



جزوه درس

# اصول طراحی پایگاه داده ها

تهیه کننده

**علی چوداری خسروشاهی**

مراجع

عنوان: **مفاهیم بنیادی پایگاه داده ها**

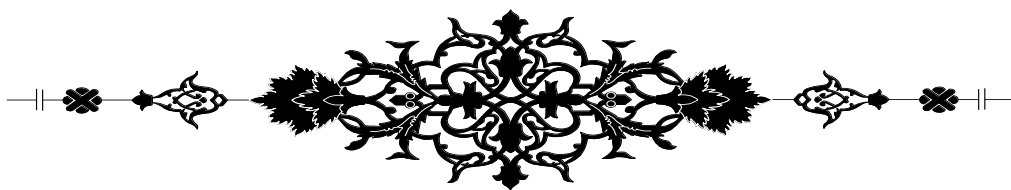
نویسنده: سید محمد تقی روحانی رانکوهی

عنوان: **بانک اطلاعاتی علمی کاربردی** (جلد ۱)

نویسنده: دکتر مصطفی حق جو

ویژه دانشجویان کارشناسی کامپیوتر - نرم افزار

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



## پیشگفتار

ما تلاش کرده ایم در این جزوه تمام سرفصلهای درس پوشش داده و مطالب مفیدی را در اختیار شما قرار دهیم تا شما با خواندن آن کلیاتی را از درس بدست آورید. در مورد این جزوه یادآوری این نکته مهم، ضروری است که این جزوه جهت کمک به دانشجویان در کاهش یادداشت برداری، ترسیم شکلها، مثالها در هنگام تدریس بوده است. بنابراین این در کنار این جزوه، هر دانشجویی مطابق با ذوق و سلیقه خود، جزوه ای دست نویس خواهد داشت که حاوی یادداشت ها به منظور تکمیل و تفهیم این جزوه است.

این جزوه، ادعای جایگزینی مراجع اصلی این درس را ندارد بلکه تنها خلاصه ای از مطالب مهم از مراجع درس است. در گردآوری این جزوه از مراجع زیر استفاده شده است:

- **Data Base System Concepts**

نویسنده: *Silberschatz*

- **مفاهیم بنیادی پایگاه داده ها**

نویسنده: سید محمد تقی روحانی رانکوهی

- **بانک اطلاعاتی علمی کاربردی (جلد ۱)**

نویسنده: دکتر مصطفی حق جو

در اینجا لازم است از تلاش تمام دانشجویانی که اینجانب را در تدوین این جزوه یاری نموده اند کمال تشکر را داشته باشم. همچنین در این جزوه احتمال اشتباه وجود دارد، به این جهت در مطالعه مطالب جزوه دقت کافی را داشته باشید. از دانشجویان عزیز تقاضا می شود در مطالعه مطالب آن دقت کافی را داشته باشند و وجود هرگونه ایراد، و همچنین نظرات و پیشنهادات خود را به آدرس پست الکترونیکی اینجانب اطلاع دهند.

Akhosroshahi@IAUT.ac.ir

Akhosroshahi@Gmail.com

باتشکر

علی چوداری خسروشاهی

## فهرست مطالب

۵	فصل اول
۵	۱- مقدمه درس
۵	1-1- DBMS
۵	DBA-۲-۱
۵	DBP-۳-۱
۵	۴-۱- تاریخچه
۶	۵-۱- پایگاه داده ها و عناصر اصلی محیط آن
۶	۱-۵-۱- تعریف پایگاه داده ها
۶	۱-۵-۲- ایرادات روش فایلینگ
۷	3-5-1- عناصر سیستم پایگاه داده
۸	فصل دوم
۸	۲- مدلسازی معنایی داده ها
۸	2-1- مدل سازی با روش ER
۸	۱-۱-۲- نوع موجودیت
۹	۲-۱-۲- صفت
۹	1-2-1- ساده یا مرکب
۹	۲-۲-۱-۲- تک مقداری یا چند مقداری
۹	۳-۲-۱-۲- صفت شناسه یا نا شناسه
۱۰	۴-۲-۱-۲- Derived یا مشتق یا ذخیره شده
۱۰	۳-۱-۲- نوع ارتباط
۱۰	4-1-2- نمودار ER
۱۲	۵-۱-۲- خصوصیت نوع ارتباط
۱۲	۱-۵-۱-۲- خصوصیت کلی
۱۲	۲-۵-۱-۲- وضعیت مشارکت در ارتباط
۱۲	۶-۱-۲- نوع ارتباط متابه نوع موجودیت
۱۳	۷-۱-۲- درجه ارتباط
۱۴	8-1-2- چندی ارتباط
۱۷	9-1-2- حد ارتباط
۱۹	۱۰-۱-۲- نوع موجودیت ضعیف
۱۹	۱۱-۱-۲- ارتباط به متابه نوع موجودیت ضعیف
۲۰	12-1-2- تجزیه و ترکیب
۲۰	۱۳-۱-۲- تخصیص و تعمیم
۲۲	۱۴-۱-۲- دسته بندی
۲۳	۱۵-۱-۲- تجمع
۲۹	فصل سوم

۲۹	۳- مفاهیم اساسی مدل رابطه‌ای .....
۲۹	۳-۱- تعاریف .....
۲۹	۳-۲- ویژگیهای رابطه .....
۳۰	۳-۳- کلید در مدل رابطه‌ای .....
۳۰	۳-۳-۱- ابر کلید .....
۳۰	۳-۳-۲- کلید کاندید .....
۳۰	۳-۳-۳- کلید اصلی .....
۳۰	۳-۳-۴- کلید دیگر .....
۳۱	۳-۳-۵- کلید خارجی .....
۳۱	۳-۴- قواعد جامعیت در مدل رابطه‌ای .....
۳۱	۳-۴-۱- جامعیت دامنه .....
۳۱	۳-۴-۲- جامعیت درون رابطه‌ای .....
۳۱	۳-۴-۳- جامعیت ارجاع .....
۳۲	فصل چهارم .....
۳۲	۴- طراحی پایگاه داده های رابطه ای .....
۳۳	۴-۱- تبدیل مدلسازی به طراحی منطقی (روش بالا به پایین) .....
۳۳	۴-۱-۱- روش تبدیل نمودار ER به رابطه ها .....
۳۳	۴-۱-۱-۱- حالت اول .....
۳۴	۴-۱-۱-۲- حالت دوم .....
۳۴	۴-۱-۱-۳- حالت سوم .....
۳۵	۴-۱-۱-۴- حالت چهارم .....
۳۶	۴-۱-۱-۵- حالت پنجم .....
۳۶	۴-۱-۱-۶- حالت ششم .....
۳۷	۴-۱-۱-۷- حالت هفتم : نمایش موجودیت ضعیف .....
۳۷	۴-۱-۱-۸- حالت هشتم : وجود صفت چند مقداری .....
۳۸	۴-۱-۱-۹- حالت نهم : بیش از یک ارتباط بین دو موجودیت .....
۴۱	فصل پنجم .....
۴۱	۵- جبر رابطه‌ای .....
۴۱	۵-۱- جداول نمونه .....
۴۳	۵-۲- عملوند یا نوع داده ای .....
۴۳	۵-۳- عملگر ها .....
۴۳	۵-۳-۱- دسته ی اول عملگرهای $\Pi$ (project) و $\sigma$ (select) .....
۴۵	۵-۳-۲- دسته ی دوم: عملگر های مجموعه ای .....
۴۶	۵-۳-۳- عملگر های پیوند .....
۴۶	۵-۳-۳-۱- عملگر ضرب دکارتی یا کارتین .....
۴۷	۵-۳-۳-۲- پیوند شرطی .....
۴۷	۵-۳-۳-۳- پیوند طبیعی .....

۴۷	..... نیم پیوند..... ۴-۳-۳-۵
۴۸	..... عملگر های دیگر..... ۴-۳-۵
۴۸	..... عملگر نام گذاری..... ۱-۴-۳-۵
۴۹	..... دستور جایگزینی..... ۲-۴-۳-۵
۴۹	..... عملگر تقسیم..... ۳-۴-۳-۵
۴۹	..... بهینه سازی..... ۵-۳-۵
۵۱	..... 5-3-6 تغییر داده های جداول.....
۵۱	..... افزودن داده به جداول..... ۱-۶-۳-۵
۵۱	..... حذف داده های از جداول..... ۲-۶-۳-۵
۵۱	..... تغییر داده های جداول..... ۳-۶-۳-۵
۵۲	..... فصل ششم.....
۵۲	..... 6- آشنایی با یک زبان رابطه ای.....
۵۲	..... 6-1- زبان SQL.....
۵۲	..... ۲-۶- انواع داده ای.....
۵۲	..... ۳-۶- دستور تعریف میدان.....
۵۳	..... ۴-۶- دستور تغییر دامنه.....
۵۳	..... ۵-۶- دستور حذف میدان.....
۵۳	..... ۶-۶- دستور ایجاد جدول.....
۵۴	..... ۷-۶- دستور تغییر جدول.....
۵۵	..... ۸-۶- دستور حذف جدول.....
۵۵	..... ۹-۶- دستور بازیابی.....
۵۶	..... ۱-۹-۶- مرتب سازی خروجی.....
۵۶	..... ۲-۹-۶- استفاده از عبارات اسکالر در دستور SELECT.....
۵۶	..... ۳-۹-۶- دگر نامی (قید AS).....
۵۷	..... ۴-۹-۶- توابع جمعی.....
۵۷	..... 5-9-6- دستور Like و Not like.....
۵۸	..... ۶-۹-۶- آزمون وجود هیچمقدار در یک ستون.....
۵۸	..... 7-9-6- Union All و Union.....
۵۸	..... ۸-۹-۶- گروه بندی.....
۵۹	..... 6-9-9- Having.....
۵۹	..... 6-9-10- Between.....
۶۰	..... ۱۱-۹-۶- پیوند در Select.....
۶۰	..... ۱۲-۹-۶- زیر دستور.....
۶۱	..... 6-9-13- دستور وجودی Exists و Not Exists.....
۶۲	..... ۱۰-۶- دستورات عملیات ذخیره سازی.....
۶۲	..... 6-10-1- دستور Update.....
۶۲	..... 6-10-2- دستور Delete.....
۶۳	..... ۳-۱۰-۶- دستور Insert.....

ضمیمه A ..... ۶۴

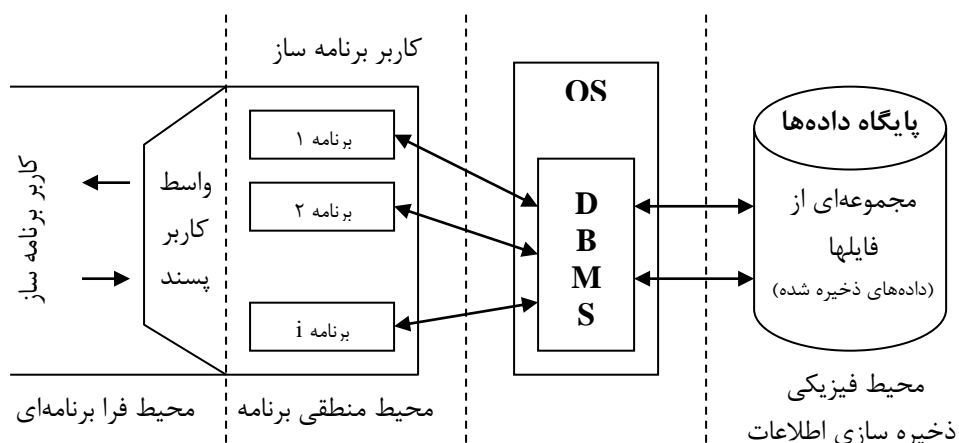
ضمیمه B ..... ۶۵

## فصل اول

## ۱- مقدمه درس

۱-۱- DBMS<sup>۱</sup>

DBMS یکی از سیستمهای ذخیره و بازیابی اطلاعات است و یک نرم افزار می باشد که کلیه دسترسی به داده ها از طریق DBMS انجام میشود. پایگاه دادهها تحت کنترل متمرکز نرم افزاری به نام سیستم مدیریت پایگاه دادهها (DBMS) ایجاد و بهره برداری میشوند. در حقیقت DBMS یک واسط ما بین دادهها و برنامه میباشد.



شکل ۱-۱: DBMS نرم افزار واسط ذخیره و بازیابی اطلاعات

۱-۲- DBA<sup>۲</sup>

یک شخص و یا یک تیم می باشد که وظیفه ی طراحی و سیاست گذاری پایگاه داده را بر عهده دارد.

۱-۳- DBP<sup>۳</sup>

یک شخص و یا یک تیم می باشد که وظیفه ی پیاده سازی و تصمیمات گرفته شده توسط DBA را بر عهده دارد.

## ۱-۴- تاریخچه

سیستم های پایگاه داده ای را میتوان به ۳ گروه تقسیم بندی شوند:

- سیستمهای پیش رابطه ای<sup>۴</sup>
- سیستمهای رابطه ای
- سیستمهای پسا رابطه ای<sup>۵</sup>

از دسته ی اول میتوان مدلهای سلسله مراتبی<sup>۱</sup> و مدل شبکه ای<sup>۲</sup> را در نظر گرفت و از دسته ی سوم میتوان مدل تابعی<sup>۳</sup> و مدل شی گرا<sup>۴</sup> را نام برد.

<sup>۱</sup> Data Base Management System (DBMS)

<sup>۲</sup> Data Base Administrator (DBA)

<sup>۳</sup> Data Base Programmer (DBP)

<sup>۴</sup> Pre-relational

<sup>۵</sup> Post relational

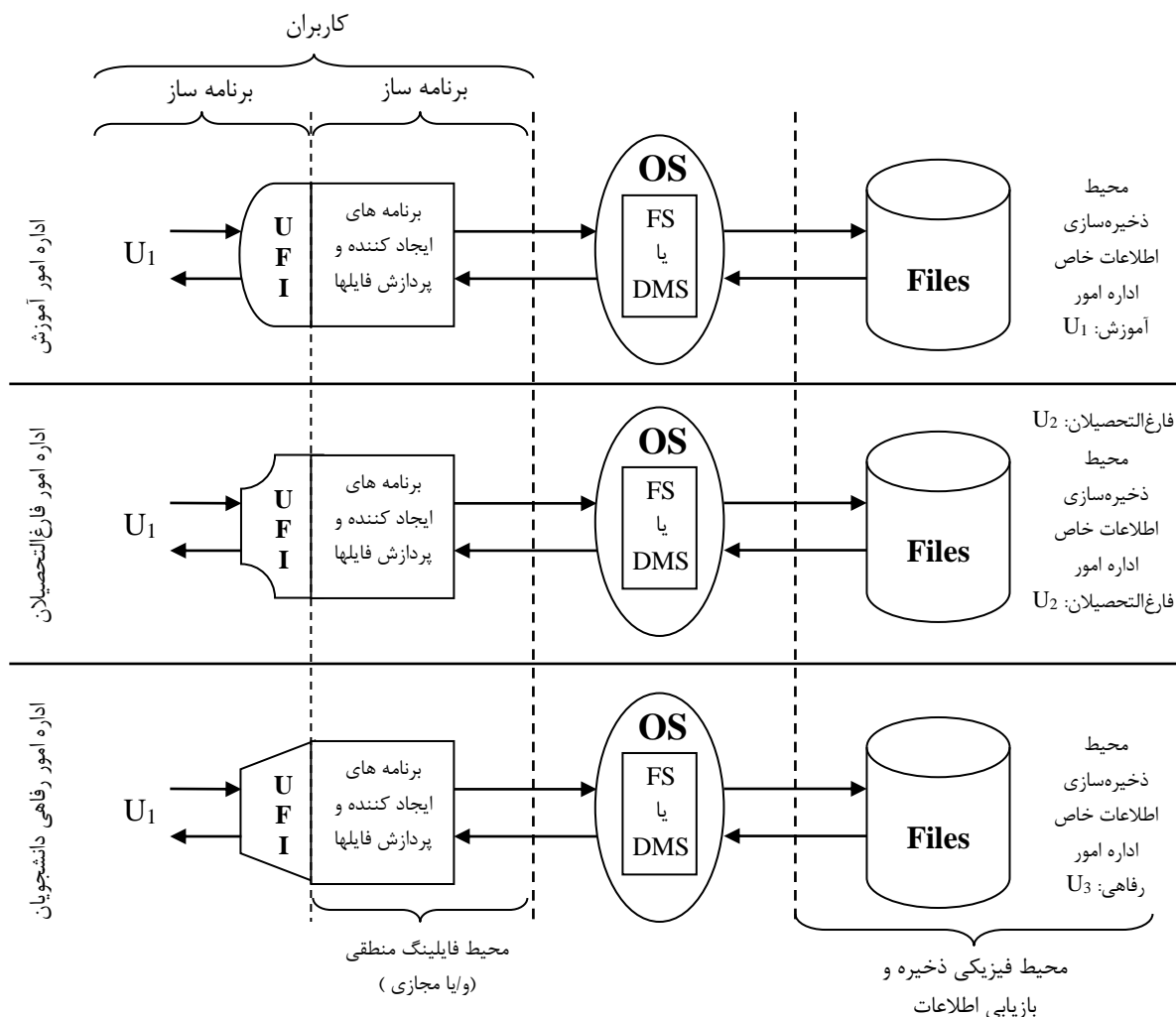


## ۵-۱- پایگاه داده ها و عناصر اصلی محیط آن

## ۵-۱-۱- تعریف پایگاه داده ها

پایگاه داده ها مجموعه ای است از داده های ذخیره شده و پایا به صورت مجتمع به هم مرتبط ، حد الا مکان با کمترین افزونگی ، تحت مدیریت یک سیستم کنترل متمرکز مورد استفاده ی یک یا چند کاربر ، به طور همزمان و اشتراکی.

قبل از مبحث پایگاه داده از سیستم فایلینگ برای ذخیره و بازیابی اطلاعات استفاده می شد. در حالت کلی می توان سیستم فایلینگ را به صورت شکل زیر در نظر گرفت:



شکل ۲-۱: نمایش ساده شده مشی فایلینگ

## ۵-۲- ایرادات روش فایلینگ

روش فایلینگ ایراد اتی را داشت که برخی از این ایراد ها عبارتند از:

- عدم وجود محیط مجتمع ذخیره اطلاعات و عدم وجود سیستم یک پارچه.

