smith' , 'New York');
```

۳- نتیجه دستور SQL زیر چیست؟ (دولتی ۸۷)

```
SELECT  P.*
FROM    S,P,SP
WHERE   S.CITY='LONDON'  AND  S.S#=SP.S#  AND  P.P#=SP.P#
```

۱) دستور دارای خطای نحوی است.

۲) کلیه مشخصات قطعاتی که توسط تمام تولید کنندگان لندن تولید می شود.

۳) کلیه مشخصات قطعاتی که حداقل توسط یکی از تولید کنندگان لندن تولید می شود.

۴) کلیه مشخصات قطعاتی که دقیقا توسط یکی از تولید کنندگان لندن تولید می شود.

۳-۳) برای تعیین کلیه مشخصات قطعاتی که حداقل توسط یکی از تولید کنندگان لندن تولید می شود، از دستور SQL زیر استفاده می کنیم:

```
SELECT  P.*
FROM    S,P,SP
WHERE   S.CITY = 'LONDON'  AND  S.S#=SP.S#  AND  P.P#=SP.P#;
```

۴- جمله SQL زیر در بانک اطلاعاتی تولید کنندگان و قطعات چه کار می کند؟ (IT- دولتی ۸۹)

```
SELECT  DISTINCT  SNAME
FROM    S
WHERE   S.S#  IN (  SELECT  SP.S#
                    FROM    SP
                    WHERE   SP.P#  IN ( SELECT   P.P#
                                         FROM   P
                                         WHERE   PCOLOR='Red' )
                 )
```

۱) اسامی تولید کنندگانی که حداقل یک قطعه قرمز رنگ تولید می کنند.

۲) اسامی تولید کنندگانی که حداکثر یک قطعه قرمز رنگ تولید می کنند.

۳) اسامی تولید کنندگانی که بیش از یک قطعه قرمز رنگ تولید می کنند.

۴) هیچ کدام

۴-۱) توسط داخلی ترین select ، شماره قطعات قرمز رنگ مشخص می شوند:

```
SELECT P.P#
FROM    P
WHERE   PCOLOR = 'Red'
```

توسط select میانی، مشخص می شود که این قطعات توسط کدام تهیه کنندگان تولید می شوند:

```
SELECT  SP.S#
FROM    SP
WHERE   SP.P#
```

در نهایت توسط اولین select اسامی تولید کنندگانی که حداقل یک قطعه قرمز رنگ را تولید کرده اند، مشخص می شود:

```
SELECT  DISTINCT SNAME
FROM    S
WHERE  S.S#
```

۵- جمله SQL زیر در بانک اطلاعاتی تولید کنندگان و قطعات چه کار می کند؟ (IT- دولتی ۸۹)

```
SELECT  S.SNAME
FROM  S
WHERE  EXISTS (  SELECT  *
                 FROM  SP
                 WHERE  SP.S# = S.S#  AND  SP.S#='P2' )
```

۱) اسامی تولید کنندگانی که قطعه 'P2' را تولید می کنند بدون تکرار.

۲) اسامی تولید کنندگانی که قطعه 'P2' را تولید می کنند.

۳) اسامی تولید کنندگانی که قطعه ای غیر از 'P2' هم تولید می کنند.

۴) اسامی تولید کنندگانی که قطعه 'P2' را تولید نمی کنند.

۲-۵) جمله SQL داده شده، اسامی تولید کنندگانی که قطعه 'P2' را تولید می کنند، را مشخص می کند. این سئوال به کمک پیوند جدول ها نیز انجام پذیر است:

```
SELECT   s.sname
FROM     s,sp
WHERE   s.s#=sp.s#  AND  sp.p#='p2';
```

فصل ۶:

انواع وابستگی ها

در این فصل وابستگی، مجموعه کهینه وابستگی و نحوه تعیین کلید کاندید بررسی می شود.

انواع وابستگی ها

انواع وابستگی ها عبارتند از:

۱- وابستگی تابعی(FD)

۲- وابستگی تابعی کامل(FFD)

۳- وابستگی با واسطه

۴- وابستگی تابعی چند مقداری(MVD)

۵- وابستگی پیوندی(JD)

FD : Function Dependency FFD: Full Function Dependency
MVD : Multi Valued Dependency JD : Join Dependency

وابستگی تابعی

رابطه R (A , B , …) را در نظر بگیرید. می گوییم B با A وابستگی تابعی(FD) دارد و نشان می دهیم A→B ، اگر و فقط اگر در هر مقدار ممکن از متغیر رابطه R ، به هر مقدار A فقط یک مقدار B متناظر باشد.

مثال

در رابطه R(A,B,C) که در زیر آورده شده است، FD ها را مشخص کنید.

A	B	C
A1	B2	C1
A1	B1	C2
A2	B1	C3
A2	B2	C1

حل: کلید رابطه (A,B) است، بنابراین وابستگی (A,B)→C وجود دارد. همچنین مشخص است که وابستگی تابعی C→B نیز در رابطه وجود دارد، یعنی با معلوم بودن مقدار C می توان مقدار B را مشخص کرد و جواب یکتا است. مثلا مقدار متناظر با C1 همواره B2 است. دقت شود که وابستگی B→C وجود ندارد، چون مثلا مقدار متناظر با B1 یکتا نیست.

وجود FD، A→B در رابطه لزوماً موجب برقرار بودن FD، B→A نخواهد بود.

اگر A کلید اصلی رابطه R (A ,B ,C) باشد، در اینصورت هر صفت خاصه دیگر رابطه با A دارای وابستگی تابعی است:

A→B , A→C

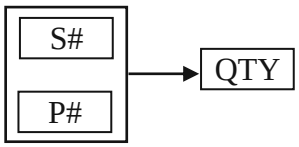
۳-۶ وابستگی تابعی کامل(FFD)

اگرX وY دو زیر مجموعه از مجموعه عنوان رابطه R باشند، می گوییمY با X وابستگی تابعی کامل دارد و نشان می دهیمX⇒Y ، اگر و فقط اگرY با X وابستگی تابعی(FD) داشته باشد ولی با هیچ زیر مجموعه از X وابستگی تابعی نداشته باشد. بدیهی است اگر سمت چپ FD صفت ساده باشد، وابستگیFFD خواهد بود.

مثال

در رابطه (S# , P# , QTY) SP ، صفتQTY با صفت مرکب(S# , P#) وابستگی تابعی کامل دارد، یعنی وابستگی تابعی (S# , P#)→QTY وجود دارد، در حالیکه هیچ کدام از وابستگی های S#→QTY و P#→QTY برقرار نمی باشد.