<sup>1</sup> Hierarchical model

<sup>2</sup> Network model

<sup>3</sup> Functional

<sup>4</sup> Object oriented

- عدم وجود سیستم کنترل متمرکز روی کل داده های سازمان.
- تکرار در ذخیره سازی اطلاعات.
- خطر بروز پدیده نا مطلوب ناسازگاری داده ها.
- مصرف نا بهینه ی امکانات سخت افزاری و نرم افزاری ، حجم زیاد برنامه سازی ، استفاده ی نا بهینه از مهارت و وقت تیمهای برنامه سازی.
- دشواری در گسترش سیستم های کاربردی و ایجاد کاربرد های جدید.
- وابسته بودن برنامه های کاربردی به محیط ذخیره سازی داده ها

### ۳-۵-۱- عناصر سیستم پایگاه داده

سیستم پایگاه داده از ۴ عنصر تشکیل شده است:

- سخت افزار
- نرم افزار
- کاربر
- داده

#### سخت افزار:

در سیستم پایگاه داده، سه دسته سخت افزار وجود دارد:

۱. سخت افزار ذخیره سازی داده ها
۲. سخت افزار پردازش گر
۳. سخت افزار ارتباط

نحوه ی اتصال این اجزا ۶ نوع معماری مختلف را ایجاد خواهد کرد:

۱. معماری متمرکز<sup>۱</sup>
۲. معماری مشتری-خدمتگذار
۳. معماری توزیع شده<sup>۲</sup>
۴. معماری با پردازش موازی<sup>۳</sup>
۵. معماری چند پایگاهی<sup>۴</sup>
۶. معماری موبایل<sup>۵</sup>

<sup>1</sup> Centralized

<sup>2</sup> Distributed

<sup>3</sup> Parallel processing

<sup>4</sup> Multidatabase

<sup>5</sup> Mobile database system

## فصل دوم

## ۲- مدلسازی معنایی داده ها

داده های ذخیره شدنی در پایگاه داده ها ابتدا باید در بالاترین سطح انتزاع مدلسازی معنایی شوند. مدلسازی معنایی داده ها یعنی ارایه ی مدلی از محیط عملیاتی با توجه به معنای داده ها، به کمک مفاهیمی مستقل از جنبه هایی مربوط به نمایش منطقی و نمایش فیزیکی داده ها. به مدلسازی معنایی گاهی طراحی ادراکی<sup>۱</sup> (مفهومی) گفته می شود.

## ۲-۱- مدل سازی با روش ER

در روش ER<sup>۲</sup> سه مفهوم معنایی وجود دارد و معنای داده های هر محیطی به کمک همین سه مفهوم نمایش داده می شود که عبارتند از:

- نوع موجودیت
- صفت
- نوع ارتباط

## ۲-۱-۱- نوع موجودیت

عبارتست از مفهوم کلی “شی”، “چیز”، “پدیده” و بطور کلی هر آن چه که می خواهیم در موردش “اطلاع” داشته باشیم و شناخت خود را در موردش افزایش دهیم، اعم از اینکه وجود فیزیکی یا ذهنی داشته باشد. هر نوع موجودیت از نظر کاربر نام و معنای مشخصی دارد.

نکته: هر نوع موجودیت نمونه ها<sup>۳</sup> یا مصادیق در خور جهان واقع دارد. باید سه ضابطه ی زیر را در تشخیص موجودیت ها در نظر بگیریم.

۱. یک نوع موجودیت از یک محیط، معمولاً نمونه هایی (بیش از یک نمونه) متمایز از یکدیگر دارد.
  ۲. یک نوع موجودیت معمولاً بیش از یک صفت دارد و کاربر به مجموعه ای از اطلاعات در مورد آن نیاز دارد.
  ۳. معمولاً حالت کنشگری (فاعلیت) یا کنش پذیری (مفعولیت) دارد.
- نکته: یک نوع موجودیت ممکن است قوی<sup>۴</sup> (مستقل) یا ضعیف<sup>۵</sup> (وابسته) باشد.

- موجودیت قوی موجودیتی است که مستقل از هر نوع موجودیت دیگر باشد و به خودی خود در محیط مطرح است.
- موجودیت ضعیف موجودیتی است که وجودش وابسته به یک نوع موجودیت دیگر است. اگر موجودیت قوی از مدل معنایی حذف شود موجودیت ضعیف هم حذف میشود.

نکته: در یک محیط موجودیت های زیادی وجود دارد اما از بین این موجودیت ها مدل سازی پایگاه داده باید با توجه به نیازهای اطلاعاتی کاربران انتخاب شود. بنابراین اساساً قبل از مدل سازی باید خواسته ها و نیازها به درستی تحلیل و برآورد شوند.

<sup>1</sup> Conceptual design  
<sup>2</sup> Entity Relationship  
<sup>3</sup> instance  
<sup>4</sup> Strong  
<sup>5</sup> Weak

با توجه به مطالبی که گفته شد یک نوع موجودیت ، خصوصیات زیر را خواهد داشت:

- نام
- معنای مشخص
- مجموعه ای از صفات
- مجموعه ای از نمونه ها
- حالت کنش گری یا کنش پذیری
- عدم وابستگی به یک نوع دیگر

### ۲-۱-۲-۲- صفت

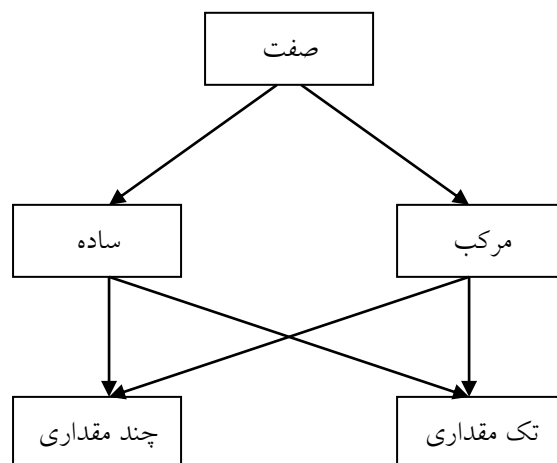
صفت در واقع خصیصه یا ویژگی یک نوع موجودیت است و هر نوع موجودیت مجموعه ای از صفات دارد که حالت یا وضعیت آن را توصیف میکند. هر صفت از نظر کاربر یک نام ، یک نوع و یک معنای مشخص دارد. یک فقره اطلاعات عبارت است از نام یک صفت و یک مقدار مطلوب متناسب به آن. صفت ها را می توان از چندین نظر رده بندی کرد:

#### ۲-۱-۲-۱- ساده یا مرکب

صفت ساده صفتی است که مقدار آن از لحاظ معنایی ساده یا اتمیک یا تجزیه نشدنی باشد. صفت مرکب صفتی است که از چندین صفت ساده تشکیل شده باشد، به صورتی که تجزیه شدنی باشد و اجزای حاصل از تجزیه خود صفات ساده باشند.

#### ۲-۱-۲-۲- تک مقداری یا چند مقداری

هر صفت ساده یا مرکب می تواند تک مقداری یا چند مقداری باشد صفت تک مقداری صفتی است که برای نمونه از یک موجودیت حداکثر یک مقدار می تواند داشته باشد. صفت چند مقداری صفتی می باشد که برای حداقل یک نمونه از موجودیت بیش از یک مقدار را می تواند بگیرد. به عنوان مثال شماره ی دانشجویی صفت تک مقداری است ولی صفت مدرک تحصیلی استاد صفت چند مقداری است، چون هر استاد چندین مدرک تحصیلی می تواند داشته باشد.



#### ۲-۱-۲-۳- صفت شناسه یا نا شناسه

صفت شناسه موجودیت ، صفتی است که دو ویژگی را داشته باشد،

- مقدار یکتایی داشته باشد، یعنی در هیچ دو نمونه از یک موجودیت مقدارش یکسان نباشد و آن صفت وجه تمایز ما بین موجودیت ها باشد. حداقل امکان طول مقادیرش کوتاه باشد.
- هیچ مقدار پذیر یا Null value: یعنی اینکه صفت میتواند مقدار ناشناخته ای داشته باشد. یعنی اینکه می توان مقدار صفت را مشخص نکرد، به عنوان مثال شماره تلفن برای دانشجو ممکن است دانشجو هیچ شماره ی تلفنی نداشته باشد، بر عکس نام دانشجو نمی تواند این وضعیت را داشته باشد، یعنی اینکه دانشجو باید حتما نام مشخص داشته باشد.

#### ۲-۱-۲-۴ Derived یا مشتق یا ذخیره شده

صفت ذخیره شده صفتی است که مقدارش در پایگاه داده ذخیره شده باشد. صفت مشتق صفتی است که در پایگاه داده ذخیره شده نباشد، بلکه بتوان مقدار آن را از سایر صفات محاسبه کرد، به عنوان مثال سن برای دانشجو صفت مشتق میباشد چون در پایگاه داده ذخیره نمیشود و می توانیم مقدار آن را از طریق تاریخ تولد محاسبه کنیم. با توجه به آنچه که گفته شده هر صفت جنبه های زیر را خواهد داشت:

- نام
- معنا
- دامنه مقادیر
- نوع مقدار
- واحد مقدار
- یک یا چند محدودیت ناظر به صفت
- طول مقدار
- کد نمایش مقدار

#### ۲-۱-۳-۳ نوع ارتباط

معمولا ما بین موجودیت ها ارتباطاتی وجود دارد نوع ارتباط عبارتست از تعامل<sup>۱</sup> ما بین  $n$  موجودیت ( $n \geq 1$ ) و ماهیتا نوعی بستگی بین انواع موجودیت ها می باشد. هر نوع ارتباط یک معنای مشخص دارد و با یک نام بیان می شود، همچنین می توان گفت نوع ارتباط عملی است که بین انواع موجودیت های جاری بوده، است و یا خواهد بود. به عنوان مثال در جملات زیر افعال بیان کننده ی ارتباط ما بین موجودیت ها می باشد:

دانشجو درس را انتخاب می کند.

استاد درس را تدریس می کند.

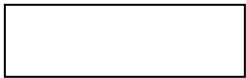

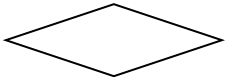
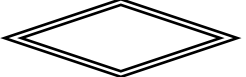
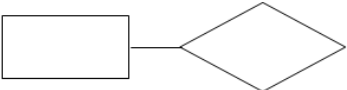
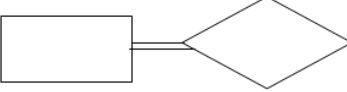
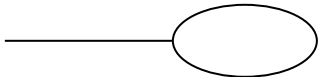
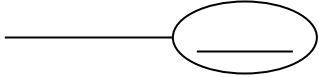
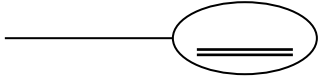

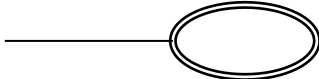
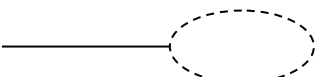
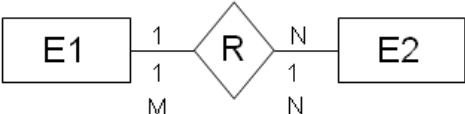
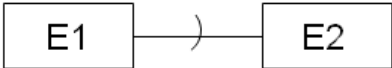
و مثالهای دیگر...

#### ۲-۱-۴-۴ نمودار ER

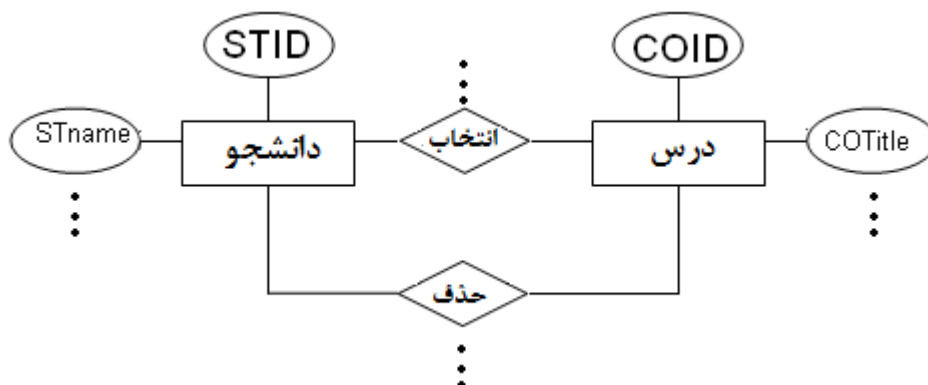
نموداری است که در آن سه مفهوم اساسی مدل ER، یعنی نوع موجودیت، صفت و نوع ارتباط نمایش داده می شوند.

جدول زیر نمادهای رسم نمودار ER و EER را نشان می دهد.

<sup>1</sup> Interaction \$

نماد	معنا
	نوع موجودیت
	نوع موجودیت ضعیف
	نوع ارتباط
	نوع ارتباط موجودیت ضعیف با قوی
	مشارکت نوع موجودیت در نوع ارتباط
	مشارکت الزامی
	صفت
	صفت شناسه اول
	صفت شناسه دوم (در صورت وجود)
	صفت شناسه مرکب (دو صفتی)
	صفت چند مقداری
	صفت مرکب
	صفت مشتق (مجازی یا محاسبه شدنی)
	چندی ارتباط R
	ارتباط گونه ای است از... E1 is a E2

ما بین دو موجودیت ممکن است بیش از یک ارتباط باشد به عنوان مثال دانشجو درس را انتخاب و حذف می کند. انتخاب و حذف کردن دو ارتباط ما بین دانشجو و درس می باشد علاوه بر این ارتباطات دیگری هم می تواند وجود داشته باشد. مثال زیر را می توان برای این دو ارتباط رسم کرد.



شکل ۱-۲: دو نوع ارتباط بین دو نوع موجودیت

### ۲-۱-۵- خصوصیت نوع ارتباط

#### ۲-۱-۵-۱- خصوصیت کلی:

۱. هر ارتباط یک نام دارد.
۲. هر ارتباط یک معنای مشخص دارد و این معنا با معنای هر ارتباط دیگر متفاوت است.
۳. هر ارتباط نمونه هایی دارد.

#### ۲-۱-۵-۲- وضعیت مشارکت در ارتباط

مشارکت یک نوع موجودیت و یک نوع ارتباط ممکن است الزامی یا اختیاری باشد. مشارکت یک نوع موجودیت در یک ارتباط را الزامی گوئیم اگر تمام نمونه های آن موجودیت در آن نوع ارتباط شرکت کنند، در غیر این صورت مشارکت اختیاری خواهد بود. شکل زیر الزامی بودن هر دو موجودیت را در انتخاب ارتباط نشان می دهد.



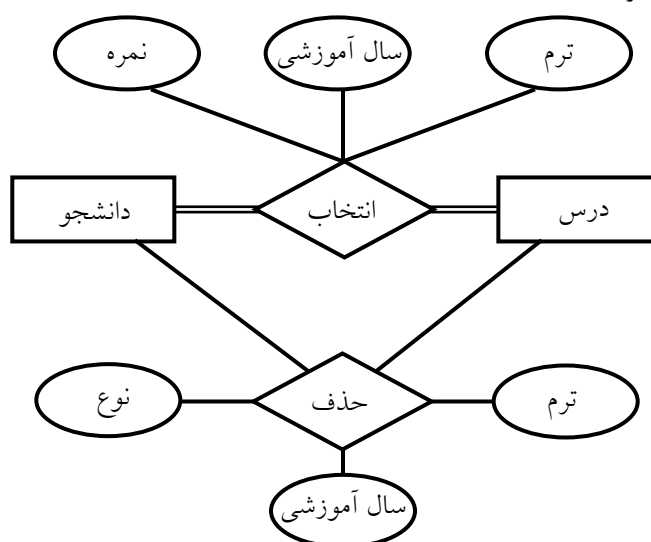
شکل ۲-۲: نمایش مشارکت الزامی

**نکته:** هر نوع موجودیت شرکت کننده در یک ارتباط یک نقش مشخص را ایفا میکند که معمولاً با یک عبارت فعلی بیان می شود، این نقش در واقع همان تاثیر عملی است که انجام می دهد.

#### ۲-۱-۶- نوع ارتباط مشابه نوع موجودیت

در حالت کلی می توان گفت ارتباط خود نوعی موجودیت می باشد چون لازمه ی موجود ارتباط وجود حداقل یک موجودیت می باشد پس منطقی می توان ارتباط را نوعی موجودیت ضعیف در نظر گرفت.

نکته: چون ارتباط خود نوعی موجودیت می باشد بنابراین می تواند شامل صفت یا صفاتی باشد گاهی به صفت ارتباط صفت توصیفی<sup>۱</sup> می گویند.



شکل ۳-۲: ارتباط می تواند صفت (صفات) داشته باشد

#### ۷-۱-۲- درجه ارتباط

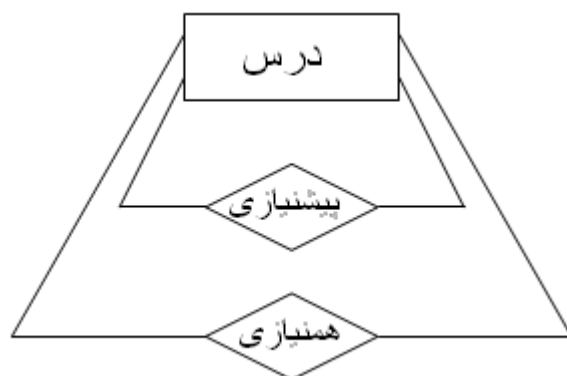
تعداد موجودیتهای شرکت کننده در ارتباط را درجه ی ارتباط گویند. جدول زیر انواع درجات را نشان می دهد.

درجه		تعداد شرکت کنندگان
فارسی	لاتین	
یگانی	unary	۱
دو گانی	binary	۲
سه گانی	ternary	۳
⋮	⋮	⋮
چند گانی	n-ary	N

در ارتباط درجه ۱ ارتباط یک موجودیت با خودش نشان داده می شود یعنی در حقیقت نحوه ی ارتباط یک نمونه از موجودیت با نمونه های دیگر از همان موجودیت بررسی میشود. به این گونه ارتباطات ارتباط با خود یا بازگشتی می گویند. در مثال زیر یک نمونه ارتباط با خود را می بینیم. در این شکل ارتباط پیش نیاز و هم نیاز از نوع ارتباط درجه ۱ می باشند.

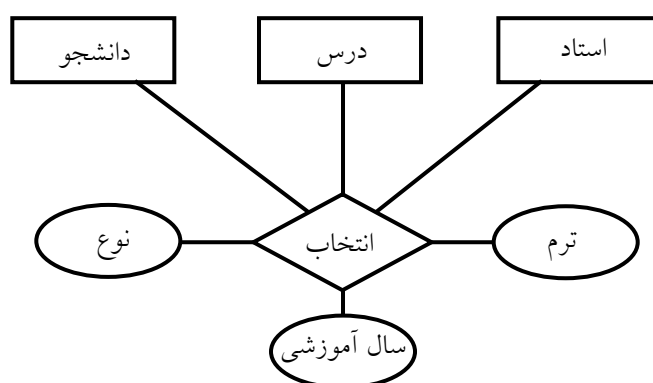
<sup>1</sup> Descriptive attribute





شکل ۴-۲: دو نوع ارتباط یک نوع موجودیت با خودش

چون در ارتباط فقط یک موجودیت دخالت دارد (درس) ارتباط درجه ۱ است و درس با نمونه های دیگر درس می تواند هم نیاز یا پیش نیاز باشد.



شکل ۵-۲: ارتباط بین سه موجودیت

معنی این ارتباط چنین است: "دانشجوی st ، درس c را با استاد pr ، انتخاب می کند".

## ۸-۱-۲- چندی ارتباط

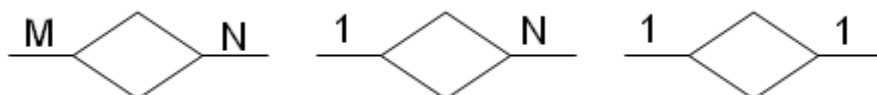
یک ارتباط از نظر چندی می تواند یکی از حالات زیر را داشته باشد.

۱. یک به یک<sup>۱</sup>

۲. یک به چند<sup>۲</sup>

۳. چند به چند<sup>۳</sup>

در نمودار ER این سه گونه تناظر را بصورت زیر نمایش می دهیم:



شکلهای زیر حالت های مختلف این تناظر را مابین دو مجموعه A و B نشان می دهند.

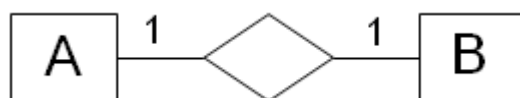
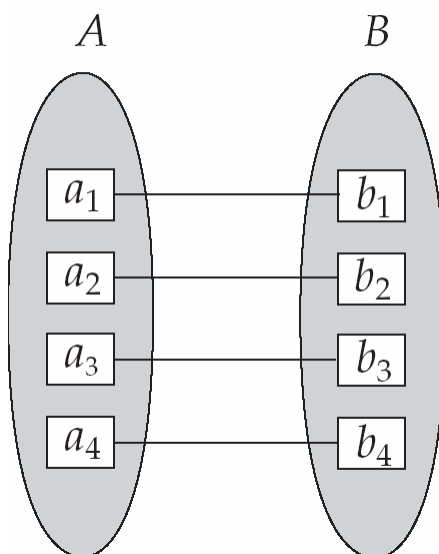
<sup>1</sup> one-to-one

<sup>2</sup> one-to-many

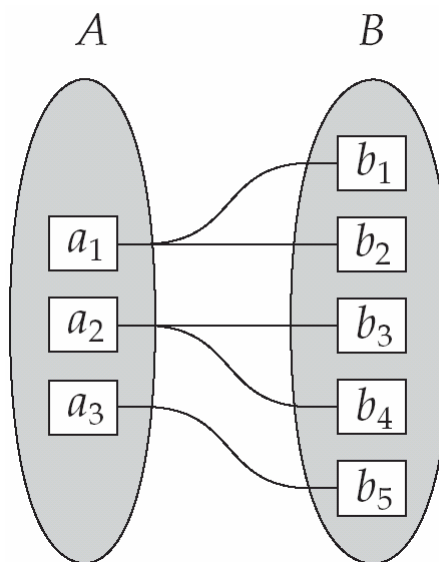
<sup>3</sup> many-to-many

اگر دو موجودیت E و F را داشته باشیم ارتباط E و F از نظر چندی ارتباط می تواند به یکی از صورت های زیر باشد.

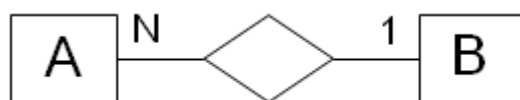
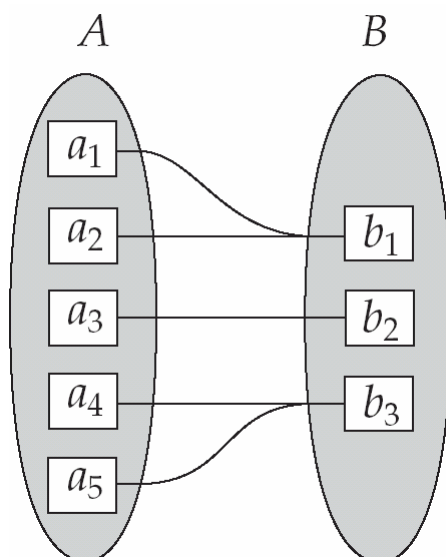
- در این حالت یک نمونه از E حد اکثر با یک نمونه از F ارتباط دارد و بر عکس (one-to-one).



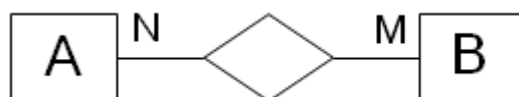
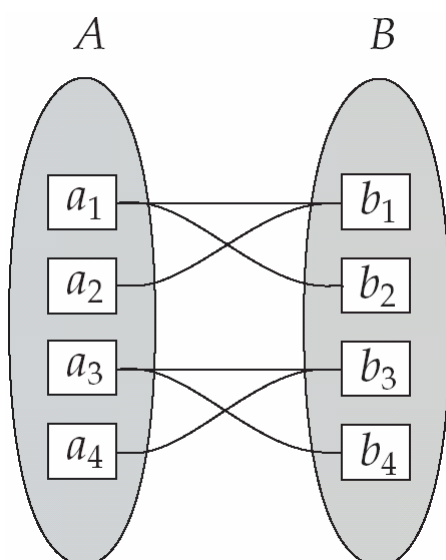
- در این حالت یک نمونه از E با چند نمونه از F ارتباط دارد و هر نمونه از F فقط با یک نمونه از E ارتباط دارد (one-to-many).



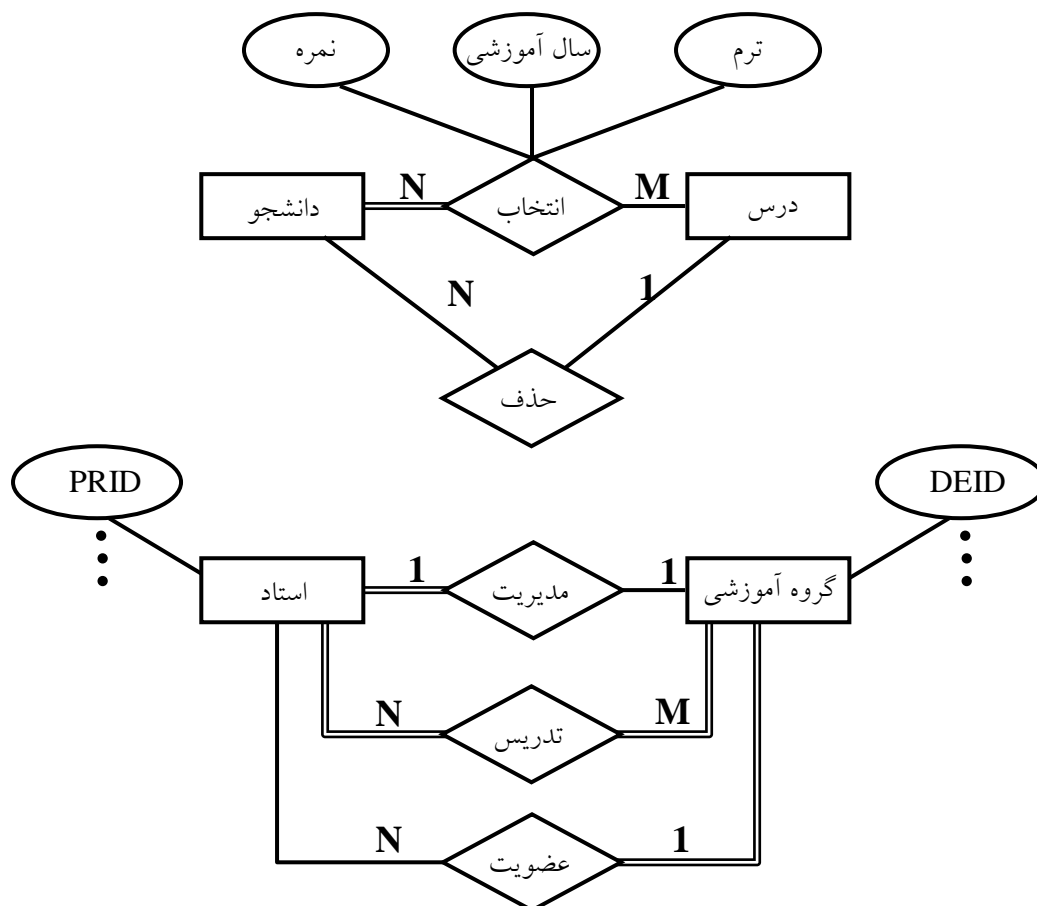
- در این حالت هر نمونه از E با یک نمونه از F اما هر نمونه از F با چند نمونه از E ارتباط دارد (many-to-one).



- در حالت چهارم هر نمونه از E با چند نمونه از F و هر نمونه از F با چند نمونه از E ارتباط دارد (-many to-many).



مثال:



شکل ۶-۲: نمایش چندی ارتباط

در مثال قبل ارتباط حذف از نوع یک به چند می باشد و مفهوم آن است که هر دانشجو می تواند یک درس را حذف کند و هر درس توسط چند دانشجو حذف می شود. همچنین شرکت استاد در مدیریت گروه آموزشی اختیاری می باشد اما هر گروه آموزشی باید توسط یک استاد مدیریت شود (الزامی).  
چندی نوع ارتباط براساس قواعد جاری در خرد جهان واقع موسوم به قواعد فعالیت ها تشخیص داده می شود یعنی چندی را براساس قوانین سیستم تشخیص می دهیم. ممکن است چندی یک ارتباط از یک سیستم به یک سیستم دیگر کاملاً متفاوت باشد.

## ۹-۱-۲- حد ارتباط

این روش دیگر برای نمایش چندی ارتباط می باشد حد یک ارتباط براساس جفت عدد صحیح به صورت (max,min) نشان داده می شود که در آن:

$$0 \leq \text{Min} \leq \text{Max} \quad , \quad 1 \leq \text{Max}$$

معنای این دو عدد این است که در هر لحظه هر نمونه ی e از موجودیت E باید حداقل در min و حداکثر در Max نمونه از ارتباط R شرکت داشته باشد. اگر min=0 باشد یعنی مشارکت غیر الزامی و اگر min>0 باشد یعنی مشارکت الزامی است. از طریق این اعداد می توان حتی حالت یک به یک بودن، یک به چند و چند به چند بودن را نیز تشخیص داد. این روش نمایش دقیقتر از روش قبل می باشد.

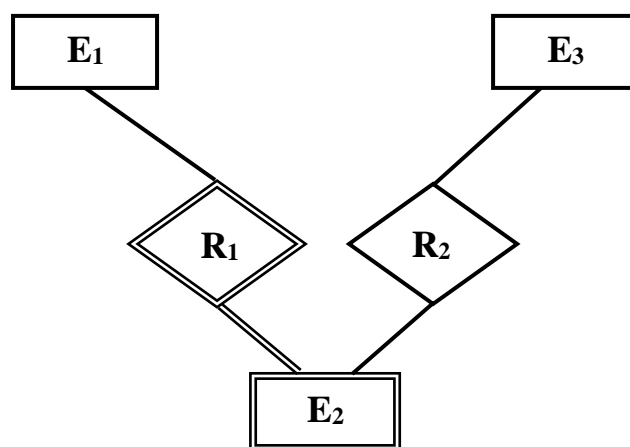
<sup>1</sup> Cardinality



• چندی

## ۱۰-۱-۲- نوع موجودیت ضعیف

موجودیت  $E$  ضعیف است اگر وجودش وابسته به یک نوع موجودیت دیگر مانند  $F$  باشد. به بیان دیگر مابین  $E$  و  $F$  وابستگی وجودی برقرار است، به گونه ای که اگر  $F$  حذف شود  $E$  نیز حذف خواهد شد. باید توجه داشت که هر چند موجودیت ضعیف شناسه ندارد اما یک صفت ممیزه دارد که به آن کلید جزئی نیز گفته می شود، نوع موجودیت ضعیف از نوع موجودیت قوی شناسه می گیرد. در واقع شناسه ی نوع موجودیت ضعیف ترکیب شناسه ی موجودیت قوی و صفت ممیزه ی موجودیت ضعیف است. همچنین موجودیت ضعیف خود می تواند موجودیت ضعیف دیگر نیز داشته باشد، در شکل زیر  $E_2$  موجودیت ضعیف و  $E_1$  موجودیت قوی آن میباشد، همچنین یک موجودیت قوی دیگر با نام  $E_3$  وجود دارد، که با  $E_2$  ارتباط  $R_2$  را دارد.



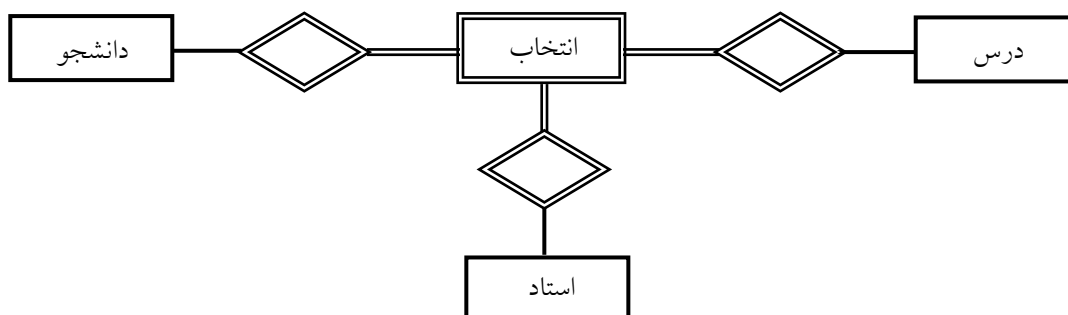
شکل ۹-۲: ارتباط موجودیت ضعیف با موجودیت قوی

نکات مهم در مورد نوع موجودیت ضعیف عبارتست از:

- درجه ی ارتباط معمولا ۲ است اما می تواند بیشتر هم باشد.
- چندی ارتباط همیشه ۱ به  $N$  است از طرف موجودیت قوی چندی ۱ و از طرف موجودیت ضعیف چندی  $N$ .
- موجودیت ضعیف از خود شناسه ندارد بلکه یک صفت ممیزه دارد.
- مشارکت نوع موجودیت ضعیف در ارتباط همیشه الزامی است.

## ۱۱-۱-۲- ارتباط به مثابه نوع موجودیت ضعیف

میتوانیم یک ارتباط را به صورت یک موجودیت ضعیف هم در نظر بگیریم، به عنوان مثال ارتباط درجه ی ۳ انتخاب که قبلا توضیح داده شد و با سه موجودیت دانشجو، درس و استاد در ارتباط می باشد، می توان با یک موجودیت ضعیف به صورت زیر نشان داد.

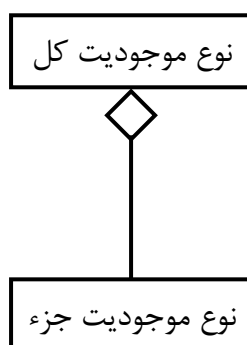


شکل ۱۰-۲: ارتباط به صورت نوع موجودیت ضعیف

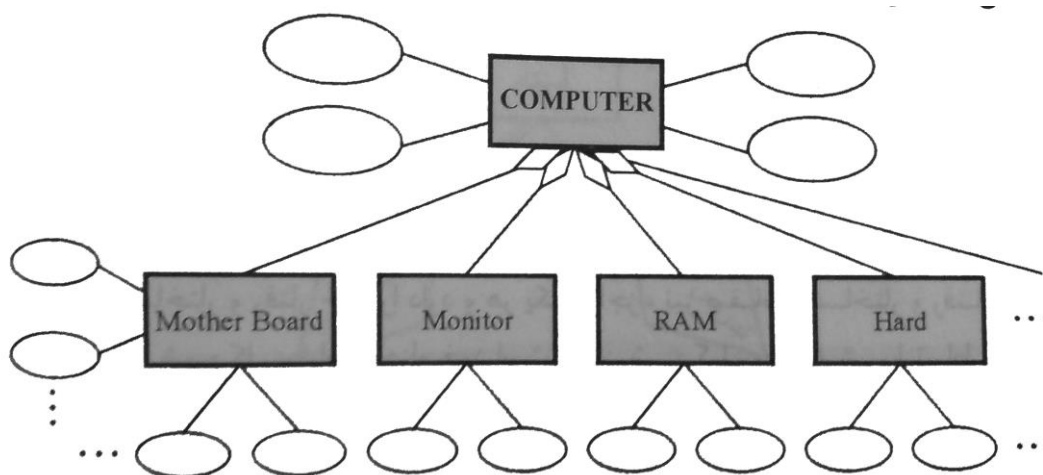
### ۱۲-۱-۲- تجزیه و ترکیب

تجزیه یا جدا سازی یعنی یک شی کل را به اجزای تشکیل دهنده ی آن تقسیم کنیم. شی کل شامل اجزای خود است و ما بین شی کل و اجزایش ارتباط شمول<sup>۱</sup> وجود دارد. این نوع ارتباط در روش EER ارتباط "جزیی است از..."<sup>۲</sup> گفته میشود.

نماد زیر برای نمایش ارتباط موجودیت کل و نوع موجودیت جز استفاده می شود.



مثال: در زیر یک نمونه از یک تجزیه و ترکیب را می بینید.



شکل ۱۱-۲: مثال تجزیه و ترکیب

### ۱۳-۱-۲- تخصیص و تعمیم

تخصیص عبارتست از بازشناسی گونه های خاص یک شی براساس یک ضابطه ی مشخص. می گوئیم بین هر زیر نوع موجودیت<sup>۳</sup> و زبر نوع<sup>۴</sup> آن ارتباط "هست یک..."<sup>۱</sup> وجود دارد. به بیان دیگر هر زیر نوع گونه ی خاصی است از حداقل یک زبر نوع.