S#	P#	QTY
S1	P1	100
S2	P2	400
S3	P6	100
S4	P2	300
S4	P5	400



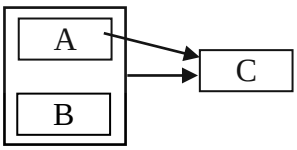
مثال

در رابطه R(A,B,C,D) اگر وابستگی های تابعی زیر را داشته باشیم :

(A,B)→C

A →C

در این صورت وابستگی تابعیC به (A,B) ، کامل نمی باشد.(جزئی است)، چون صفت C با قسمتی از کلید اصلی وابستگی تابعی دارد.



وابستگی با واسطه

رابطهR (A,B,C) مفروض است. اگر صفت B با صفت A ، FD داشته باشد $(A \rightarrow B)$و صفت C نیز با صفت B ، FD داشته باشد $(B \rightarrow C)$، ولی A با B، FD نداشته باشد، می گوییم C باA ، وابستگی با واسطه دارد. برای از بین بردن این وابستگی، رابطه را به دو رابطه زیر تجزیه می کنیم:

R1 (A,B)

R2 (B,C)

تذکر: اگر در تعریف بالاA با B وابستگی تابعی داشته باشد، وجود نوعی وابستگی بینC و A طبیعی و محرز است و موجب آنومالی نخواهد بود.

قواعد استنتاج آرمسترانگ

با فرض اینکهA , B , C , D زیر مجموعه هایی از صفات رابطه R باشند، قواعد زیر برقرارند:

انعکاسی	اگر $B \subseteq A$ آنگاه A→B
تعدی (تراگذاری)	اگر A→B و B→C آنگاه A→C
افزایش	اگر A→B آنگاه AC →BC
تجزیه	اگر A→BC آنگاه A→B و A→C
اجتماع	اگر A→B وA→C آنگاه A→BC
ترکیب	اگر C→D و A→B آنگاه AC→BD
شبه تعدی	اگر CB→D و A→B آنگاه AC→D

اگرA→B و AB→C آنگاه A→C .

اگرA→B وAB→CD آنگاه A→CD .

کاربرد قوانین آرمسترانگ عبارتند از:

۱- کاهش مجموعه وابستگی های تابعی

۲- پیدا کردن کلید کاندید

۳- پیدا کردن F^+

تذکر: هر رابطه ای یک مجموعه از وابستگی های تابعی دارد که ممکن است مجموعه ای دیگر از آنها منطقاً قابل استنتاج باشد. این مجموعه را بستارF گویند و با F^+ نمایش می دهند.

مثال

مجموعه کهینه وابستگی های رابطه زیر را بیابید.

R= { S# , CITY , STATUS }
F ={ S# →CITY, CITY → STATUS , S# → STATUS }

حل: وابستگی سوم از دو وابستگی دیگر منطقاً قابل استنتاج است و می توان آن را ذکر نکرد:

S# → CITY
⇒ S# →STATUS
CITY → STATUS

بنابراین داریم:

F = { S# →CITY, CITY → STATUS }

■

مثال

مجموعه کهینه وابستگی های رابطه زیر را بیابید.

R = {u, v, w , x , y , z }
F = {u →xy , x→y , xy→zv }

حل:

u →xy ⇒ u→x , u→y
x→y , xy→zv ⇒ x→zv ⇒ x→z , x→v
u→xy , xy→zv⇒ u→zv ⇒ u→z , u→v

بنابراین F بهینه برابر است با :

F = { u→x , x→y , x→z , x→v , u→y, u→z , u→v }

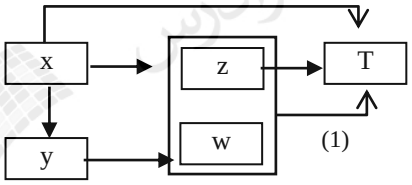
که u→y , u→z , u→v اضافی هستند، چون از رابطه های دیگر می توان آنها را بدست آورد. در نتیجه کهینه آن برابر است با:

F = { u→x , x→y , x→z , x→v }

■

مثال

نمودار حاصل پس از حذف وابستگی های نمودار زیر را بدست آورید.

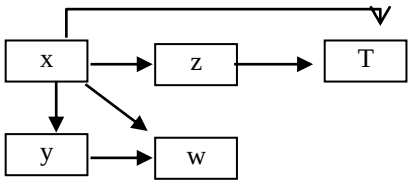


حل :

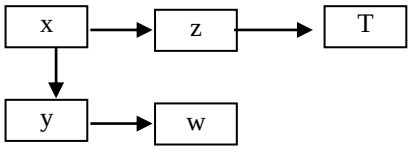
خط (۱) را می توان حذف کرد، چون: $Z \to T \Rightarrow (Z , W) \to T$

از وابستگی $X \to ZW$ می توان استنتاج کرد: $X \to Z , X \to W$

بنابراین داریم:



از وابستگی $X \rightarrow Y$ و $Y \rightarrow W$ می توان X->W را استنتاج کرد، پس می توان $X \rightarrow W$ را حذف کرد. همچنین از $X \rightarrow Z$ و $Z \rightarrow T$ می توان $X \rightarrow T$ را استنتاج کرد. پس می توان $X \rightarrow T$ را نیز حذف کرد. در نهایت داریم:



پیدا کردن کلید کاندید

کلید کاندید ، صفتی است که از طریق آن به همه صفتهای دیگر می توان رسید.

مثال

کلید کاندید رابطه زیر را بدست آورید.

$R = (A , B, C , D , E , F, G)$

$F = \{AF \rightarrow BE , FC \rightarrow DE , F \rightarrow CD , D \rightarrow E , C \rightarrow A \}$

حل :

$AF \rightarrow BE \Rightarrow AF \rightarrow B , AF \rightarrow E$

$FC \rightarrow DE \Rightarrow FC \rightarrow D , FC \rightarrow E$

$F \rightarrow CD \Rightarrow F \rightarrow C , F \rightarrow D$

$F \rightarrow C , FC \rightarrow D \Rightarrow F \rightarrow D$

$F \rightarrow C , FC \rightarrow E \Rightarrow F \rightarrow E$

$F \rightarrow C , C \rightarrow A \Rightarrow F \rightarrow A$

$F \rightarrow A , AF \rightarrow B \Rightarrow F \rightarrow B$

$F \rightarrow A , AF \rightarrow E \Rightarrow F \rightarrow E$

بنابراین مشاهده می شود که از F به همه صفتها به جزء G می توان رسید. بنابراین (F,G) کلید کاندید است.