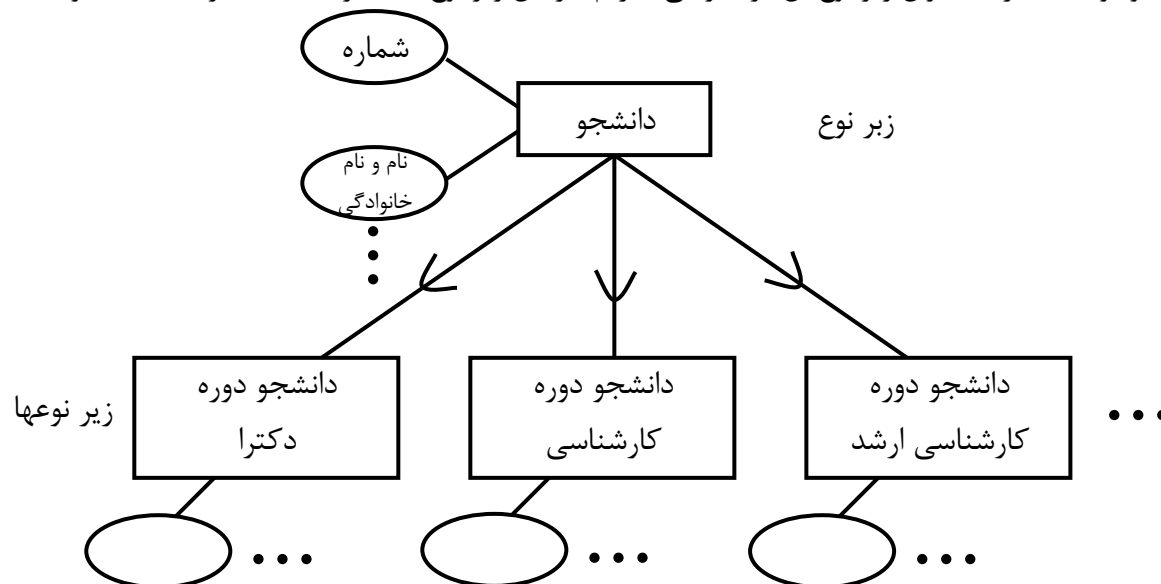
<sup>1</sup> contentment

<sup>2</sup> IS-A PART-OF

<sup>3</sup> Subentity type

<sup>4</sup> Superentity type

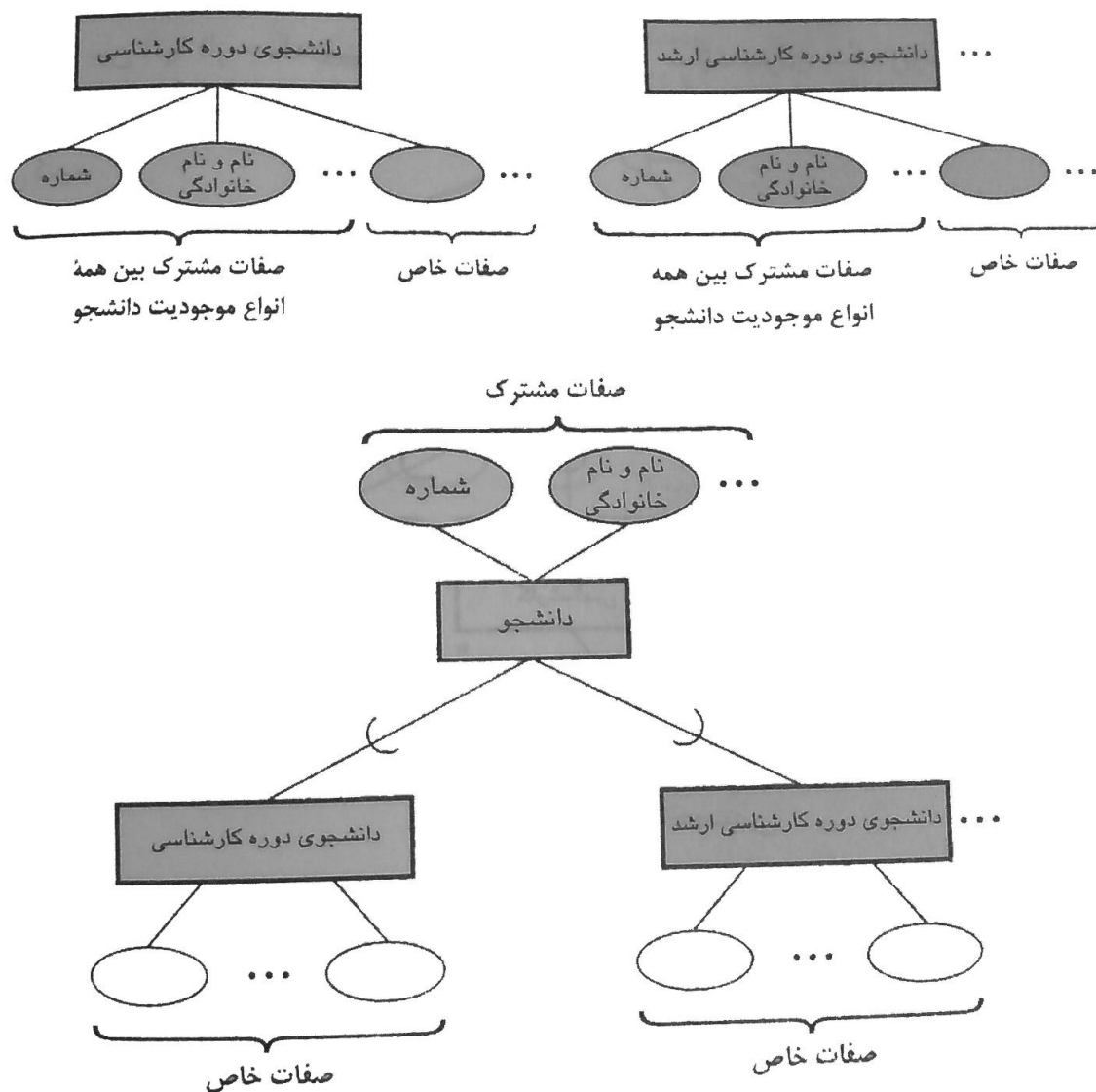
زیر نوع مجموعه ای از صفات دارد که بین تمام زیر نوع های آن مشترک است و بنابر این ، هر زیر نوع، صفات زیر نوع خود را به ارث می برد. به علاوه هر زیر نوع ، صفات خاص خود را دارد. تعمیم ، عکس عمل تخصیص است ، به این معنا که با داشتن زیر نوعهای خاص ، صفات مشترک بین آنها (از جمله شناسه) را در یک مجموعه صفات برای یک زیر نوع موجودیت در نظر می گیریم، طبعاً این مجموعه صفات مشترک جدا شده را ، دیگر برای هر زیر نوع تکرار نمی کنیم. از این خاصیت زمانی استفاده خواهیم کرد که در سیستم چندین موجودیت داشته باشیم که دارای صفات مشترکی باشند در این حالت یک زیر نوع ایجاد کرده و صفات مشترک را برای آن در نظر می گیریم و بقیه ی موجودیت ها را به عنوان زیر نوع آن در نظر می گیریم. در این زیر نوع ها دیگر صفات مشترک لحاظ نخواهد شد.



شکل ۱۲-۲: مثال تخصیص

IS-A<sup>1</sup>

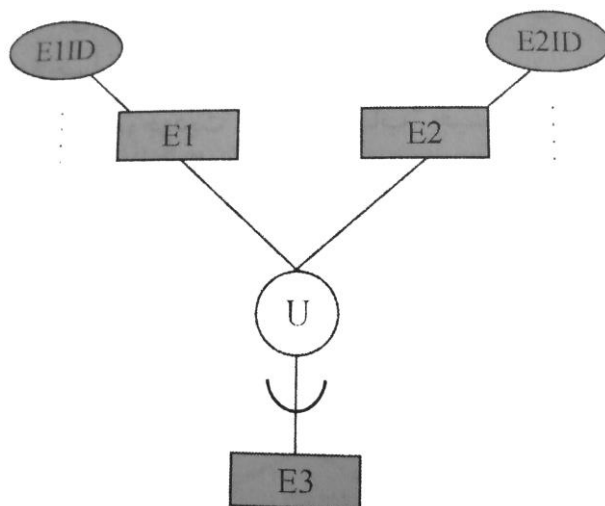




شکل ۱۳-۲: مثال تعمیم

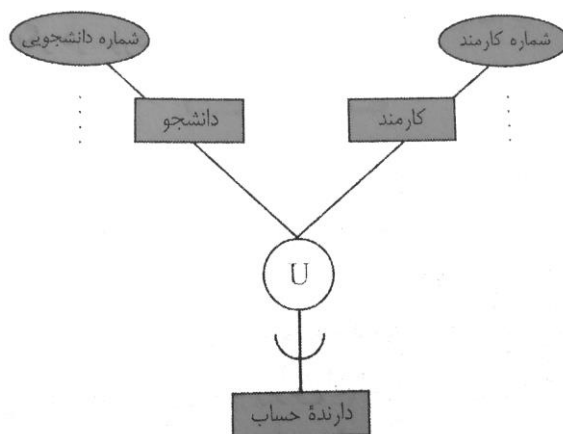
۱۴-۱-۲- دسته بندی

ممکن است زبر نوع ها یک زبر نوع از یک نوع نباشند. به این زبر نوع ها اصطلاحاً دسته (طبقه) گفته می شود و برای نمایش آن از نماد U استفاده می شود، به صورت شکل زیر نشان داده می شود.



شکل ۱۴-۲: دسته بندی

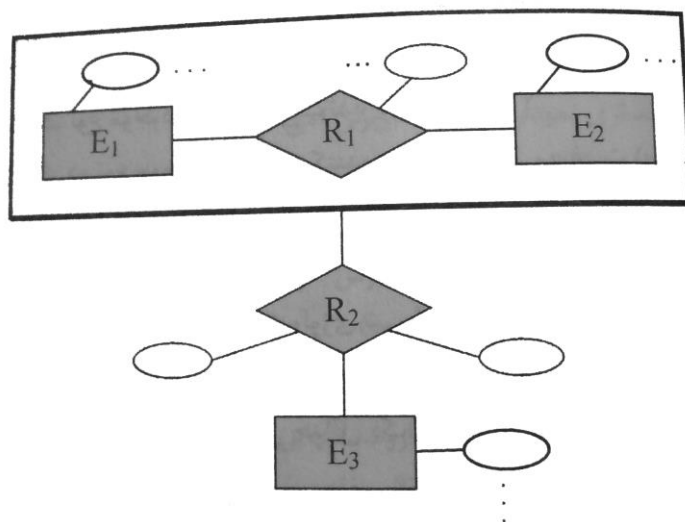
مثال زیر نمونه ای از این حالت را نشان می دهد. در این مثال دارنده ی حساب از اجتماع کارمند و دانشجو تشکیل شده است یعنی دارنده ی حساب یا کارمند است یا دانشجو.



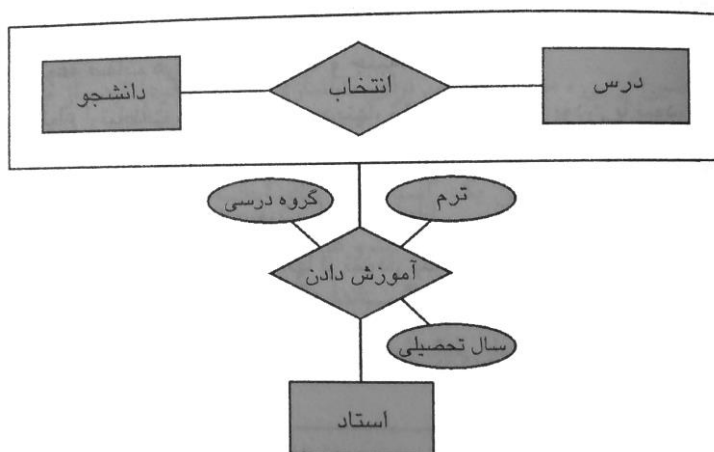
شکل ۱۵-۲: مثال دسته بندی

## ۱۵-۱-۲-تجمع

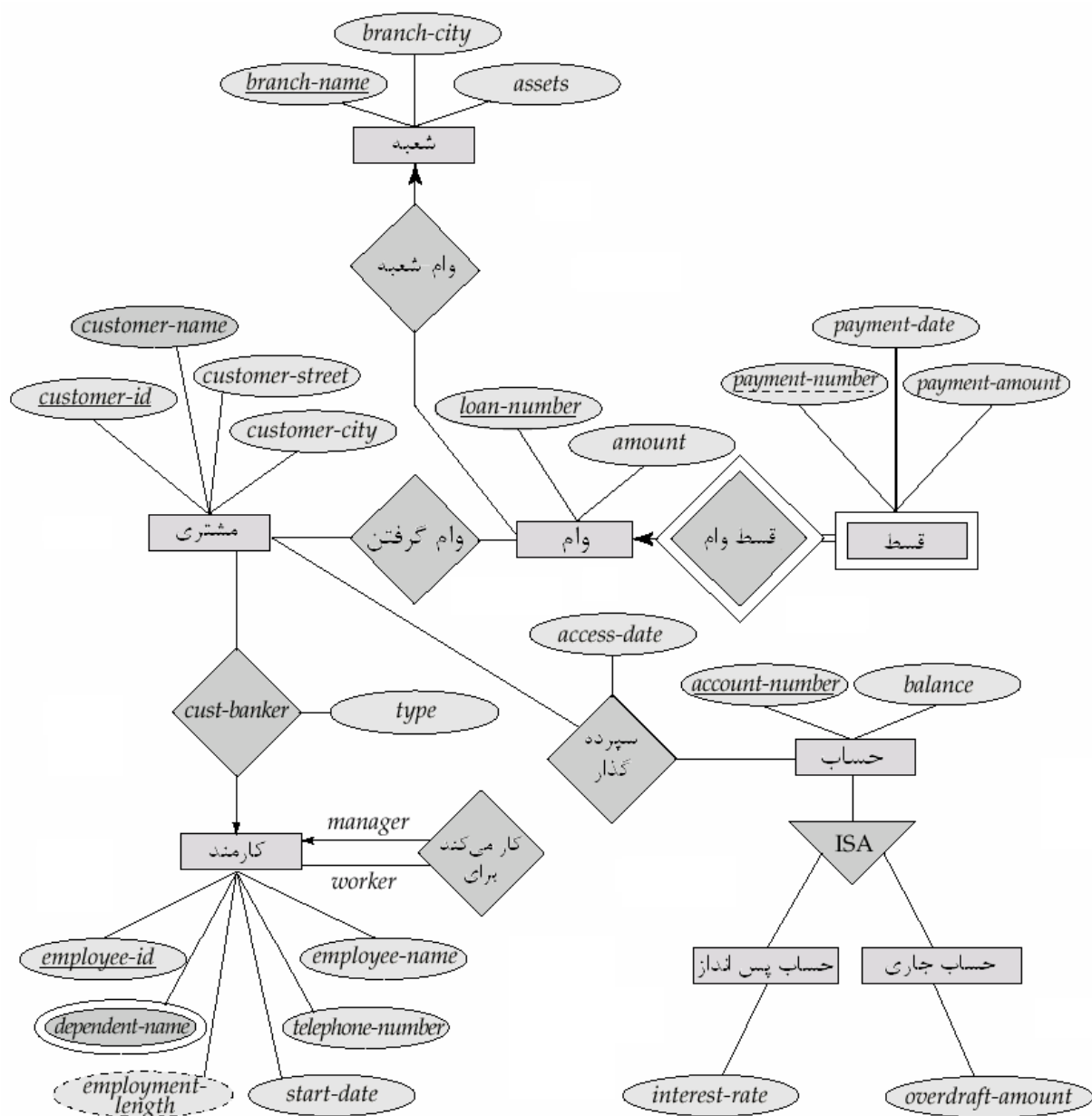
عبارتست از ساختار یک موجودیت جدید با ۲ یا بیش از ۲ موجودیت که با هم در یک ارتباط شرکت دارند، به صورت یک نوع موجودیت واحد. به عنوان مثال شکل زیر عمل تجمع را نشان می دهد.



شکل ۱۶-۲: نمایش تجمع

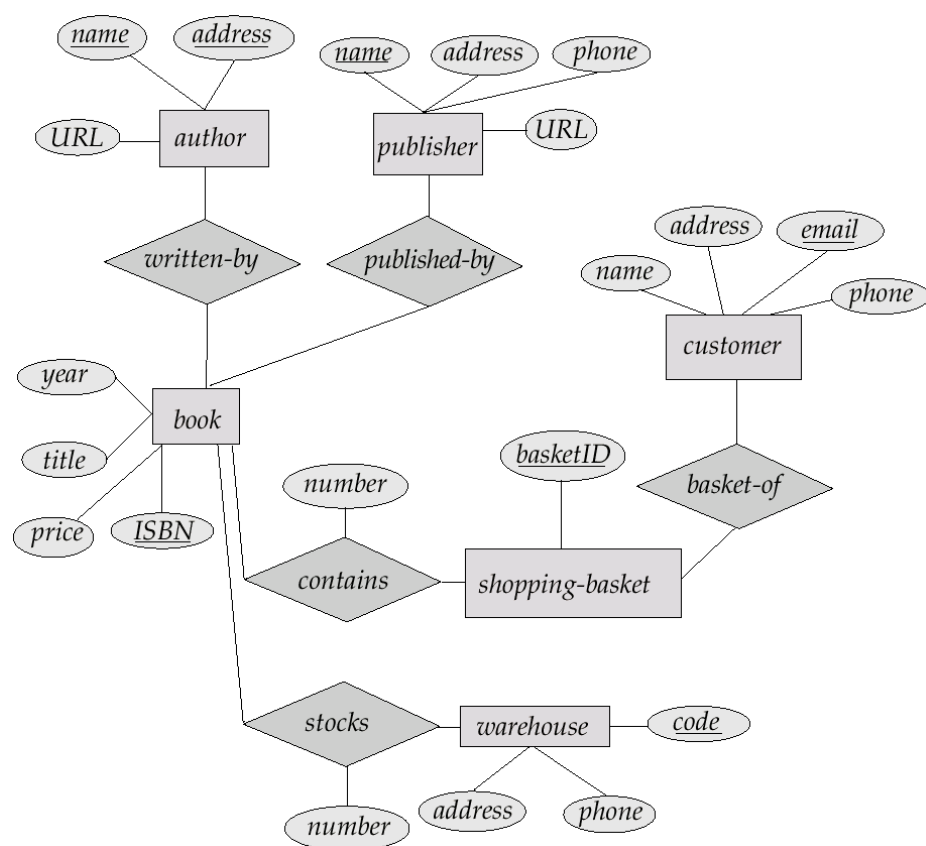


شکل ۱۷-۲: مثال تجمع



شکل ۱۸-۲: مثالی از نمودار ER یک موسسه بانکی (مثالی از کتاب silberschots)

توجه: برای فهم نمادهای استفاده شده در این نمودار به ضمیمه A مراجعه کنید.



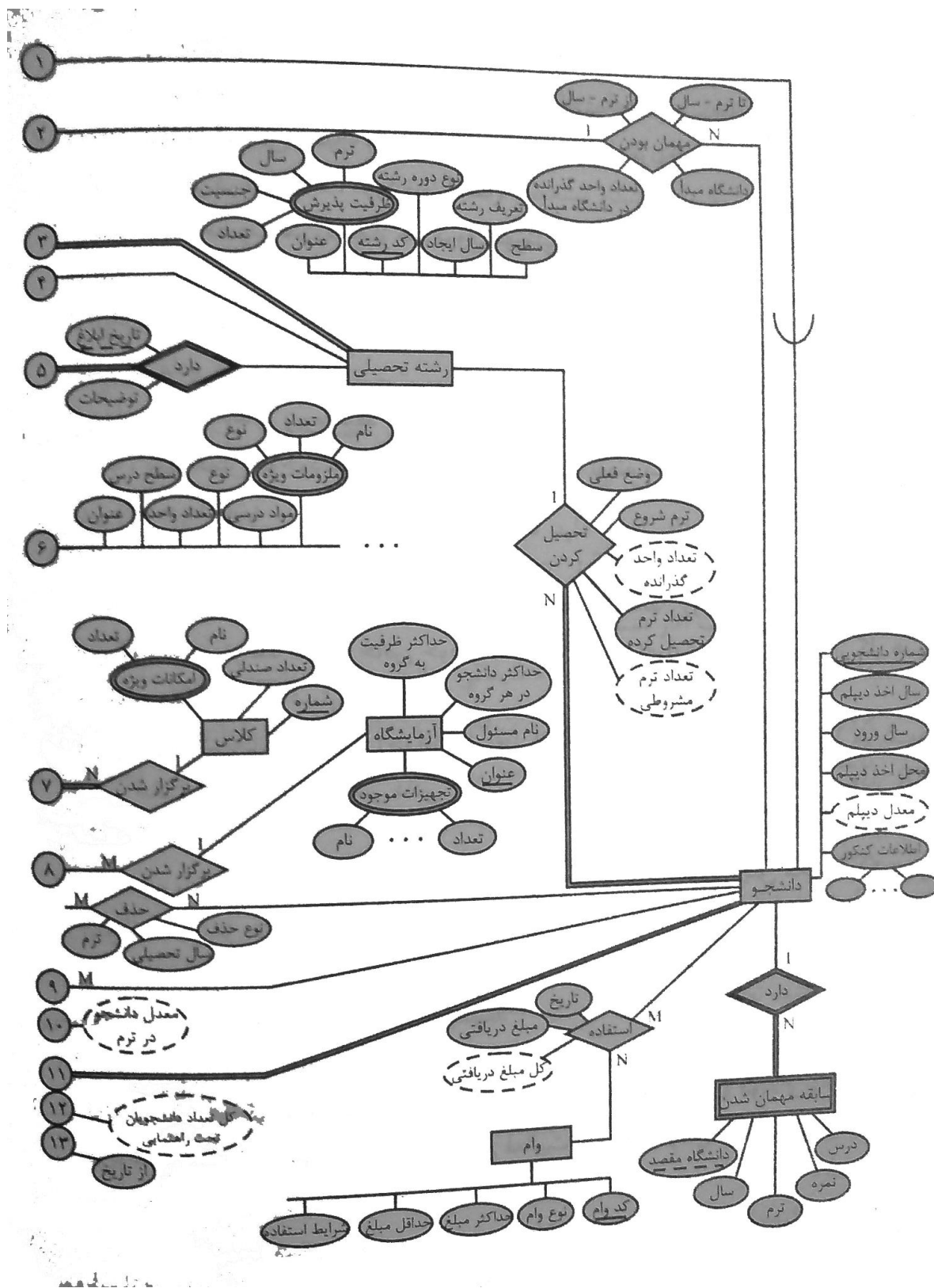
شکل ۱۹-۲: silberschots کتاب online مثالی از کتاب

تمرین: شکل ۱۹-۲ را در نظر بگیرید که یک کتابفروشی online را مدل می کند.

۱. در نظر بگیرید که کتابفروشی می خواهد کاست موسیقی و دیسک فشرده را به مجموعه خود اضافه کند. موسیقی های یکسان ممکن است در شکل کاستها یا دیسکهای فشرده، با قیمتهای متفاوت ارائه شوند. نمودار ER را برای مدل کردن این موارد اضافی تغییر دهید، از اثرات آن روی shopping-basket صرفنظر شود.

۲. حالا نمودار ER را برای عمومیت دادن به منظور مدل کردن حالتی که shopping-basket ممکن است شامل هر ترکیبی از کتابها، کاستهای موسیقی، یا دیسکهای فشرده باشد تغییر دهید.





شکل ۲۰-۲: بخشی از نمودار EER دانشکده (از کتاب مرجع جزوه)

## فصل سوم

## ۳- مفاهیم اساسی مدل رابطه‌ای

## ۳-۱- تعاریف

دامنه<sup>۱</sup>

مجموعه‌ای از مقادیر ممکن صفت (attribute) است. از نظر ریاضی، مجموعه‌ای است از مقادیر که یک یا بیش از یک صفت از آن مقدار می‌گیرند.

رابطه<sup>۲</sup>

رابطه عبارت است از زیرمجموعه‌ای از ضرب کارتیزین  $S_1 \times S_2 \times S_3 \times \dots \times S_n$ ، و می‌گوییم رابطه R از درجه n است. هر یک از  $S_1, S_2, S_3, \dots$  و ... میدان یا دامنه نامیده می‌شود.

تاپل<sup>۳</sup>

ارتباط مجموعه‌ای از مقادیر در یک رابطه را تاپل گویند.

تناظر بین مفاهیم رابطه‌ای و مفاهیم جدولی

در جدول زیر این تناظر نشان داده شده است.

مفهوم تئوریک	مفهوم جدولی
رابطه	جدول
تاپل	سطر
صفت	ستون
میدان	مجموعه مقادیر ستون
درجه	تعداد ستونها
کاردینالیته	تعداد سطرها

## ۳-۲- ویژگیهای رابطه

- ویژگی ۱: رابطه تاپل تکراری ندارد.  
دلیل: زیرا بدنه رابطه مجموعه ای است و مجموعه نمی تواند عضو تکراری داشته باشد.
- ویژگی ۲: تاپل نظم ندارند.  
دلیل: بدنه رابطه مجموعه است و مجموعه در حالت کلی، فاقد نظم است.
- ویژگی ۳: صفات رابطه نظم مکانی ندارند (از چپ به راست).  
دلیل: سرآیند رابطه مجموعه است و مجموعه در حالت کلی فاقد نظم است.
- ویژگی ۴: تمامی صفات، تک مقداری هستند.  
دلیل: این ویژگی دلیل نظری ندارد.

<sup>1</sup> Domain

<sup>2</sup> Relation

<sup>3</sup> Tuple



**نکته:** در مدل رابطه‌ای هم نوع موجودیت و هم نوع ارتباط با مفهوم رابطه نمایش داده می شوند و در نتیجه هم نمونه موجودیت و هم نمونه ارتباط ، با مفهوم تاپل نشان داده می شوند.

### ۳-۳- کلید در مدل رابطه‌ای

چند مفهوم در بحث کلید وجود دارد که عبارتند از:

- ابر کلید<sup>۱</sup>
- کلید کاندید<sup>۲</sup>
- کلید اصلی<sup>۳</sup>
- کلید دیگر<sup>۴</sup>
- کلید خارجی<sup>۵</sup>

#### ۱-۳-۳- ابر کلید

هر زیر مجموعه از مجموعه عنوان رابطه که یکتایی مقدار در گستره (بدنه) رابطه داشته باشد. به بیان دیگر، هر ترکیبی از اسامی صفات رابطه که در هیچ دو تاپل مقدار یکسان نداشته باشد.

#### ۲-۳-۳- کلید کاندید

هر زیر مجموعه از مجموعه عنوان رابطه که دو خاصیت زیر را داشته باشد، کلید کاندید رابطه است:

۱. یکتایی مقدار
۲. کاهش ناپذیری. یعنی اگر یکی از عناصر از مجموعه را حذف کنیم زیر مجموعه باقی مانده دیگر خاصیت یکتایی مقدار نداشته باشد.

**نکته:** هر ابر کلید لزوماً کلید کاندید نیست، اما هر کلید کاندید زیر مجموعه ابر کلید های رابطه است.

#### ۳-۳-۳- کلید اصلی

یکی از کلیدهای کاندید رابطه که طراح انتخاب می کند و به سیستم معرفی می شود.

ظابطه های انتخاب عبارتند از:

- ۱- از نظر کاربر، شناسه معمول نوع موجودیت باشد.
- ۲- طول کوتاهتر داشته باشد.

#### ۴-۳-۳- کلید دیگر

هر کلید کاندید، غیر از کلید اصلی، کلید دیگر (بدیل) نام دارد.

<sup>1</sup> Super Key (S.K.)

<sup>2</sup> Candidate Key (C.K.)

<sup>3</sup> Primary Key (P.K.)

<sup>4</sup> Alternate Key (A.K.)

<sup>5</sup> Foreign Key (F.K.)

### ۵-۳-۳- کلید خارجی

دو رابطه  $R1$  و  $R2$  (نه لزوماً متمایز) را در نظر می گیریم. هر زیر مجموعه از صفات رابطه  $R2$  که هر مقدار معلومش با یک مقدار از کلید کاندید  $R1$  برابر باشد، کلید خارجی در رابطه  $R2$  است. صفت (صفات) کلید خارجی باید هم میدان با صفت (صفات) کلید کاندید باشد، و معمولاً همنام با کلید کاندید است ولی گاه لازم می شود از مکانیزم دگرنامی<sup>۱</sup> استفاده کنیم.

### ۴-۳- قواعد جامعیت در مدل رابطه‌ای

جامعیت<sup>۲</sup> پایگاه داده ها یعنی : صحت، دقت و سازگاری داده‌های ذخیره شده در پایگاه در تمام لحظات. یعنی صحت داده ها و پردازشها و پی روی از مقررات سیستم.

در مدل رابطه ای سه نوع جامعیت مورد تاکید قرار می گیرد

۱- جامعیت دامنه<sup>۳</sup>

۲- جامعیت درون رابطه‌ای<sup>۴</sup>

۳- جامعیت ارجاع<sup>۵</sup>

#### ۱-۴-۳- جامعیت دامنه

تمامی صفات در تمامی رابطه ها از نوع دامنه خود باشند.

#### ۲-۴-۳- جامعیت درون رابطه‌ای

یعنی هر رابطه به تنهایی صحیح باشد. مثلاً عضو تکراری نداشته باشد و کلیدهایش درست باشند و کلیدها دارای مقدار تهی (Null) یا تکراری نباشند.

#### ۳-۴-۳- جامعیت ارجاع

یعنی کلید خارجی درست تعریف شده باشد. مثلاً کلید خارجی یک رابطه حتماً در رابطه دیگر کلید باشد و مقداری که به کلید خارجی داده می شود در جدول دیگر وجود داشته باشد.

<sup>1</sup> Renaming

<sup>2</sup> Integrity

<sup>3</sup> Domain Integrity

<sup>4</sup> Intra-relation Integrity

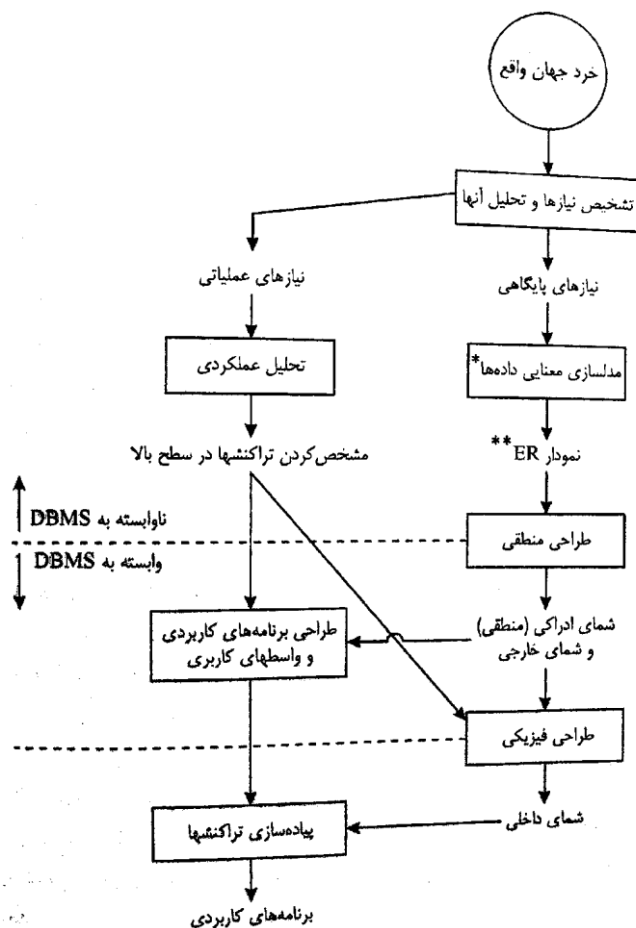
<sup>5</sup> Referential Integrity

## فصل چهارم

## ۴- طراحی پایگاه داده های رابطه ای

طراحی پایگاه داده مراحل دارد و در هر مرحله فعالیتهایی انجام می شود. مراحل اساسی طراحی پایگاه داده ها عبارتند از:

- ۱- مطالعه و شناخت خرد جهان واقع
- ۲- انجام مهندسی نیازها
- ۳- مدلسازی معنایی داده ها
- ۴- طراحی منطقی پایگاه داده ها
- ۵- طراحی فیزیکی پایگاه داده ها
- ۶- انجام تحلیل عملکرد : تعیین تراکنشها طراحی برنامه های کاربردی



شکل ۱-۳ : مراحل اساسی طراحی پایگاه داده ها

شکل ۴-۱ : مراحل اساسی طراحی پایگاه داده ها

برای طراحی منطقی پایگاه داده ها دو روش وجود دارد:

- ۱- روش بالا به پایین<sup>۱</sup>

<sup>۱</sup> top-down design method

۲- روش سنتز رابطه ای<sup>۱</sup>

۳- روش ترکیبی

## ۱-۴- تبدیل مدلسازی به طراحی منطقی (روش بالا به پایین)

فرض ما بر این است که مراحل قبل انجام شده است یعنی:

- شناخت خرد جهان واقع و انجام مهندسی نیازها
- مدلسازی معنایی داده ها

نکته: کار طراحی ماهیت هنری هم دارد و طراحی نوعی هنر است. در این کار ای بسا ظرافت و دقایق فنی وجود دارد.

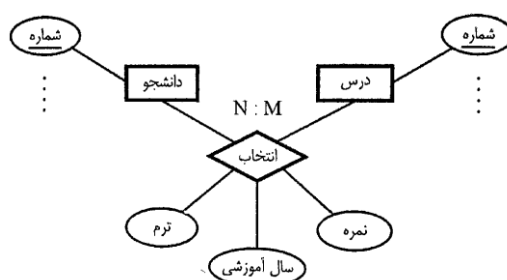
## ۱-۱-۴- روش تبدیل نمودار ER به رابطه ها

## ۱-۱-۴-۱- حالت اول

- تعداد نوع موجودیت :  $n \geq 2$
- وضع موجودیت ها : مستقل
- چندی ارتباط :  $N : M$

در این حالت  $n+1$  ارتباط نیاز است. یک رابطه برای هر یک از  $n$  موجودیت مستقل و یک رابطه برای نمایش ارتباط بین آنها. اگر ارتباط بین این انواع موجودیتها، صفت (ساده یا مرکب) چند مقداری نداشته باشد در اینصورت کلید کاندید رابطه نمایشگر ارتباط بین  $n$  موجودیت، از ترکیب کلیدهای کاندید  $n$  رابطه نمایشگر  $n$  موجودیت بدست می آید. در غیر اینصورت، حد اقل یکی از صفات ارتباط هم جزء تشکیل دهنده کلید کاندید (که مرکب است) ، می شود. پس در این حالت کلیدهای خارجی، اجزاء تشکیل دهنده کلید کاندید رابطه نمایشگر ارتباط هستند.

مثال حالت اول:



در این حالت همان رابطه های STT ، COT ، و STCOT را داریم که در گفتار دهم دیدیم.

STT( STID , ..... )  
C.K.

COT( COID , ..... )  
C.K.

STCOT( STID , COID , ..... )  
F.K. F.K.  
C.K.

<sup>1</sup> Relational synthesis

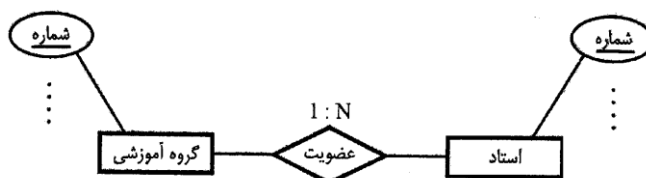
توجه داشته باشید که در این طراحی وضعیتی که در آن دانشجویی در یک درسی مردود می شود و دوباره همان درس را در ترم دیگر انتخاب می کند در نظر گرفته نشده است. اگر این وضعیت را بخواهیم در نظر بگیریم، دیگر کلید کاندید رابطه STCOT فقط ترکیب STID و COID نیست بلکه باید TR و حتی YRYR را نیز دخالت داد، یعنی کلید کاندید رابطه ، (STID , COID , TR , YRYR) می شود.

## ۲-۱-۱-۴- حالت دوم

- تعداد نوع موجودیت : دو
- وضع موجودیت ها : مستقل
- چندی ارتباط : 1 : N

در این حالت دو رابطه کفایت می کند. یک رابطه برای نمایش نوع موجودیت طرف یک و یک رابطه برای نمایش نوع موجودیت طرف N و نیز ارتباط 1:N. در رابطه اخیر ، کلید کاندید رابطه اول ، به عنوان کلید خارجی رابطه دوم ، ارتباط مورد نظر را نشان می دهد (اصطلاحاً می گوئیم طرف 1 به طرف N کلید می دهد) و کلید خارجی جزء تشکیل دهنده کلید کاندید رابطه دوم نیست. بصورت خلاصه کلید اصلی رابطه اول به عنوان کلید خارجی به این رابطه اضافه می شود.

## مثال حالت دوم



DEPT( DEID , DTITLE , ..... )  
C.K.

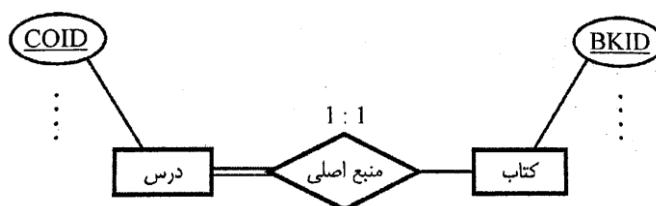
PROF( PRID , PRNAME , ..... , DEID )  
C.K. F.K.

## ۳-۱-۱-۴- حالت سوم

- تعداد نوع موجودیت : دو
- وضع موجودیت ها : مستقل
- چندی ارتباط : 1 : 1

در این حالت دو رابطه لازم است. یک رابطه برای نمایش یکی از دو نوع موجودیت و یک رابطه برای نمایش موجودیت دیگر و ارتباط بین دو نوع موجودیت.

## مثال حالت سوم



**BOOK**(  $\frac{BKID}{C.K.}$  , BKTITLE , .....)

**COT**(  $\frac{COID}{C.K.}$  , COTITLE , ..... ,  $\frac{BKID}{F.K.}$  )

توجه داشته باشید که در این حالت، کلید کاندید رابطه نشان دهنده نوع موجودیت با مشارکت غیر الزامی در ارتباط، در رابطه نشان دهنده نوع موجودیت با مشارکت الزامی در ارتباط، به عنوان کلید خارجی، در نظر گرفته می شود.

**نکته:** می توان با یک رابطه نیز نشان داد: وقتی مشارکت هر دو نوع موجودیت در ارتباط الزامی باشد و تعداد صفات موجودیت های شرکت کننده در ارتباط زیاد نباشد.

**COBK**(  $\frac{COID}{C.K.}$  , COTITLE , ... ,  $\frac{BKID}{C.K.}$  , BKTITLE , .....)

توجه داشته باشید که کلید کاندید این رابطه هر یک از دو صفت COID و BKID است و هر یک از این دو می توانند به عنوان کلید اصلی انتخاب شوند.