مثال

کلید کاندید رابطه زیر را بدست آورید؟

$R = (S, T , U , V , W)$

$F = \{ S \rightarrow T , V \rightarrow SW , T \rightarrow U \}$

حل:

$$V \to SW \quad \Rightarrow V \to S \, , V \to W$$

$$V \to S \, , S \to T \quad \Rightarrow V \to T$$

$$V \to T \, , T \to U \quad \Rightarrow V \to U$$

بنابراین V ، همه صفتهای دیگر را می دهد، یعنی کلید کاندید است.

■

مثال

رابطه $R(A,B,C,D,E,F,G)$ با وابستگی های تابعی F به صورت زیر مفروض است. کلید اصلی R کدام است؟

$$F = \{ABD \to EG, C \to DG, E \to FG, AB \to C, G \to F\}$$

حل:

$$C \to DG \Rightarrow \begin{cases} C \to D \\ C \to G \end{cases}$$

$$\left.\begin{matrix} AB \to C \\ C \to D \\ C \to G \\ G \to F \end{matrix}\right\} \Rightarrow \begin{matrix} AB \to C \\ AB \to D \\ AB \to G \\ AB \to F \end{matrix}$$

$$\left.\begin{matrix} ABD \to EG \\ AB \to D \end{matrix}\right\} \Rightarrow AB \to EG \Rightarrow \begin{cases} AB \to E \\ AB \to G \end{cases}$$

نشان دادیم که از AB می توان به همه صفت ها رسید، در نتیجه کلید است.

■

مثال

رابطه R(A,B,C,D,E) با وابستگی های تابعی زیر را در نظر بگیرید. تعداد کلیدهای کاندید رابطه چند تا است؟

$$A \to B, B \to C, C \to A, D \to E, E \to D$$

حل:

از وابستگی های داده شده می توان نتیجه گرفت:

$$A \to B, B \to C \Rightarrow A \to C$$
$$B \to C, C \to A \Rightarrow B \to A$$

از $A \to B$ و $A \to C$ و $B \to A$ و $C \to A$ مشخص می شود که A وB و C مانند هم می باشند. از $E \to D$ و $D \to E$ نیز مشخص می شود که E و D مانند هم هستند. بنابراین A و B و C در یک گروه و E وD در گروه دیگر می باشند. در تعیین کلید کاندید باید از هر گروه یک عضو در نظر گرفته شود، بنابراین تعداد کلید کاندید برابر است با: $3 \times 2 = 6$

■

مثال

رابطه R(A,B,C,E,F) با مجموعه وابستگی های FD مفروض است. کلید کاندید در رابطه R کدام است؟

$$\{A \to BE, C \to F, B \to C, B \to E, DB \to E\}$$

حل:

از A می توان به همه رفت. بنابراین کلید کاندید است:

$$A \to BE \Rightarrow \begin{cases} A \to B \\ A \to E \end{cases}$$

$$B \to C, C \to F \Rightarrow B \to F$$

$$A \to B, B \to F \Rightarrow A \to F$$

$$A \to B, B \to C \Rightarrow A \to C$$

■

وابستگی چند مقداری (MVD)

در رابطه(R (X , Y , Z با صفات ساده یا مرکب X , Y , Z می گوییم که Y با X وابستگی تابعی چند مقداری دارد و نمایش می دهیم$X \to \to Y$ ، اگر به یک مقدار X ، مجموعه ای از مقادیر Y متناظر باشد.

تعریف دوم وابستگی چند مقداری

در رابطه R ، صفت Y با X وابستگی تابعی چند مقداری دارد اگر و فقط اگر مجموعه مقادیر Y متناظر با یک مقدار از جفت (X,Z)در R فقط به مقدار X بستگی داشته باشد و وابسته به Z نباشد. یعنی مجموعه مقادیر Y فقط با تغییر X ، تغییر کند.

R

X	Y	Z
X1	Y1 Y2 Y3	Z1
X1	Y1 Y2 Y3	Z2

وجود وابستگیMVD ، موجب بروز آنومالی هایی در رابطه R می شود و بهتر است آن را به دو رابطه R1 (X , Y) , R2 (X , Z) تجزیه کنیم.

وابستگی تابعیY به X ، حالت خاصی از وابستگی تابعی چند مقداریY به X است که در آن مجموعه مقادیرY متناظر با یک مقدارX ، یک عنصر دارد.

در یک رابطه با سه صفت خاصه، وابستگیMVD به صورت جفت وجود دارد، مثلاً در R (A,B,C) ، وابستگی$A\rightarrow\rightarrow B$ برقرار است، اگر و فقط اگر وابستگی $A\rightarrow\rightarrow C$ برقرار باشد و می نویسیم:

$A\rightarrow\rightarrow B\mid C$

قواعد آرمسترانگ در مورد وابستگی چند مقداری

در رابطه R (A , B , C , …) :

۱- اگر $A\rightarrow B$ ، آنگاه $A\rightarrow\rightarrow B$

۲- اگر $A\subset B$ ، آنگاه $A\rightarrow\rightarrow B$

۳- اگر $A\rightarrow B$ ، آنگاه $AC\rightarrow\rightarrow BC$

۴- اگر $A\rightarrow\rightarrow B$ ، آنگاه $A\rightarrow\rightarrow R(H)-B-A$

۵- اگر $A\rightarrow\rightarrow B$ و $A\rightarrow\rightarrow C$ ، آنگاه $A\rightarrow\rightarrow BC$

۶- اگر $A\rightarrow\rightarrow B$ و $B\rightarrow\rightarrow C$ ، آنگاه $A\rightarrow\rightarrow C-B$

۷- اگر $A\rightarrow\rightarrow B$ و $A\rightarrow\rightarrow C$ ، آنگاه $C\rightarrow\rightarrow B\cap A$

۸- اگر $A\rightarrow\rightarrow B$ و $A\rightarrow\rightarrow C$ ، آنگاه $A\rightarrow\rightarrow (B-C)\ ,\ A\rightarrow\rightarrow (C-B)$

۹- اگر $D,A\rightarrow\rightarrow B\subset C$، آنگاه $(A,D)\subset(B,C)$

وابستگی پیوندی (JD)

رابطه R وابستگی پیوندی به n پرتوش دارد، اگر و فقط اگر R حاصل پیوند n پرتوش باشد و نه کمتر. این وابستگی را به صورت $R = JD*(R_1, R_2, ..., R_n)$ نمایش می دهیم که $R_1 ... R_n$ پرتوهای رابطه R می باشند.