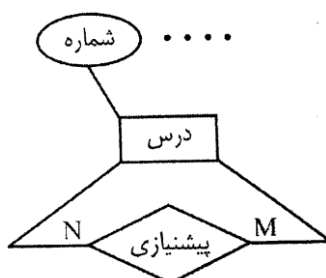
**نکته:** همیشه می توان در حالت دوم و سوم، مثل حالت اول عمل کرد، یعنی به جای دو یا یک رابطه، همان سه رابطه را طراحی کرد. این کار بویژه وقتی مشارکت در ارتباط غیر الزامی باشد و تعداد نمونه های نوع موجودیت های شرکت کننده در ارتباط کم باشد، توصیه می شود.

#### ۴-۱-۱-۴- حالت چهارم

- تعداد نوع موجودیت : یک
- وضع موجودیت ها : مستقل
- چندی ارتباط : N : M

این حالت خاص حالت اول است. دو رابطه لازم است: یک رابطه برای نمایش خود موجودیت و یک رابطه برای نمایش ارتباط، اعم از اینکه الزامی باشد یا نباشد.

#### مثال حالت چهارم



**COT**(  $\frac{COID}{C.K.}$  , TITLE , .....)

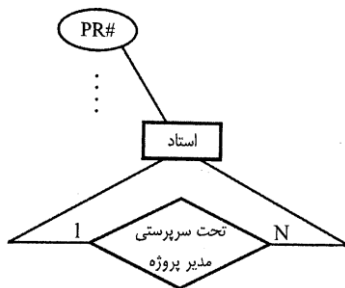
**PREREQ**(  $\frac{COID}{F.K.}$  ,  $\frac{PRCOID}{F.K.}$  )  
C.K.

۵-۱-۴- حالت پنجم

- تعداد نوع موجودیت : یک
- وضع موجودیت ها : مستقل
- چندی ارتباط : 1 : N

این حالت که حالت خاص حالت دوم است، یک رابطه کفایت می کند.

مثال حالت پنجم



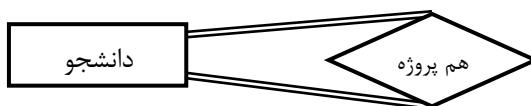
**PRJMGR**( PRID , ..... , PRMGRID )  
 C.K. F.K.

۶-۱-۴- حالت ششم

- تعداد نوع موجودیت : یک
- وضع موجودیت ها : مستقل
- چندی ارتباط : 1 : 1

این حالت که حالت خاص حالت سوم است، یک رابطه کفایت می کند، به شرط اینکه مشارکت در ارتباط الزامی باشد. البته می توان با دو رابطه هم طراحی کرد.

مثال حالت ششم



طراحی با یک رابطه :

**STUD**( STID , STNAME , ..... , PJSTID )  
 C.K. F.K. ↑ شماره دانشجویی هم تیم

و یا :

**STPJST**( STID , STNAME , ..... , PJSTID , SSTNAME , ..... )  
 C.K. F.K.

طراحی با دو رابطه:

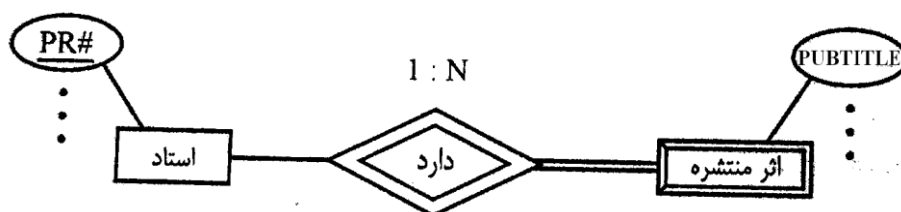
**STUD**( STID , STNAME , ..... )  
 C.K.

**STOJCOST**( STID , PJSTID )  
 F.K. F.K.  
 C.K.

۷-۱-۴- حالت هفتم: نمایش موجودیت ضعیف

موجودیت ضعیف دارای شناسه یکتا نیست بلکه صفت ممیزه دارد. برای نمایش این نوع موجودیت یک رابطه طراحی می کنیم که در عنوان آن، صفات موجودیت ضعیف و کلید کاندید موجودیت قوی که با آن ارتباط دارد وجود دارد. یعنی کلید کاندید موجودیت قوی به عنوان کلید خارجی اضافه می شود. کلید کاندید این رابطه از ترکیب کلید کاندید موجودیت قوی و صفت ممیزه بدست می آید.

مثال حالت هفتم



$$\text{PRPUB}(\underline{\text{PRID}}, \text{PUBTITLE}, \dots)$$

$\frac{\text{F.K.}}{\text{C.K.}}$

۸-۱-۴- حالت هشتم: وجود صفت چند مقداری

اگر  $MVA$ ، یک صفت چند مقداری،  $EID$  شناسه نوع موجودیت  $E$  و  $A_1, A_2, \dots, A_i$  سایر صفات تک مقداری (ساده یا مرکب) موجودیت  $E$  باشند، در اینصورت برای نمایش این موجودیت دو رابطه نیاز است.

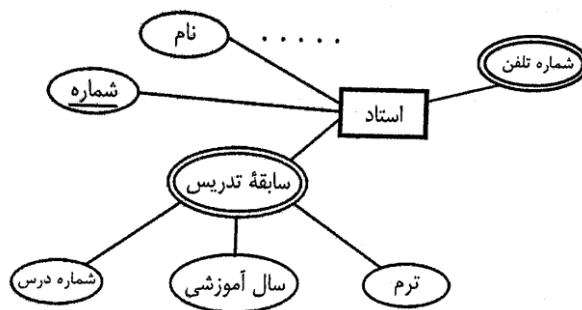
$$R_1(\underline{EID}, A_1, A_2, \dots, A_i)$$

C.K.

$$R_2(\underline{EID}, MVA)$$

$\frac{\text{F.K.}}{\text{C.K.}}$

مثال حالت هشتم



$$\text{PROF}(\underline{\text{PRID}}, \text{PRNAME}, \dots)$$

C.K.

$$\text{PRTEACHIS}(\underline{\text{PRID}}, \text{COID}, \text{TR}, \text{YR})$$

$\frac{\text{F.K.}}{\text{C.K.}}$

$$\text{PRTEACHIS}(\underline{\text{PRID}}, \text{TEL})$$

$\frac{\text{F.K.}}{\text{C.K.}}$

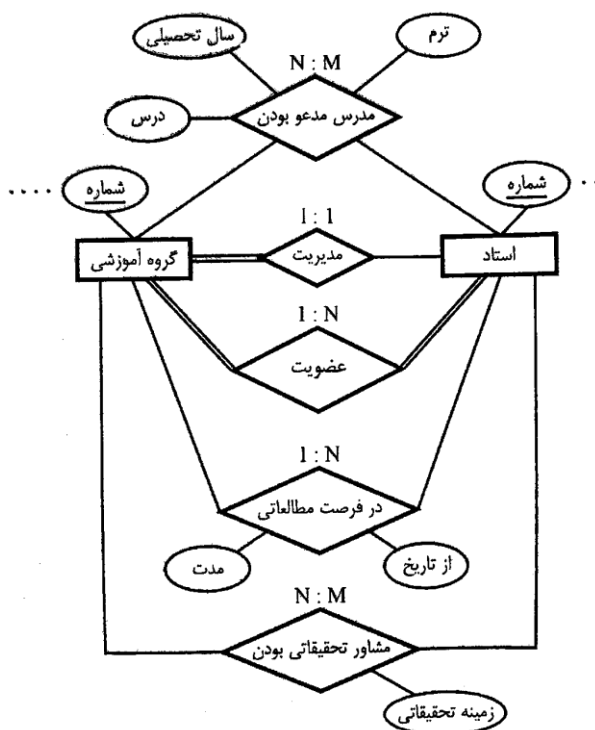


۹-۱-۱-۴- حالت نهم: بیش از یک ارتباط بین دو موجودیت

فرض می کنیم که هر دو نوع موجودیت مستقل باشند. تعداد رابطه ها لازم در این حالت بستگی به چندی هر ارتباط دارد. در طراحی این حالت، بصورت زیر عمل می شود:

- هر نوع موجودیت مستقل شرکت کننده در یک ارتباط با چندی  $N:M$  با یک رابطه نشان داده می شود.
- هر نوع ارتباط  $N:M$  را با یک رابطه نشان می دهیم.
- هر یک از ارتباطهای با چندی  $1:N$ ، اگر مشارکت دو نوع موجودیت در همه ارتباطها الزامی باشد، را می توان با یک کلید خارجی در رابطه نشان دهنده نوع موجودیت طرف ( $N$ )، نشان داد.
- اگر مشارکت دو موجودیت در یک ارتباط  $1:N$ ، الزامی نباشد بهتر است برای چنین ارتباطی یک رابطه جداگانه طراحی شود که صفات آن، شناسه دو نوع موجودیت و صفات خود ارتباط، در صورت وجود، هستند.
- اگر یک و یا بیش از یک ارتباط  $1:N$  وجود داشته باشد و مشارکت دو نوع موجودیت در ارتباطها الزامی باشد، هر یک از این ارتباطها را می توان با یک کلید خارجی در یکی از دو رابطه نشان دهنده یکی از دو نوع موجودیت، نشان داد، مگر اینکه ملاحظات خاص دیگر مطرح باشد.
- اگر مشارکت هر دو نوع موجودیت در یک ارتباط  $1:1$  الزامی نباشد، بهتر است این ارتباط را با یک رابطه چندگانه نمایش داد.

مثال حالت نهم



برای ارتباط "مدیریت"  $\swarrow$   
**DEPT**( DEID , DETITLE , ..... , PRID )  
 C.K. F.K.

برای ارتباط "عضویت"  $\swarrow$   
**PROF**( PRID , PRNAME , ..... , DEID )  
 C.K. F.K.

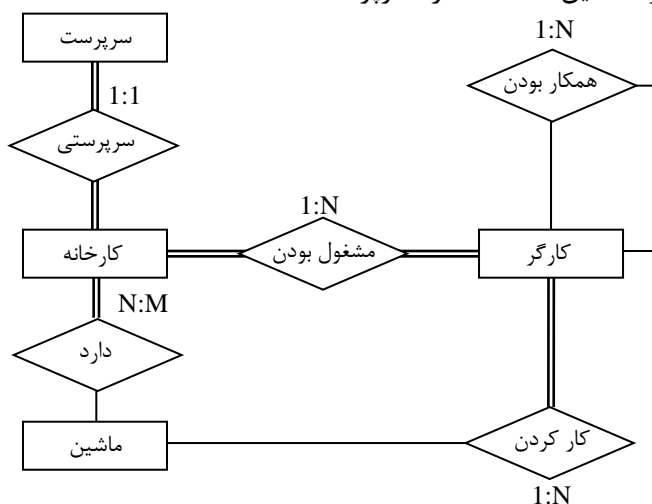
برای ارتباط "فرصت مطالعاتی"  $\swarrow$   
**PROF-IN-SAB**( PRID , BEGDATE , DURATION , DEID )  
 C.K. F.K.

**PROF-CONSULT**( PRID , DEID , RESEARCH-AREA )  
 C.K.

**PROF-INVIT**( PRID , DEID , TR , YR , COID )  
 C.K.

**تمرینات**

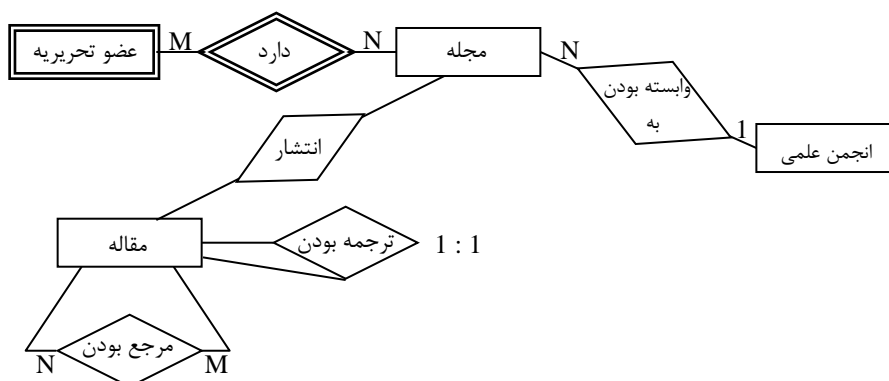
تمرین ۱: نمودار ER زیر را در نظر می گیریم: کلید اصلی موجودیتهای عبارتند از: F#: شماره کارخانه، W#: شماره کارگر، M#: شماره ماشین، A#: شماره سرپرست.



پایگاه رابطه ای این محیط را طراحی کنید.

توجه: صفت (صفات) هر موجودیت را متناسباً انتخاب کنید.

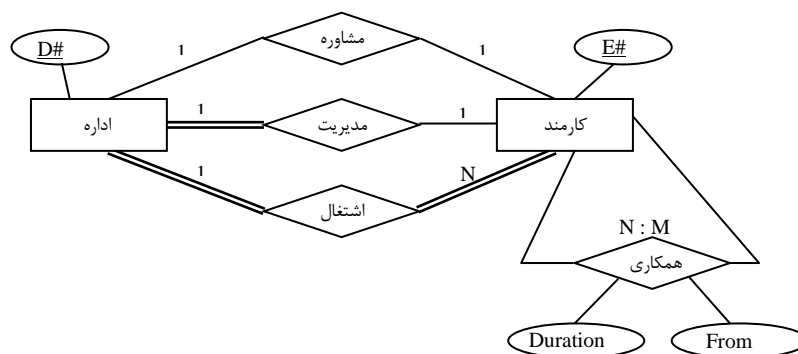
تمرین ۲: با استفاده از روش طراحی بالا به پایین نمودار EER زیر را با مدل رابطه ای تبدیل کنید؟



کلید اصلی موجودیتها عبارتند از: مجله : عنوان، انجمن علمی: عنوان، مقاله: عنوان و کد مقاله، کلید صفت ممیزه عضو تحریریه: نام

(توجه: صفت (صفات) هر موجودیت و هر ارتباط را متناسباً انتخاب کنید.)

تمرین ۳: با استفاده از روش طراحی بالا به پایین نمودار EER زیر را با مدل رابطه ای تبدیل کنید؟



## فصل پنجم

## ۵- جبر رابطه‌ای

## ۵-۱- جداول نمونه

توجه: مطالب این فصل از کتاب "بانکهای اطلاعاتی علمی کاربردی" تألیف دکتر مصطفی حق جو می باشد. پایگاه داده نمونه‌ای که مثالهای این فصل بر اساس آن مطرح شده، پایگاه داده دانشگاه است که، بصورت شکل زیر می باشد.

Stud (s#, sname, city, avg, clg#)

(شماره دانشکده، معدل کل، شهر محل تولد، نام، شماره) دانشجو

Prof (pname, office, esp, degree, clg#)

(شماره دانشکده، مدرک تحصیلی، تخصص، دفتر کار، نام) استاد

Crs (c#, cname, unit, clg#)

(شماره دانشکده ارائه دهنده، تعداد واحد، نام، شماره) درس

Sec (sec#, c#, s#, term, pname, score)

(نمره، نام استاد، ترم، شماره دانشجو، شماره درس، شماره گروه) گروه درس

Clg (clg#, clgname, city, pname)

(نام رئیس، نام شهر، نام دانشکده، شماره) دانشکده

نمونه جداول بانک اطلاعات دانشگاه در زیر نمایش داده شده اند:

Stud				
S#	Sname	city	avg	Clg#
71133848	محمدی	تهران	17.25	10
72130502	وکیلی	اصفهان	14.06	10
72203305	علینقی زاده	مشهد	16.42	1
73120504	کماني	پزد	17.56	4
73166801	احمدی	کرمان	15.44	5
74182532	جوادی	تهران	16.8	5
74209836	حسین زاده	تبریز	17.7	6

Crs			
C#	Cname	Unit	Clg#
10172	شبیه سازی	3	10
10174	مدار منطقی	3	10
12100	معارف ۱	2	12
12564	ریاضی عمومی ۱	4	1
51515	شیمی عالی	3	5
71203	کنترل خطی	3	7

Prof				
Pname	office	esp	degree	Clg#
ابوطالبی	3	مواد	دکتری	6
اشرفی زاده	8	شیمی	دکتری	5
جاهد مطلق	1	کامپیوتر	دکتری	10
جلالی	5	برق	دکتری	7
حسنی	2	ریاضی	دکتری	1
ذاکر	4	فیزیک	دکتری	2
صادقیان	3	صنایع	دکتری	4
قربانی	12	کامپیوتر	دکتری	10
مفتون	1	زبان	دکتری	3
میرشمسی	4	کامپیوتر	فوق لیسانس	10
نقره کار	3	معماری	دکتری	11
هاشمی اصل	10	کامپیوتر	فوق لیسانس	10

Sec					
Sec#	S#	C#	Term	Score	Pname
1724	71133848	10172	761	14.5	هاشمی اصل
1747	71133848	10174	752	15.75	میرشمسی
1747	72130502	10174	752	12.5	میرشمسی
1748	72203305	10172	761	16.25	قربانی
1516	74182532	51516	752	17	اشرفی زاده

Clg			
Clg#	Clgname	City	Pname
1	ریاضی	تهران	حسنی
2	فیزیک	مشهد	جاهد مطلق
3	زبان	مشهد	نقره کار
4	صنایع	تهران	صادقیان
5	شیمی	تهران	ذاکر
6	مواد	تبریز	مفتون
7	برق	تهران	صادقیان
10	کامپیوتر	تهران	اشرفی زاده
11	معماری	یزد	ابوطالبی
12	معارف	تهران	جلالی

در جبر رابطه ای ما با دو بخش روبرو خواهیم بود:

- عملوند یا نوع داده‌ای

## • عملگرها

## ۲-۵- عملوند یا نوع داده ای

در جبر رابطه ای همیشه ورودی عملگرها و نتایج خروجی آنها یک رابطه خواهد بود پس همیشه در جبر رابطه ای با رابطه (جدول) سر و کار خواهیم داشت.

## ۳-۵- عملگرها

در جبر رابطه ای عملگرها به ۴ دسته تقسیم می شوند:

۱. دسته ی اول: عملگرهای ساده مانند:  $\Pi(\text{project})$  ,  $\sigma(\text{select})$
۲. دسته ی دوم: عملگرهای مجموعه ای:  $U$  ,  $\cap$  و  $-$
۳. دسته ی سوم: عملگرهای پیوند:  $X_{\theta}$  ,  $X$  و  $\infty$
۴. دسته ی چهارم: شامل سایر عملگرها مانند:  $\div$  ,  $\leftarrow$  ,  $P$

ما این عملگرها را به دو دسته تقسیم خواهیم کرد:

عملگرهای اصلی

عملگرهای اضافی

عملگرهای اضافی عملگرهایی می باشند که بتوان نتیجه آنها را با عملگرهای اصلی بدست آورد. بعنوان مثال عملگر اشتراک اضافی و عملگر تفاضل اصلی می باشند چون داریم:

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

۱-۳-۵- دسته ی اول عملگرهای  $\Pi(\text{project})$  و  $\sigma(\text{select})$ 

دستور select و دستور project فقط شامل یک عملوند می باشند که از نوع رابطه می باشد ، عملگر select بر روی سطرهای جدول اعمال می شود و می تواند از روی سطرها ، سطرهایی را که شرط مشخص شده را داشته باشند انتخاب کند.

مثال مشخصات دانشجویان یزدی را پیدا کنید.

$$\sigma_{\text{City}="یزد"}(\text{Stud})$$

خروجی این دستور بصورت جدول زیر خواهد بود.

S#	Sname	city	avg	Clg#
73120504	کمانی	یزد	17.56	4

مثال: مشخصات دانشجویان یزدی دانشکده ی شماره ۴ را پیدا کنید.

$$\sigma_{\text{City}="یزد" \wedge \text{Clg\#}=4}(\text{Stud})$$

عملگر project بر روی ستونهای جدول اعمال می شود و می تواند ستون یا ستونهای مشخص شده را از جدول ورودی جدا کند.

مثال: ستونهای شماره ی دانشجویی ، نام دانشجو و کد دانشکده را از جدول دانشجو انتخاب کنید.

$$\Pi_{S\#,Sname,Clg\#}(Stud)$$

خروجی:

S#	Sname	Clg#
71133848	محمدی	10
72130502	وکیلی	10
72203305	علینقی زاده	1
73120504	کمائی	4
73166801	احمدی	5
74182532	جوادی	5
74209836	حسین زاده	6

مثال: شهرهای محل تولد دانشجویان

$$\Pi_{City}(Stud)$$

نکته: باید دقت داشته باشید که استفاده از عملگر project باعث حذف تکرار خواهد شد به عنوان مثال در مثال قبلی ممکن است چندین دانشجو از یک شهر وجود داشته باشد اما این عملگر تکرارها را حذف کرده و هر شهر فقط یک بار در خروجی آورده خواهد شد.

خروجی:

city
تهران
اصفهان
مشهد
پزد
کرمان
تبریز

مثال: ستون های شماره دانشجویی ، نام ، کد دانشکده و معدل دانشجویانی که معدل آنها بالای ۱۵ است. در این مثال ما مجبوریم که هم از عملگر select و هم از عملگر project استفاده کنیم اما کدامیک از اینها زودتر انجام شود اهمیت دارد.

۱ جواب  $\Pi_{S\#,Sname,Clg\#,Avg}(\sigma_{Avg>15}(Stud))$

۲ جواب  $\sigma_{Avg>15}(\Pi_{S\#,Sname,Clg\#,Avg}(Stud))$

خروجی:

S#	Sname	avg	Clg#
71133848	محمدی	17.25	10
72203305	علینقی زاده	16.42	1
73120504	کمائی	17.56	4
73166801	احمدی	15.44	5
74182532	جوادی	16.8	5

74209836	حسین زاده	1707	6
----------	-----------	------	---

در این مثال هر دو حالت صحیح می باشد اما اگر صورت مساله را به صورت زیر تغییر دهیم:

**مثال:** ستون های شماره ی دانشجویی ، نام و کد دانشکده ی دانشجویانی که معدل آنها بالای ۱۵ می باشد.

$$۱ \Pi_{S\#,Sname,C1g\#}(\sigma_{Avg>15}(Stud))$$

در این جواب ورودی جریان select جدول stud می باشد که یکی از ستونهای آن avg است و دستور select می تواند شرط  $avg > 15$  را بر روی آن اعمال کند و ورودی دستور project جدول حاصل از نتیجه ی عمل select می باشد. که این نتیجه شامل کل ستون های جدول stud می باشد سپس project ستونهای s# و clg# و sname را از آن اجرا می کند. اما جوابی که در زیر نشان داده می شود صحیح نیست.

$$۲ \sigma_{Avg>15}(\Pi_{S\#,Sname,C1g\#}(Stud))$$

در این جواب ورودی select خروجی حاصل از دستور project می باشد که جدولی است شامل ستونهای s# و sname و clg# ، اما شرطی که برای select شده  $avg > 15$  میباشد در حالی که جدول ورودی آن شامل این ستون نیست پس نمیتوان شرط را اعمال کرد.

### ۲-۳-۵- دسته ی دوم: عملگر های مجموعه ای

این عملگر ها شامل عملگر های اجتماع ، اشتراک و تفاضل می باشند. هنگام استفاده از این عملگر ها باید شرط همتا بودن<sup>۱</sup> ما بین عملوندهای آن برقرار باشد یعنی:

۱. عملوند ها تعداد ستونهای یکسانی داشته باشند.
۲. ستونهای متناظر در عملوند ها هم نوع (هم دامنه) باشند.

به عنوان مثال دستور زیر صحیح نمیشود :

$$Stud \cup Prof$$

چون با اینکه تعداد ستونهای یکسانی دارند، اما ستونهای متناظر آنها هم دامنه نیستند. به عنوان مثال s# با pname هم دامنه نیست.

مثال: لیست نام افرادی که در دانشکده هستند.

$$\Pi_{Sname}(Stud) \cup \Pi_{Pname}(Prof)$$

مثال : لیست نام دانشجویان هم نام با اساتید.

$$\Pi_{Sname}(Stud) \cap \Pi_{Pname}(Prof)$$

مثال : لیست نام دانشجویان غیر هم اسم با اساتید.

$$\Pi_{Sname}(Stud) - \Pi_{Pname}(Prof)$$

نکته : باید دقت داشته باشید که ترتیب عملوند ها در اجتماع و اشتراک مهم نیست اما در تفاضل مهم می باشد و

داریم:

$$A \cup B = B \cup A$$

$$A \cap B = B \cap A$$

<sup>1</sup> same arity



$$A - B \neq B - A$$

۳-۳-۵- عملگر های پیوند<sup>۱</sup>

زمانی که برای هر یک مسئله به بیش از یک جدول نیاز داشته باشیم یکی از راه حلها استفاده از عملگر های پیوند می باشد.

## ۱-۳-۳-۵- عملگر ضرب دکارتی یا کارتزین

این عملگر دارای دو عملوند میباشد، جدول حاصل از این عملوند مشخص کننده ی کلیه ی حالت های ممکن از تلفیق این دو جدول خواهد بود ممکن است این دو جدول شامل ستونهای همنام باشند در این حالت برای شناسایی ستون مورد نظر از نقطه گذاری استفاده خواهیم کرد. به عنوان مثال:

$$Crs \times Clg$$

کلیه حالت های ممکن از تلفیق دو جدول Crs و Clg را به ما خواهد داد. بخشی از جدول حاصل از این عملگر بصورت زیر خواهد بود.

C#	Cname	Unit	Crs.Clg#	clg.Clg#	Clgname	City	Pname
10172	شبیه سازی	3	10	3	زبان	مشهد	نقره کار
10172	شبیه سازی	3	10	1	ریاضی	تهران	حسنى
10172	شبیه سازی	3	10	11	معماری	یزد	ابوطالبی
10172	شبیه سازی	3	10	12	معارف	تهران	جلالی
10172	شبیه سازی	3	10	10	کامپیوتر	تهران	اشرفی زاده
10172	شبیه سازی	3	10	2	فیزیک	مشهد	جاهد مطلق
10172	شبیه سازی	3	10	4	صنایع	تهران	صادقیان
10172	شبیه سازی	3	10	5	شیمی	تهران	ذاکر
10172	شبیه سازی	3	10	6	مواد	تبریز	مفتون
10172	شبیه سازی	3	10	7	برق	تهران	صادقیان
71203	کنترل خطی	3	7	4	صنایع	تهران	صادقیان
71203	کنترل خطی	3	7	3	زبان	مشهد	نقره کار
71203	کنترل خطی	3	7	7	برق	تهران	صادقیان
71203	کنترل خطی	3	7	5	شیمی	تهران	ذاکر
71203	کنترل خطی	3	7	12	معارف	تهران	جلالی
71203	کنترل خطی	3	7	11	معماری	یزد	ابوطالبی
71203	کنترل خطی	3	7	6	مواد	تبریز	مفتون
71203	کنترل خطی	3	7	1	ریاضی	تهران	حسنى
71203	کنترل خطی	3	7	10	کامپیوتر	تهران	اشرفی زاده

<sup>1</sup> Join

C#	Cname	Unit	Crs.Clg#	clg.Clg#	Clgname	City	Pname
71203	کنترل خطی	3	7	2	فیزیک	مشهد	جاهد مطلق

مثال : نام و شماره ی دروسی که توسط استاد قربانی ارائه می شود را مشخص کنید.

$$\prod_{cname, crs.c\#} (\sigma_{pname="قربانی" \wedge crs.c\#=sec.c\#} (crs \times sec))$$

نکته: ما اکثرا زمانی دو جدول را با همدیگر الحاق خواهیم کرد که با همدیگر ارتباط مستقیم داشته باشند. یعنی کلید اصلی یک جدول داخل جدول دیگر به عنوان کلید خارجی قرار داده شده باشد به همین دلیل در مثال قبل بعد از ضرب دکارتی شرطی که قرار داده شده است برابر بودن کلید خارجی با کلید اصلی می باشد. (crs.c# = sec.c#)

۲-۳-۳-۵- پیوند شرطی<sup>۱</sup>

پیوند شرطی همانند ضرب دکارتی می باشد با این تفاوت که در پیوند شرطی، شرطی نیز محقق می شود، پس از انجام ضرب این شرط بر روی جدول بدست آمده اعمال شده و نهایتا سطر یا سطریهایی باقی میماند که شرط مشخص شده را دارا هستند.

مثال : جواب مساله ی قبلی را این بار با استفاده از پیوند شرطی بررسی کنید.

$$\prod_{cname, crs.c\#} (crs \times_{pname="قربانی" \wedge crs.c\#=sec.c\#} sec)$$

۳-۳-۳-۵- پیوند طبیعی<sup>۲</sup>

پیوند طبیعی با ضرب دکارتی و پیوند شرطی تفاوتی زیر را خواهد داشت:

- به صورت خود کار شرط تساوی روی ستونهای هم نام اعمال می شود. در صورتی که دو جدول ورودی ستون هم نام نداشته باشند نتیجه پیوند طبیعی برابر ضرب دکارتی خواهد بود.
- ستونهای تکراری، حذف خواهند شد و دیگر نیازی به نقطه گذاری نخواهد بود.
- پیوند طبیعی بر خلاف ضرب دکارتی و پیوند شرطی عملگر گرانی نیست. ضرب دکارتی و پیوند شرطی عملگرهای گرانی می باشند چون اولاً سرعت پایینی دارند و ثانياً به حافظه ی زیادی نیاز دارند.

مثال : جواب مساله ی قبلی را این بار با استفاده از پیوند طبیعی بنویسید.

$$۱ \prod_{cname, c\#} (\sigma_{pname="قربانی"} (crs \bowtie sec)) \text{ جواب}$$

$$۲ \prod_{cname, c\#} (crs \bowtie (\sigma_{pname="قربانی"} (sec))) \text{ جواب}$$

$$۳ \prod_{cname, c\#} (crs) \bowtie (\prod_{c\#} (\sigma_{pname="قربانی"} (sec))) \text{ جواب}$$

۴-۳-۳-۵- نیم پیوند<sup>۳</sup>

نیم پیوند همانند پیوند طبیعی می باشد با این تفاوت که جدول حاصله فقط شامل ستونهای عملوند سمت چپ خواهد بود.

<sup>1</sup> Teta join

<sup>2</sup> Natural join

<sup>3</sup> Semi-join

نکته: درست است که گفتیم جدول نتیجه شامل تمام ستونهای عملوند سمت چپ خواهد بود اما ممکن است شامل تمام سطرهای عملوند سمت چپ به عنوان مثال:

### $Stud \propto Sec$

Stud شامل اطلاعات کلیه دانشجویان می باشد و Sec شامل اطلاعات انتخاب واحد دانشجویان می باشد پس از انجام نیم پیوند از بین کل دانشجویان، اطلاعات دانشجویانی باقی می ماند که انتخاب واحد کرده اند ولی اطلاعات دانشجویانی که ثبت نام کرده اند اما انتخاب واحد نکرده اند در جدول نتیجه آورده نخواهد شد.

نکته: باید توجه داشته باشید که  $A \propto B \neq B \propto A$

مثال: جواب مساله ی قبلی را این بار با استفاده از نیم پیوند بنویسید.

$$1 \quad \prod_{cname, c\#} (crs) \propto (\sigma_{pname="قریبانی"} (sec)) \quad \text{جواب}$$

$$2 \quad \prod_{cname, c\#} (crs) \propto (\prod_{c\#} (\sigma_{pname="قریبانی"} (sec))) \quad \text{جواب}$$

مثال: مشخصات کامل رؤسای دانشکده ها.

$$1 \quad prof \propto (\prod_{pname} (clg)) \quad \text{جواب}$$

$$2 \quad prof \propto (\prod_{pname} (clg)) \quad \text{جواب}$$

مثال: خروجی دستور  $prof \propto clg$  چیست؟

جواب: دو جدول prof و clg در ستونهای pname و clg# با همدیگر مشترک می باشند و این دو ستون در این جداول باید با همدیگر برابر باشند پس نتیجه ی این عبارت مشخص کننده ی اطلاعات اساتیدی می باشد که رییس دانشکده ی خودشان می باشند.

مثال: خروجی دستور زیر چیست؟

$$\sigma_{term=771 \wedge clg\#=1} (crs \propto sec)$$

ظاهراً جواب این دستور مشخص کننده ی دروسی می باشد که در ترم اول سال ۷۷ در دانشکده ی شماره ی ۱ ارائه شده است اما در حالی که جواب، جواب صحیحی نمی باشد چون جدول حاصل از  $crs \propto sec$  شامل ستون ترم نمی باشد پس این دستور، دستور صحیحی نیست.

$$\sigma_{term=771} (crs) \propto \sigma_{clg\#=1} (sec)$$

۴-۳-۵- عملگرهای دیگر

۱-۴-۳-۵- عملگر نام گذاری

عملگر نامگذاری با علامت  $\rho_b^a$  مشخص می شود با این کار نام b نیز بر روی جدول a گذاشته می شود. زمانیکه مجبور باشیم در دو طرف عملگر join از یک جدول استفاده کنیم، با استفاده از این عملگر می توان نام یکی از عملوندها را به صورت موقت تغییر داد. به عنوان مثال اگر بخواهیم اساتیدی که دفتر کارشان مشترک باشد را پیدا کنیم خواهیم داشت:

$$prof \times_{prof.office=p.office \wedge prof.pname \neq p.pname} \rho_p (\prod_{pname, office} (prof))$$

## ۲-۴-۳-۵- دستور جایگزینی

از این عملگر استفاده خواهیم کرد تا بتوانیم جدول حاصل از دستورات را برای استفاده های بعدی ذخیره کنیم.  
مثال: اطلاعات دروسی را مشخص کنید که توسط استاد قریانی تدریس می شوند.

$$temp \leftarrow \prod_{c\#} (\sigma_{pname="قریانی"}(sec))$$

$$crs \bowtie temp$$

## ۳-۴-۳-۵- عملگر تقسیم

کاربرد عملگر تقسیم زمانی می باشد که بخواهیم همه ی حالت های یک اتفاق را بررسی کنیم. به عنوان مثال:

- ۱- دانشجویانی که همه ی درس های استاد تبریزی را انتخاب کردند.
  - ۲- اساتیدی که در همه ی دفاتر کار کرده اند.
  - ۳- درس هایی که توسط همه ی دانشکده ها ارائه می شوند.
- برای حل مسئله هایی به از این صورت باید از عملگر تقسیم استفاده کنیم. هنگام استفاده از این عملگر اولین کار مشخص کردن مقسوم علیه می باشد. مقسوم علیه بخشی از مساله می باشد که شامل شرط همه است. سپس مقسوم را بر مقسوم علیه تقسیم کرده تا جواب بدست آید.
- هنگام استفاده از این عملگر باید نکات زیر رعایت شوند :

۱. صفات موجود در مقسوم علیه باید زیر مجموعه ای از صفات مقسوم باشد.
  ۲. جدول حاصل شده از تقسیم شامل صفات مقسوم به غیر از صفات مشترک آن با مقسوم علیه خواهد بود.
- مثال: دانشجویانی که همه ی درس های استاد تبریزی را انتخاب کرده اند.

$$temp \leftarrow \prod_{c\#} (\sigma_{pname="تبریزی"}(sec))$$

$$\prod_{s\#,sname,c\#} (stud \bowtie sec) \div temp$$

## تمرین :

۱. نام و شماره ی دانشجویی ، دانشجویانی که دروس ۴ واحدی دانشکده شماره ۱ را در ترم اول سال ۷۷ انتخاب کرده اند
۲. اطلاعات دانشجویان دانشکده ی کامپیوتر
۳. شماره و نام دروس همچنین تعداد واحد دروسی که دانشجویی به نام علی آنها را انتخاب کرده اند.

## ۵-۳-۵- بهینه سازی

برای یک مساله ممکن است جواب های زیادی وجود داشته باشد اما سعی خواهیم کرد از بین جوابها جواب بهینه را انتخاب کنیم. یعنی دستوری را انتخاب خواهیم کرد که سرعت بالا و مصرف حافظه ی کمتری داشته باشد. برای بهینه سازی سعی می کنیم تا حد ممکن تا قبل از join کردن اندازه ی جداول را کوچک کنیم، این کوچکتر کردن می تواند از لحاظ تعداد ستون یا از لحاظ تعداد سطر انجام شود.

مثال: دروس ۴ واحدی که در ترم اول سال ۷۷ (۷۷۱) ارایه شده اند:

۱ جواب  $\sigma_{unit=4}(crs \propto (\sigma_{term=771}(sec)))$

۲ جواب  $\sigma_{unit=4}(crs) \propto (\sigma_{term=771}(sec))$

۳ جواب  $\sigma_{unit=4}(crs) \propto (\sigma_{term=771}(\prod_{rem,c\#}(sec)))$

۴ جواب  $\sigma_{unit=4}(crs) \propto (\prod_{c\#}(\sigma_{term=771}(sec)))$

در این جوابها مشاهده می کنید که جوابها از جواب شماره ی ۱ تا جواب شماره ی ۴ به ترتیب بهینه می شوند.

### قواعد بهینه سازی

۱. تا حدی که ممکن است باید سعی کنیم عملگر select را قبل از بقیه ی عملگر ها استفاده کنیم.
۲. شرطهای ترکیبی را به شرطهای ترتیبی تبدیل کنید به عنوان مثال می توانیم به جای عبارت  $p1 \wedge p2$  از عبارت  $p1(p2(e))$  استفاده کنیم.
۳. سعی خواهیم کرد تا حد ممکن عملگر project را زودتر انجام دهیم ولی و دیرتر از select .

### استفاده از حالت‌های معادل

می توان برای بهینه از دستورات معادل استفاده کرد ، به عنوان مثال  $crs \propto sec$  از لحاظ جواب معادل  $sec \propto crs$  می باشد اما کارایی آنها با هم برابر نیست، به عنوان مثال فرض کنید جدول crs جدول کوچکی می باشد و می توانیم کل آن را یکجا روی حافظه داشته باشیم اما جدول sec جدول کوچکی نمی باشد و می توانیم بخش کوچک آن را روی حافظه داشته باشیم ، زمانی که به سطری از sec که روی حافظه قرار دارد نیاز داشته باشیم مجبور هستیم بخشی از sec را از حافظه خارج کنیم تا بتوانیم بخش مورد نیاز را به حافظه بیاوریم. این کار باعث کاهش کارایی خواهد شد.