مثال

رابطه SPJ (SP , PJ , JS) ، وابستگی پیوندی به ۳ پرتوش دارد و به صورت SPJ = JD * (SP , PJ , JS) نمایش داده می شود. در واقع اگر پرتوهای این رابطه را یک بار روی صفات P# , S# و بار دیگر روی صفات خاصه P# , J# بدست آوریم و آنها را با یکدیگر JOIN کنیم و سپس نتیجه را با رابطه حاصل از پرتو روی صفات S# و J# و , Join کنیم، حاصل همانSPJ خواهد بود و هیچ سطری اضافه یا کم نخواهد شد.

مراحل کار:

SPJ

S#	P#	J#
S1	P1	J2
S1	P2	J1
S2	P1	J1
S1	P1	J1

⇒

SP

S#	P#
S1	P1
S1	P2
S2	P1

PJ

P#	J#
P1	J2
P2	J1
P1	J1

⇒

SPJ

S#	P#	J#
S1	P1	J2
S1	P1	J1
S1	P2	J1
S2	P2	J2
S2	P1	J1

حال اگر حاصل پیوند را با پرتو JS الحاق کنیم این تاپل افزونه از بین می رود و همان رابطه اول بدست می اید:

S#	P#	J#
S1	P1	J2
S1	P1	J1
S1	P2	J1
S2	P2	J2
S2	P1	J1

J#	S#
J2	S1
J1	S1
J1	S2

⇒

S#	P#	J#
S1	P1	J2
S1	P2	J1
S2	P1	J1
S1	P1	J1

آزمون

۱- رابطه R(A,B,C,D,E) و وابستگی های تابعی (Functional Dependency) زیر را در نظر بگیرید. فرض کنید نمی دانیم □ چیست و □ می تواند هر زیر مجموعه غیر تهی از صفات R باشد. کدام یک از گزاره های زیر مستقل از □ صحیح است؟ (دولتی ۹۰)

$A \to BC \ , \ CD \to E \ , \ \square \to D$

الف- هر کلید کاندید R شامل A است.

ب- هیچ کلید کاندیدی از R شامل C نیست.

ج- بعضی از کلیدهای کاندید R شامل C هستند و بعضی دیگر شامل C نمی باشند.

۱) فقط الف ۲) فقط ب ۳) الف و ب ۴) الف و ج

۳-۱) در رابطه R(A,B,C,D,E) با وابستگی های تابعی داده شده، هر کلید کاندید شامل A است و هیچ کلید کاندیدی شامل C نیست.

۲- رابطه R(A,B,C,D,E) و وابستگی های تابعی زیر را در نظر بگیرید. کدام گزینه کلید رابطه است؟ (دولتی ۸۹)

$A \to B \ , \ AB \to CD \ , \ D \to ABC$

۱) AD ۲) AE ۳) AB ۴) ABD

۳-۲) از $A \to B$ و $AB \to CD$ نتیجه می گیریم که $A \to CD$ (معادل با $A \to C, A \to D$). بنابراین از A به همه صفت ها به غیر از E می توان رسید. در نتیجه AE کلید می باشد.

۳- اگر در جدول $ST(\underline{S\#, T\#}, Date, Time, Code)$ **داشته باشیم** $Code \to T\#$**، آنگاه این جدول چند نامزد کلیدی (Candidate Key) دارد؟ (دولتی ۸۸)**

۱) یک ۲) دو ۳) سه ۴) چهار

۳-۲) رابطه دارای دو کلید کاندید است:

الف- (S#,T#) : همان کلید اصلی

ب- (S#,Code) : با توجه به وابستگی $Code \to T\#$

۴- کدام یک از گزینه های زیر نادرست است؟ **(IT – دولتی ۹۰)**

۱) در SQL عملگرهای $(\neq \text{all})$ و not in هم ارز هستند.

۲)رابطه R با وابستگی تابعی F در BCNF است اگر و فقط اگر رابطه R با وابستگی های تابعی F^+ (بستار F) در BCNF باشد.

۳) در رابطه ای با اسکیمای R(A,B,C,D) به طور منطقی می توان از وجود وابستگی چند مقداری $A \twoheadrightarrow BC$ وابستگی های منطقی $A \twoheadrightarrow C$ و $A \twoheadrightarrow B$ را نتیجه گیری کرد.

۴) دو مجموعه وابستگی های زیر هم ارز(معادل) نیستند:

$$F_1 = \{AB \to E, A \to CD, A \to E, E \to D, D \to A\}$$
$$F_2 = \{AB \to D, E \to AD, A \to CD, D \to AE, EC \to B\}$$

۳-۴) از $A \twoheadrightarrow C$ و $A \twoheadrightarrow B$ می توان نتیجه گرفت: $A \twoheadrightarrow BC$ ولی عکس این رابطه برقرار نمی باشد.

فصل ۷:

نرمال تر سازی رابطه ها

هر رابطه هر چند نرمال نیز ممکن است در عملیات درج، حذف و بهنگام سازی مشکلاتی داشته باشد. بنابراین بهتر است یک رابطه نرمال را تا حد امکان نرمال تر کرد.

آنومالی

دشواری و وضع غیرعادی را آنومالی می گویند. مثلاً وقتی که سطری را حذف می کنیم و پی آمد آن اطلاعات ناخواسته ای نیز حذف شود. یا مقدار صفتی را برای یک سطر تغییر می دهیم در حالیکه در سطرهای دیگر نیز امکان دارد نیازبه تغییر داشته باشد. در واقع آنومالی در عملیات ذخیره سازی به هر یک از سه حالت زیر گفته می شود:

۱- بروز پیامد بد، بعد از انجام یک عمل

۲- عدم امکان انجام یک عمل

۳- بروز اضافه کاری در انجام یک عمل

مثال

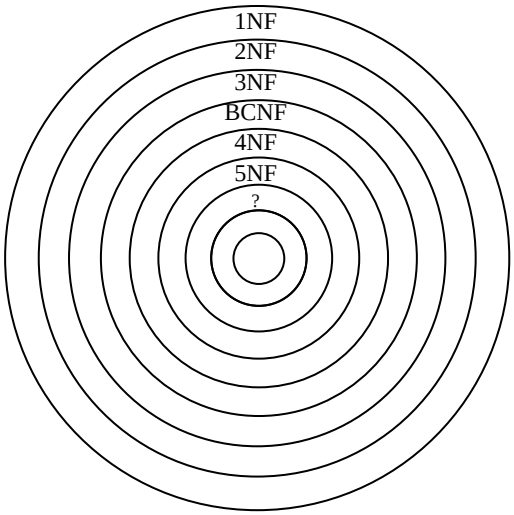
رابطه R در عملیات ذخیره سازی دارای آنومالی است. وجود پدیده افزونگی موجب آنومالی هایی در عملیات درج، حذف و بهنگام سازی شده است. مثلا اگر شهرS1 را ازC2 بهC1 تغییر دهیم، این عمل باید در تمام سطرهایی که s1 وجود دارد نیز انجام شود (فزونکاری) و در واقع برای جلوگیری از ناسازگاری باید بهنگام سازی منتشر شونده انجام شود. در حذف نیز آنومالی دارد. مثلاً با حذف اطلاع "S3 ازP2 به تعداد ۲۰۰ تا تهیه کرده است" این اطلاع کهS3 ساکنC3 است نیز حذف می شود.
همچنین در درج نیز آنومالی دارد، مثلاً نمی توان اطلاع" تهیه کننده S5 در شهرC5 ساکن است" را درج کرد، چون باید بدانیم چه قطعه ای را تهیه کرده است. کلید اصلی (S# , P#) است و طبق قاعده جامعیت موجودیتی هیچ جزء کلید اصلی نباید تهی باشد.

S#	STATUS	CITY	P#	QTY
S1	20	C2	P1	300
S1	20	C2	P2	200
S2	10	C3	P1	300
S3	10	C3	P2	200
S4	20	C2	P5	400

صورت های نرمال (NORMAL FORMS)

توسط کاد در ابتدا سه فرم نرمال ارائه شد و سپس فرم های دیگر توسط افراد دیگر ارائه شد. ایـن فـرم هـا عبارتنـد از: 1NF ، 2NF ، 3NF، BCNF، 4NF و 5NF. که هر یک از صورتهای نرمال، از قبلی نرمالتر هستند.

شکل زیر رابطه بین صورتهای نرمال را نشان می دهد:



از جمله مواردی که شکل بالا بیان می کند:

۱- از مجموعه رابطه های نرمال، زیر مجموعه ای1NF است و از مجموعه رابطه های1NF ، زیر مجموعـه ای2NF اسـت والـی آخر.

۲- فقط بعضی از رابطه های1NF در فرم 2NF هستند، اما تمام رابطه های2NF در فرم 1NF می باشند.

۳- BCNF از 3NF قوی تر است.

تعریف صورتهای نرمال

1NF	هر صفت خاصه در هر تاپل، تک مقداری باشد.
2NF	هر صفت خاصه غیرکلید با کلید اصلی، وابستگی تابعی کامل داشته باشند.
3NF	هر صفت خاصه غیر کلید با کلید اصلی، وابستگی تابعی بی واسطه داشته باشد.
BCNF	هر دترمینان، کلید کاندید باشد.
4NF	وابستگی تابعی چند مقداری وجود نداشته باشد.
5NF	تمام وابستگی های پیوندی، ناشی از کلیدهای کاندید باشد.

تذکر: هر رابطه نرمال،1NF است.

مزایای نرمالترسازی

۱- کاهش بعضی از آنومالی ها

۲- کاهش بعضی از انواع افزونگی

۳- تسهیل اعمال بعضی از قواعد جامعیت

۴- ارائه یک طرح بهتر و واضح تر با کمترین اختلاط اطلاعات

معایب نرمالترسازی

۱- بروز فزونکاری در سیستم در عمل بازیابی

۲- ایجاد نوعی افزونگی

۳- زمانگیر بودن فرایند نرمالترسازی

۴- مشکل شدن تصمیم گیری ها در تعدد تجزیه ها در مواردی

رابطه 1NF

هر رابطه نرمالی1NF است، اما رابطه1NF ای که2NF نیست دارای آنومالی هایی می باشد. به طور نمونه جدول FIRST را در نظر می گیریم:

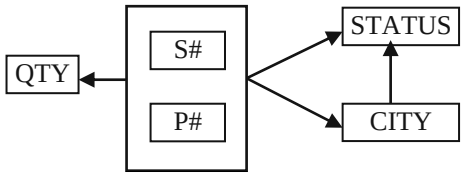
FIREST (S# , P# , STATUS , CITY , QTY)

کلید اصلی این رابطه، صفت مرکب (S#, P#) است و فرض کرده ایم که وابستگی CITY → STATUS وجود داشته باشد. یعنی وضعیت یک تهیه کننده از طریق شهر او، تعیین می شود. قبلاً دیدیم که این رابطه در درج، حذف و بهنگام سازی دارای آنومالی می باشد که برای رفع این آنومالی ها باید آن را به دو رابطه تجزیه کرد.

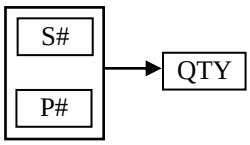
SECOND (S# , STATUS , CITY) و SP (S# , P# , QTY)

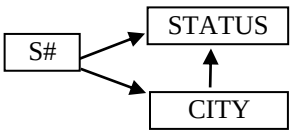
با بررسی رابطه های SECOND وSP مشخص می شود که آنومالی های موجود درFIRST در آنها وجود ندارند، یعنی در سطح نرمالتری از FIRST قرار دارند.

قبل از تجزیه:



بعد از تجزیه:



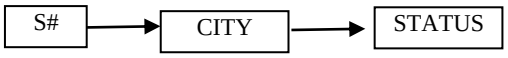


رابطه1NF ای که2NF نیست، حتما دارای کلید اصلی مرکب است.

رابطه 2NF

رابطه های حاصل از تجزیه FIRST هر دو در صورت دوم نرمال قرار دارند. رابطه SECOND هنوز دارای آنومالی هایی می باشد و علت این مشکلات در این است که وابستگیSTATUS به #S از یک سو کامل است و از سوی دیگر STATUS از طریقCITY نیز با #S وابستگی دارد.

یعنی بینSTATUS و#S وابستگی با واسطه وجود دارد.



برای رفع این مشکل رابطه SECOND را به دو رابطه زیر تجزیه می کنیم:

SC (S# , CITY)
CS (CITY , STATUS)

این دو رابطه آنومالی هایSECOND را ندارند.

رابطه3NF

ابتدا FIRST را به دو رابطه SECOND و SP تجزیه کردیم. رابطه SECOND در صورت 3NF نبود، چون صفت غیر کلید STATUS با کلید اصلی #S ، وابستگی باواسطه داشت و این برخلاف تعریف صورت3NF است. به همین علت آن را به دو رابطه تجزیه کردیم. اما رابطهSP در صورت سوم نرمال است. چون صفت غیر کلیدQTY با کلید اصلی (S# , P#) وابستگی باواسطه ندارد.

بنابراین رابطهFIRST به سه رابطه SC و CS وSP که همگی3NF هستند تجزیه شد.

در رابطه3NF ، تمام صفات غیرکلید متقابلاً با یکدیگر وابسته نمی باشند و وابستگی آنها با کلید اصلی، کاهش ناپذیر است.

مثال

رابطه R را در نظر می گیریم:

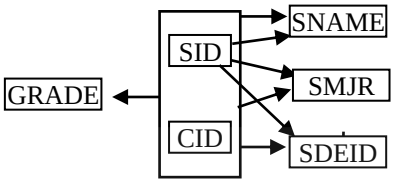
R (SID , CID , SNAME , GRADE , SMJR , SDEID)

قواعد جامعیت این رابطه عبارتند از:

۱- هر دانشجو یک نام دارد. ۲- هر دانشجو در یک درس یک نمره دارد.

۳- هر دانشجو در یک رشته تحصیل می کند. ۴- هر رشته تحصیلی در یک گروه آموزشی وجود دارد.

نمودار وابستگی های تابعی این رابطه به صورت زیر است:



رابطه R در فرم 1NF است و به علت وجود وابستگی جزئی 2NF نیست. این رابطه دارای آنومالی هایی می باشد و بنـابراین آن را به دو رابطه دیگر تجزیه می کنیم:

R1 (SID , SNAME , SMJR , SDEID) , **R2** (SID , CID , GRADE)

این دو رابطه در سطح2NF هستند و آنومالی های رابطه R را ندارند.

تذکر: علت وجود آنومالی در رابطه R ، وجود وابستگی جزئی در آن است، یعنی با وجود وابستگی تابعی

(SID , CID) → SNAME وجود وابستگی تابعی SID → SNAME نیز برقرار است.

تجزیه رابطه R به دو رابطه R1 وR2 یک تجزیه مطلوب محسوب می شود، چون با پیوند رابطه های R1 وR2 همـان محتـوای اطلاعاتی رابطه R بدست می آید و اطلاعات اضافی ایجاد نمی شود و همچنین تمام وابستگی های تابعی رابطه R محفوظ می ماند.

رابطه R2 در بالاترین سطح نرمال قرار دارد اما رابطه R1 دارای آنومالی های رابطه R می باشد و فقـط افزونگـی در آن کمتـر شده است. بنابراین رابطه R1 را به دو رابطه زیر تجزیه می کنیم.

R3 (SID , SNAME , SMJR) , **R4** (SMJR , SDEID)

این رابطه ها از R1 نرمالتر هستند، بنابراین3NF می باشند.

علت وجود آنومالی در رابطه R1 وجود وابستگی باواسطه در آن است:

SID → SMJR , SMJR → SDEID

یعنی SDEID از طریق SMJR یا SID مشخص می شود.

خلاصه: رابطه R در فرم1NF بود و به علت وجود وابستگی جزئی در فرم 2NF قرار نداشت. با حذف این وابستگی دو رابطـهR1 وR2 حاصل شد. این دو رابطه حداقل در فرم2NF قرار دارند و رابطه R2 در سطح 3NF نیز قرار دارد امـا رابطـه R1 بـه علـت وجود وابستگی با واسطه در سطح3NF قرار ندارد و آنرا به دو رابطه R3 وR4 تجزیه کردیم. در نهایت رابطه R به سه رابطه R2 وR3 وR4 تجزیه شد که هر سه رابطه در سطح3NF قرار دارند.

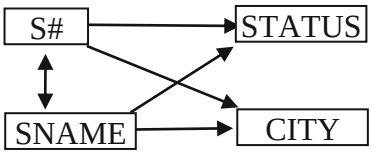
رابطه BCNF

رابطه R در سطح BCNF است اگر و فقط اگر هر دترمینان آن، کلید کاندید باشد.(هر صفت ای که صفت دیگر با آن وابستگی تابعی کامل داشته باشد دترمینان نام دارد. یعنی در وابستگی $A \rightarrow B$ ، صفتA دترمینان است.)

هر رابطه3NF، BCNF نیست. اگر رابطه تنها یک کلید کاندید داشته باشد و در مجموعه وابستگی های تابعی کاهش ناپذیر آن، وابستگی تابعی دیگری غیر از وابستگی های ناشی از کلید کاندید وجود نداشته باشد آنگاه رابطه BCNF هم هست. اگر رابطه3NF بیش از یک کلید کاندید داشته باشد و کلیدهای کاندید صفت مشترک نداشته باشند رابطه BCNF هم هست و اگر صفت مشترک داشته باشند ممکن استBCNF نباشد.

مثال

رابطه **S** (S# , SNAME , STATUS , CITY) با دو کلید کاندید S# , SNAME در فرمBCNF است. چون صفات S# , SNAME علاوه بر اینکه دترمینان های رابطه هستند، کلیدهای کاندید آن نیز می باشند.



تذکر: صفاتSTATUS , CITY به یکدیگر وابسته نمی باشند.

مثال

رابطه **R** (S# , SNAME , P# , QTY) با کلیدهای کاندید (S#,P#) و(SNAME , P#) آیا در فرم BCNF است؟

حل: خیر- چون S# , SNAME دترمینان هستند اما کلید کاندید نمی باشند. با تجزیه رابطه R به دو رابطه زیر، هر دو رابطه در سطحBCNF خواهند بود:

R1 (S# , SNAME) , **R2** (S# , P# , QTY)

مثال

رابطه **R** (A,B,C) با وابستگی تابعیB→C مفروض است. این رابطه دارای دو کلید کاندید(A,B) و (A,C) می باشد که دارای صفت مشترکA هستند. آیا این رابطه در فرم نرمال BCNF می باشد؟

حل : با توجه به وابستگی تابعی $C \rightarrow B$ معلوم می شود که C دترمینان است، در حالی که کلید کاندید نمی باشد. به همین علت رابطهBCNF نمی باشد.

می توان این رابطه را به دو رابطهBCNF ، **R1** (A , C) , **R2** (C , B) تجزیه کرد. یک بسط ممکن از رابطه R به صورت زیر است:

A	B	C
A1	B1	C1
A1	B2	C2
A2	B1	C1
A2	B2	C3

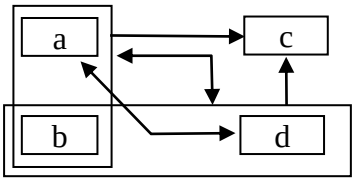
تجزیه ⇒

A	C
A1	C1
A1	C2
A2	C1
A2	C3

C	B
C1	B1
C2	B2
C3	B2

مثال

آیا رابطه 3NF زیر، BCNF هم هست؟



حل: خیر- چون a دترمینان است، اما کلیدکاندید نیست.

علت3NF بودن رابطه بالا:

۱- 1NF است، چون هم صفات اتومیک هستند.

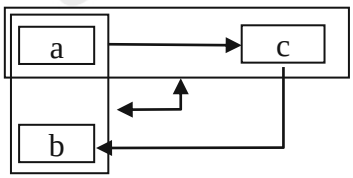
۲- 2NF است، چون وابستگی جزئی وجود ندارد. (وابستگی a→d موجب نقض این مورد نیست چون خود جزیی از کلید کاندید است.)

۳- 3NF است، چون وابستگی تابعی باواسطه ندارد.

در حالت وجود صفت مشترک در بین دو کلید کاندید، رابطه ممکن است3NF باشد اماBCNF نباشد.

مثال

آیا رابطه 3NF زیر،BCNF هم هست؟



حل: رابطه BCNF نیست، چون c دترمینان است اما کلید کاندید نیست. اگر این رابطه را به دو رابطه R1 (a , c) , R2 (c , b) که هر دوBCNF هستند تجزیه کنیم، وابستگی (a,b)→c از بین می رود. بنابراین بهتر است عمل تجزیه انجام نشود.

گاهی تجزیه یک رابطه3NF به دو رابطهBCNF، ضوابط ریسانن را ندارد. به همین علت اگر در اثر تجزیه وابستگی هایی حذف شود، در این صورت همان 3NF کفایت می کند و رابطه را نباید تجزیه کرد.

رابطه 4NF

در رابطهBCNF نیز احتمال وجود آنومالی خواهد بود. در این حالت رابطه را باید نرمالتر کرد.

مثال

رابطه غیرنرمال R1 (A,B,C) مفروض است:

A	B	C
A1	{B1, B2}	{C1, C2}
A2	{B1}	{C1, C3, C4}

این رابطه را به شکل نرمالش تبدیل می کنیم:

A	B	C
A1	B1	C1
A1	B1	C2
A1	B2	C1
A1	B2	C2
A2	B1	C1
A2	B1	C3
A2	B1	C4

رابطه R تمام کلید است و حداقل در سطحBCNF است، اما به علت وجود افزونگی دارای آنومالی هایی می باشد. در این رابطه افزودن اطلاع (A2 , B3) منجر به افزودن سه تاپل زیر خواهد شد:

(A2 , B3 , C1)
(A2 , B3 , C2)
(A2 , B3 , C3)

یعنی عمل در سطح تاپل به عملی در سطح مجموعه ای ازتاپلها تبدیل می شود.

با تجزیه این رابطه به دو رابطه زیر آنومالی های آن از بین می رود:

R1

A	B
A1	B1
A1	B2
A2	B1

و

R2

A	C
A1	C1
A1	C2
A2	C1
A2	C3
A2	C4

تذکر: در رابطه R ، دو وابستگی چند مقداری(MVD) به صورت $A \rightarrow\rightarrow B$ و $A \rightarrow\rightarrow C$ وجود دارد. در این رابطه وابستگی تابعی A→B وجود ندارد. یعنی تمامMVD های رابطه FD نمی باشند. بنابراین رابطه 4NF نیست. بنابراین رابطه R که BCNF است را به دو رابطه R1 و R2 تجزیه کردیم که 4NF می باشند.

رابطه 5NF

رابطه ای در سطح5NF است که در صورت وجود وابستگی پیوندی در آن این وابستگی ها ناشی از کلیدهای کاندید باشند.

تذکر: به5NF رابطه PJNF نیز می گویند.

مثال

رابطه تمام کلید SPJ(S# , P# , J#) در فرم5NF نمی باشند، چون دارای وابستگی پیوندی(JD) است که ناشی از کلید کاندید رابطه نمی باشد. یعنی کلید کاندید در همه پرتوهای این رابطه حضور ندارد.

وابستگی پیوندی در این رابطه برابر است با:

* (SP , PJ , JS)

یعنی اگر رابطه SPJ را به سه رابطه SP , PJ , JS تجزیه کنیم وسپس دو تجزیه مثلاً SP وPJ را با هم پیوند کنیم و در نهایت نتیجه را با JS پیوند بزنیم آنگاه حاصل همانSPJ خواهد بود، بدون اینکه تاپلی اضافه یا کم شده باشد.

مثال

رابطه S (S# , SNAME , STATUS , CITY) در سطح5NF است. چون وابستگی های پیوندی موجود در این رابطه ناشی ازکلید کاندید است. یعنی کلید کاندید در همه پرتوهای رابطه وجود دارد.

وابستگی های زیر ناشی از کلید کاندید S# است:

* ((S# , SNAME , STATUS , CITY) , (S# , CITY))

و وابستگی زیر ناشی کلید کاندید S# یا SNAME است:

* ((S# , SNAME) , (S# , STATUS) , (SNAME , CITY))

رابطه هایی کهBCNF باشند اما4NF نباشند و یا4NF باشند و5NF نباشند، نادرند.

اگر رابطه3NF باشد و تمام کلیدهای کاندید آن صفات ساده باشند، در اینصورت رابطه5NF است.

ضوابط ریسانن برای تجزیه مطلوب

تجزیه رابطه R به دو رابطه R1 و R2 مطلوب است، اگر R1 وR2 مستقل از یکدیگر باشند و شرایط زیر برقرار باشند:

۱- صفت مشترک در دو رابطه، حداقل در یکی از آنها کلید کاندید باشد.

۲- تمامFD های موجود در R یا در مجموعه FD های R1 وR2 موجود باشند و یا از این مجموعه FD ها منطقاً قابل استنتاج باشند.

در واقع با پیوند دو رابطه نباید سطر اضافی تولید شود و همچنین تجزیه رابطه وابستگی های تابعی آن را از بین نبرد.

مثال

کدام یک از تجزیه های رابطه(R (a , b , c با وابستگی های زیر مطلوب است؟

$a \to b \ , \ a \to c \ , \ b \to c$

الف- R1 (a , b) , R2 (b , c)

ب- R1 (a , b) , R2 (a , c)

ج- R1 (a , c) , R2 (b , c)

حل:

تجزیه الف، هر دو شرط قضیه ریسانن را دارد. تجزیه ب، مطلوبیت تجزیه الف را ندارد. در این تجزیه نمی توان وابستگی $b \to c$ را از دو وابستگی موجود در روابط R1 وR2 یعنی $a \to b$ و $a \to c$ منطقاً استنتاج کرد.

تجزیه ج نیز مطلوب نیست. چون نمی توان وابستگی $a \to b$ را از دو وابستگی $a \to c$ و $b \to c$ منطقاً استنتاج کرد. همچنین شرط اول قضیه ریسانن را نیز ندارد.

خلاصه: تجزیه الف هر دو شرط ریسانن را دارد. تجزیه ب فقط شرط اول را دارد و تجزیه ج هیچکدام از شرط ها را ندارد.

■

مثال

تجزیه جدول R(A,B,C,D,E,F) با وابستگی های تابعی $F = \{AB \to CF, C \to DE, E \to F\}$ به ۳ جدول هر یک با وابستگی های زیر، آیا تجزیه خوبی است؟

$F1 = \{AB \to C\}$

$F2 = \{C \to DE\}$

$F3 = \{E \to F\}$

حل:

یک تجزیه خوب است، چون از این وابستگی ها می توان وابستگی های جدول اصلی را استنتاج کرد.

■

مثال

تجزیه جدول R(A,B,C,D,E,F) با وابستگی های تابعی $F = \{AB \to CD, C \to DE, E \to F\}$ به سه جدول با وابستگی های زیر، آیا تجزیه خوبی است؟

$F1 = \{AB \to C\}$
$F2 = \{C \to DE\}$
$F3 = \{E \to F\}$

حل:

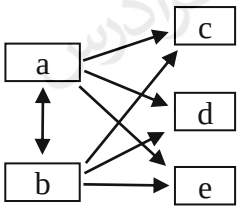
یک تجزیه خوب است، چون از این وابستگی ها می توان وابستگی های جدول اصلی را استنتاج کرد.

قضیه هیث

رابطه R(A,B,C) که در آن A و B و C سه مجموعه از صفات هستند، مفروض است. اگر A→B ، آنگاه می توان R را به دو رابطه R1 (A,B) و R2 (A,C) تجزیه کرد و این تجزیه خوب است و اگر A→B و B→C آنگاه تجزیه R1 (A,B) و R2 (B,C) خوب است.

مثال

آیا رابطه 3NF زیر، BCNF هم هست؟



حل: رابطه BCNF است، چون دارای دو دترمینان a و b است که هر دو کلید کاندید هستند.

مثال

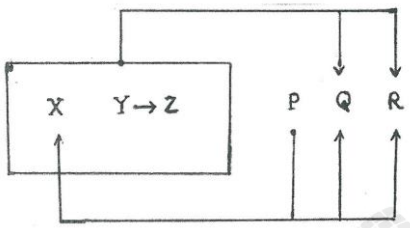
در رابطه R(A,B,C,D) صفات B و C چند مقداری هستند. اگر R را نرمال کنیم، کدام مورد نمی تواند کلید کاندید R باشد؟

۱) (A,B) ۲) (A,C) ۳) (B,C) ۴) (A,D)

حل: گزینه ۴.

آزمون

۱- رابطه A(X,Y,Z,P,Q,R) با وابستگی های تابعی(Function Dependency) شکل زیر را در نظر بگیرید. این رابطه در کدام سطح نرمال است؟ (دولتی ۹۰)



۱) 1NF ۲) 2NF ۳) 3NF ۴) BCNF

۱-۱) در رابطه 2NF باید تمام صفات غیرکلید با کلید اصلی وابستگی تابعی کامل داشته باشد و چون وابستگی $X \to Q$ یک وابستگی جزئی می باشد، یعنی Q با جزئی از کلید اصلی وابستگی دارد، رابطه داده شده 2NF نمی باشد، در نتیجه 3NF و BCNF هم نمی باشد.

۲- فرض کنید رابطه R با مجموعه وابستگی های تابعی F داده شده است. صفت $A \in R$ را ناهنجار می گوییم اگر و فقط اگر $(\exists x \subseteq R) : x \to R \notin F^+ \wedge A \in (x^+ - x)$ که x^+ نمایانگر بستار(closure) مجموعه صفات x تحت وابستگیهای تابع F می باشد. کدام یک از گزینه های زیر نادرست است؟(IT- دولتی ۹۰)

۱) اگر رابطه R در 3NF باشد، آنگاه هیچ صفت غیر کلیدی در رابطه R ناهنجار(abnormal) نخواهد بود.

۲) اگر رابطه R در BCNF باشد، آنگاه هیچ صفت ناهنجاری در R وجود نخواهد داشت.

۳) اگر هیچ صفت ناهنجاری در R وجود نداشته باشد، آنگاه رابطه R در BCNF خواهد بود.

۴) هیچکدام

۲-۴) با توجه به تعریف داده شده، همه موارد صحیح می باشند.

۳- رابطه R(A,B,C,D,E) و وابستگی های تابعی زیر را در نظر بگیرید:

$A \to B, BC \to D, E \to C$

اگر نگاشت R بر روی S(B,C,D,E) را در نظر بگیریم، کدام یک از وابستگی های تابعی در S برقرار است و شرط BCNF را برای S نقض نمی کند؟ **(IT- دولتی ۹۰)**

۱) $BE \to D$ ۲) $E \to C$ ۳) $BC \to D$ ۴) $B \to E$

۱-۳) رابطه ای در فرم BCNF است که هر دترمینان، کلید کاندید باشد. بنابراین چون در رابطه S مجموعه (B,E) کلید است، هر رابطه ای که دترمینان آن (B,E) باشد، شرط BCNF را نقض نمی کند.

۴- اگر در جدول $ST(S\#, T\#, Date, Time, Code)$**، وابستگی تابعی** $Code \to S\#$ **را داشته باشیم، آنگاه این جدول در کدام فرم نرمال نمی باشد؟** **(IT- دولتی۸۹)**

۱) BCNF ۲) 2NF ۳) 3NF ۴) CCNF

۱-۴) رابطه ای در فرم BCNF است که هر دترمینان در آن، کلید کاندید باشد. بنابراین با توجه به وابستگی $Code \to S\#$، Code دترمینان است ولی کلید کاندید نیست، رابطه BCNF نمی باشد.