زمانی بخواهیم  $crs \propto sec$  را محاسبه کنیم باید هر یک از سطرهای crs با کل سطرهای sec مقایسه شود. در چنین شرایطی هر یک از بخشهای جدول sec به تعداد سطرهای crs به حافظه آورده شده و از حافظه خارج می شود ولی زمانی که بخواهیم  $sec \propto crs$  را محاسبه کنیم هر یک از سطرهای sec باید با کل سطرهای crs مقایسه شود. در این شرایط هر یک از بخشهای جدول sec فقط یک بار به حافظه آورده و از حافظه خارج می شود که پس این جواب نسبت به جواب قبلی بهینه تر است پس هنگام join کردن سعی خواهیم کرد بجای اینکه یک جدول کوچک را با یک جدول بزرگ join کنیم ، یک جدول بزرگ را با یک جدول کوچک join خواهیم کرد.

### دستورات معادل

$$A \propto B = B \propto A$$

$$(A \propto B) \propto C = A \propto (B \propto C)$$

$$A \times B = B \times A$$

$$(A \times B) \times C = A \times (B \times C)$$

$$A \cup B = B \cup A$$

$$(A \cup B) \cup C = A \cup (B \cup C)$$

$$A \cap B = B \cap A$$

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

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

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

## ۵-۳-۶- تغییر داده های جداول

تغییر داده های جداول در ۳ بخش انجام می شود.

۱. افزودن داده به جدول (Insert)
۲. حذف داده ها از جدول (delete)
۳. تغییر داده های جدول (update)

## ۵-۳-۶-۱- افزودن داده به جدول

در جبر رابطه ای افزودن داده به جدول با استفاده از اجتماع و انتساب انجام می شود.

مثال: می خواهیم اطلاعات درس جدیدی به نام database را به جدول crs اضافه کنیم این کار بصورت زیر انجام خواهد شد.

$$crs \leftarrow crs \cup \{ "C14", "database", 3, 4 \}$$

در این مثال با یک دستور فقط یک سطر به جدول crs اضافه می شود می توانیم با یک دستور بیش از یک سطر را به جدول اضافه کنیم.

مثال: جدول good stud را در نظر بگیرید که شامل صفات sname و avg می باشد اگر بخواهیم اطلاعات دانشجویان ممتاز دانشکده ی شماره ی ۱۰ را به این جدول اضافه کنیم خواهیم داشت:

$$goodstud \leftarrow goodstud \cup (\Pi_{sname, avg} (\sigma_{avg \geq 17 \wedge c1g \# = 10} (stud)))$$

## ۵-۳-۶-۲- حذف داده ها از جدول

این کار در جبر رابطه ای با استفاده از تفاضل و جایگزینی انجام می شود. با این کار ممکن است یک یا چند سطر از جدول حذف شود یا ممکن است هیچ سطری از جدول حذف نشود.

مثال: دانشجویانی که معدل آنها کمتر از ۱۸ می باشد از جدول goodstud حذف کنید:

$$goodstud \leftarrow goodstud - (\sigma_{avg < 18} (goodstud))$$

## ۵-۳-۶-۳- تغییر داده های جدول

این کار در جبر رابطه ای با استفاده از گزینش و جایگزینی انجام می شود.

مثال: افزودن یک واحد به تمام دروس

$$\sigma_{unit \leftarrow unit + 1} (crs)$$

مثال: تغییر تعداد واحد درس database از ۳ به ۴.

$$\sigma_{unit \leftarrow 4} (\sigma_{cname = "database"} (crs))$$

## فصل ششم

## ۶- آشنایی با یک زبان رابطه ای

۶-۱- زبان SQL<sup>۱</sup>

زبان SQL یکی از زبانهایی می باشد که برای پایگاه داده ها مدل رابطه ای طراحی شده است. نسخه های SQL عبارتند از SQL1, SQL2, SQL3.

هر زبان پایگاه داده ها از ۳ بخش زیر تشکیل شده است:

- <sup>۲</sup> DDL زبان تعریف داده ها
- <sup>۳</sup> DML زبان کار با داده ها
- <sup>۴</sup> DCL زبان کنترل داده ها

نکته: کلیه مثالهای این بخش بر اساس جداول زیر می باشد

STT (STID, STNAME, STDEG, STMGR, STDEID)  
 (شماره گروه آموزشی، رشته تحصیلی، سطح دوره تحصیل، نام، شماره) دانشجو  
 COT(COID, COTITLE, CREDIT, COTYPE, CODEID)  
 (شماره گروه آموزشی ارائه کننده درس، نوع، تعداد واحد، عنوان، شماره) درس  
 STCOT (STID, COID, TR, YRYR, GRADE)  
 (نمره، سال تحصیلی، ترم، شماره درس، شماره دانشجو) درس دانشجو

شکل ۶-۱: پایگاه داده دانشگاه (رانکوهی)

## ۶-۲- انواع داده ای

Char(n), Varchar(n)  
 Integer, Smallint, Numeric(p,d), Real, Double, Float, Bit  
 Data, Time, TimeStamp, Interval

ضمیمه B انواع داده ای SQL و کاربرد آنها را مشخص می کند.

## ۶-۳- دستور تعریف میدان

**Create Domain** domain\_name datatype  
 [default-definition]  
 [domain-constraint-definition-list]

برای تعریف دامنه از دستور Create Domain استفاده می شود. بخش Default که نوشتن آن اختیاری می باشد برای مشخص کردن مقدار پیش فرض دامنه استفاده می شود، و بخش constraints نیز که نوشتن آن اختیاری است به منظور اعمال محدودیت به دامنه می باشد.

<sup>1</sup> SQL(Standard Query Language)

<sup>2</sup> DDL (Data Declaration Language)

<sup>3</sup> DML (Data Manipulation Language)

<sup>4</sup> DCL (Data Control Language)

مثال:

```

Create Domauin Degree Char(3) Default '???'
Constraints Valid_Degrees
Check Value in ('bs','ms','doc','???')

```

۴-۶- دستور تغییر دامنه

```

Alter Domain domain-name

```

این دستور میتواند تغییراتی از جمله تغییر در مقدار پیش فرض، تغییر محدودیتهای جامعیتی و برخی امکانات دیگر ایجاد کند.

۵-۶- دستور حذف میدان

```

Drop Domain domain-name option

```

با این دستور میتوان دامنه تعریف شده را حذف کرد. در این دستور option میتواند یکی از عبارات Cascade و Restrict باشد. تمرین: عملکرد Restrict و Cascade در دستور Drop Domain بررسی کنید.

۶-۶- دستور ایجاد جدول

```

Craete Table table_name
{ (ColumnName DataType[Not Null][Unique]
  [Default defayltOption][Check (search condition)][,...] ) }
[Primary Key (list of Columns),]
{[Unique (listOfColumns),][,...]}
{[Foreign Key (listOfForeignKeyColumns)
Refrences ParentTableName [(ListOfCondidateColumns)],
  [match {partial | full}]
  [on update RefrentialAction]
  [on delete RefrentialAction]][,...]}
{[Check (SearchCondition)][,...]}

```

دستور Create Table برای ایجاد جدول استفاده میشود در این دستور همچنان که شکل کلی آن نشان میدهد ابتدا باید نام ستون و نوع داده ای آن را مشخص کنیم. اگر برای یک ستون عبارت not null مشخص شود که هنگام وارد کردن داده به برای جدول صفت مربوطه نمیتواند خالی باشد، اگر برای یک ستون عبارت Unuque مشخص شده باشد نشان میدهد که مقدار ستون باید منحصر به فرد باشد و نمیتواند تکراری باشد. بخش default نیز مقدار پیش فرض را نشان میدهد. یک جدول میتواند بیش از یک ستون داشته باشد. بخش Primary Key مشخص کننده کلید اصلی جدول میباشد. بخش Foreign Key مشخص کننده کلید خارجی جدول است و بعد عبارت Refrences باید نام جدولی را مشخص کنیم که این کلید خارجی به آنجا ارجاع داده میشود. برای هر کلید خارجی می توان حالت On Delete و On Update را تعریف کرد. برای هر یک از این حالتها میتوان یکی از فعالیتهای زیر انجام شود:

- 1-No Action
- 2-Cascade
- 3-Set Null



## 4-Set Default

یک جدول می تواند بیش از یک کلید خارجی داشته باشد. بخش Check نیز برای تعریف محدودیتهای جامعیتی بر روی جدول استفاده شود.

مثال:

```

Create table STCOT
(STID stnum not null,
COID conum not null,
TR Term not null,
YRYR year not null,
Garde grade,
Primary key(STID,COID),
Foreign Key (STID) refrences STT
On update cascade
On delete cascade,
Foreign key(COID) refrences COT
On update cascade
On delete cascade,
Check (TR>=1 and TR<=3)
Check (YRYR>=60-61 and YRYR<=98-99)
Check (Grade>=0 and Grade<=20);

```

**Alter Table** table-name alteration

۶-۷- دستور تغییر جدول

بخش Altaeration مشخص کننده نوع تغییر بوده و یکی از مقادیرها ی زیر را دارد:

۱. اضافه کردن ستون به جدول ADD Column
  ۲. تغییر ستون یک جدول ALTER Column
  ۳. حذف ستون از جدول DROP Column
  ۴. وضع یک قاعده جامعیتی جدید ADD Constraint
  ۵. حذف قاعده جامعیتی از جدول DROP Constraint
- با توجه به این موارد دستور تغییر جدول چنین خواهد بود:

```

Alter Table TableName
[Add [Column] ColumnName DataType [Not Null][Unique]
[Default DefaultOptions] [Check (SearchOptions)]]
[Drop [Column] ColumnName [Restrict | Cascade]]
[Add [Constrant [ConstrantName]] TableConstrantDefinition]
[Drop Constrant ConstrantName [Restrict | Cascade]]
[Alter [Column] Set Default DefaultOptions]
[Alter [Column] Drop Default]

```

مثال زیر ستون وضعیت را به جدول STT اضافه میکند:

<i>Alter Table</i> STT
<i>Drop Table</i> table_name Option;

### ۸-۶- دستور حذف جدول

در این تعریف Option می تواند یکی از عبارات Cascade یا Restrict باشد.

### ۹-۶- دستور بازیابی

این دستور یکی از قدرتمندترین دستورات در SQL است که میتواند اعمال select، project و join جبر رابطه ای را انجام دهد. مقابل Select نام ستونهایی را می آوریم که می خواهیم اطلاعات آنها در خروجی آورده شود (عمل project جبر رابطه ای) و اگر علامت \* آورده شود به منزله همه ستون ها است.

<p><i>Select</i> [Distinct   All] item(s)-list  <i>From</i> table(s)-list            [Where Condition(s)]            [Group by column(s)]            [Having Condition(s)]</p>
--

در بخش From نام جدول یا جداولی را مشخص میکنیم که می خواهیم اطلاعات را از آنها استخراج کنیم. اگر در این بخش بیش از یک جدول نوشته شود عمل join رابطه ای از نوع ضرب دکارتی انجام خواهد شد اما اگر فقط یک جدول نوشته شود عمل join انجام نخواهد شد.

در بخش where شرطی را مشخص خواهیم کرد و این شرط بر روی تک تک سطرهاى جدول اعمال می شود. در صورتی که سطر شرط مشخص شده را دارا باشد در خروجی آورده خواهد شد و در غیر این صورت در خروجی آورده نمی شود (عمل select جبر رابطه ای).

یک دستور SQL نمونه بصورت زیر می باشد:

<p><i>select</i> A<sub>1</sub>, A<sub>2</sub>, ..., A<sub>n</sub>  <i>from</i> r<sub>1</sub>, r<sub>2</sub>, ..., r<sub>m</sub>  <i>where</i> P</p>
---

در این مثال A<sub>j</sub> ها صفت، r<sub>j</sub> ها رابطه و P گزاره شرطی می باشند.

دستور جبر رابطه ای معادل با این دستور برابر است با:

$$\prod_{A_1, A_2, \dots, A_n} (\sigma_P(r_1 \times r_2 \times \dots \times r_m))$$

مثال: دستور select بنویسید که نام و شماره دانشجویان دوره کارشناسی را مشخص کند؟

<p><i>Select</i> STID,STname  <i>From</i> STT  <i>Where</i> SREDG='bs';</p>
---

مثال: مشخصات کامل دانشجویان گروه آموزشی D222 را بنویسید؟

```
Select STID,STName,STDEG,STMJR,STDEID
From STT
Where STDEID = 'D222';
```

```
Select *
From STT
Where STDEID = 'D222';
```

نکته: همچنان که در مثال قبل دیده میشود می توان بجای مشخص کردن کلیه ستونها از \* استفاده کرد.

### ۱-۹-۶- مرتب سازی خروجی

برای مرتب سازی خروجی میتوان از Order by استفاده کرد. Order by جزء آخرین بخشهای دستور Select می باشد و می توان خروجی را بر اساس صفت یا صفات مشخص شده مرتب کند. پیش فرض مرتب سازی به صورت صعودی است (ASC) اما میتوان با استفاده از قید (DESC) ترتیب مرتب سازی را به نزولی تغییر داد.

مثال: دستور زیر شماره و تعداد واحد تمامی دروس را مشخص کرده و آنها را بر اساس تعداد واحد بصورت نزولی مرتب می کند.

```
Select COID,CREDIT
From COT
Order by CREDIT DESC;
```

### ۲-۹-۶- استفاده از عبارات اسکالر در دستور SELECT

میتوان در بخش item(s)-list در دستور SELECT از عبارات اسکالر جمع , تفریق , ضرب و تقسیم استفاده کرد.

مثال: دستوری بنویسید که شماره و عنوان هر کتاب و قیمت آنرا به ریال بدهد. (جدول Book بصورت زیر است Book(BKID,BKTITLE,BKPRICE,.....))

```
Select BKID , BKTITLT , BKPRICR*f
From Book;
```

در این مثال f ضریب تبدیل دلار به ریال است.

### ۳-۹-۶- دگر نامی (قید AS)

با استفاده از قید AS می توان نام دیگری برای یک ستون در نظر گرفت. به عنوان مثال در دستور زیر با استفاده از قید AS برای ستون دوم نام BT و برای ستون سوم نام PR انتخاب شده است.

```
Select BKID , BKTITLE AS BT , BKPRICE*f AS PR
From Book;
```

۴-۹-۶- توابع جمعی<sup>۱</sup>

در SQL برخی توابع همچون Avg, Count, Sum, Min, Max قرار داده شده است. این توابع تنها استفاده نمیشود بلکه همراه با دستور Select استفاده خواهند شد. هنگام استفاده از این توابع می توان از قید All یا Distinct استفاده کرد. Distinct سبب حذف تکرارها میشود ولی All باعث عدم حذف تکرار خواهد شد. نکته: در تابع Count قید Distinct باید لحاظ شود. اما در count به صورت Count(\*) که تعداد کل سطرها را مشخص میکند، استفاده از Distinct مجاز نیست. همچنین در توابع Max و Min استفاده از Distinct بی معنی است و سیستم انرا نادیده میگیرد.

مثال: دستوری بنویسید که تعداد کل درسها را تعیین کند.

```
Select Count(*)
From COT;
```

مثال: تعداد درسهایی که دانشجو با شماره ۷۸۱۱۰۵۵۵ آنها را انتخاب نموده و نمره قبولی گرفته.

```
Select Count (Distinct COID)
From STCOT
Where STID = '78110555'
And Grade >= 10;
```

نکته: در این مثال Distinct COID باعث حذف شماره درسهای تکراری می شود و Count تعداد درسها را بعد از حذف تکرار مشخص می کند.

مثال: دستوری بنویسید که بیشترین و کمترین نمره را در ترم دوم سال ۷۹-۸۰ برای درس Com222 بدست آورد

```
Select Max(Grade),Min(Grade)
From StCOT
Where TR = '2'
And YR YR = '79-80'
And COID = 'Com222';
```

## ۵-۹-۶- دستور Like و Not like

برای الحاق رشته ها مورد استفاده قرار می گیرد و می تواند رشته ها را با الگوی خاصی مقایسه کند. در این دستور دو کاراکتر ویژه % به مفهوم هر تعداد کاراکتر و \_ (underscore) به مفهوم فقط یک کاراکتر استفاده می شود. مثال : مشخصات اساتیدی که آخر نامشان به Y ختم می شود.

```
Select *
From Prof
Where Pname like '%Y';
```

مثال : مشخصات اساتیدی که نام آنها ۸ کاراکتری بوده و کاراکتر ۳ و ۴ آن BA باشد.

```
Select *
From Prof
Where Pname like '__BA__';
```

## ۶-۹-۶- آزمون وجود هیچمقدار در یک ستون

برای تست خالی بودن یک ستون می توان از Is Null استفاده کرد.

مثال: شماره دانشجویانی که نمره آنها در درس SOC33658 در ترم دوم ۷۸-۷۹ هنوز اعلام نشده است.

```

Select STID
From STCOT
Where TR = '2'
      And YRYR = '78-79'
      And COID = 'SOC33658'
      And Grade Is Null;

```

## ۶-۹-۷- Union All و Union

این عملگر برای بدست آوردن اجتماع بکار می رود. هنگام استفاده از Union سطرهای تکراری حذف میشود ولی در Union All اینطور نیست.

نکته: این دستور دو عملوند دارد و باید شرط همتا بودن مابین عملوندهای آن وجود داشته باشد.

مثال: شماره دانشجویانی که یا در دوره کارشناسی باشند یا درس Com36584 را انتخاب کرده باشند.

```

Select STID
From STT
Where STDEG = 'bs'
Union [All]
Select STID
From STCOT
Where COID = 'Com36584';

```

## ۶-۹-۸- گروه بندی

با استفاده از Group by می توان سطرهای جدول را بر حسب مقدار یک ستون ساده (صفت ساده) گروه بندی کرد به نحوی که در هر گروه، مقدار آن ستون یکسان باشد. هنگام استفاده از Group by باید موارد زیر رعایت شود:

- صفتی که گروه بندی بر اساس آن انجام شده است حتماً باید در خروجی Select آورده شود.
- به غیر از صفتهای گروهبندی اگر صفت دیگری را بخواهیم در خروجی بیاریم حتماً باید همراه با توابع جمعی باشند.
- نمی توان آنرا در پرسشهای فرعی استفاده کرد.

مثال: میاگین نمرات هر درس

```
Select COID, Avg(Grade) As Avggr
From STCOT
Group by COID;
```

```
Select *
From STT
Where STEDID = 'D222'
Group by STDEG;
```

مثال: دانشویان گروه آموزشی 'D222' را بر حسب سطح دوره تحصیلی گروه بندی کنید.

### Having - ۶-۹-۹

با استفاده از این بخش می توان شرطی را بر روی هر یک گروهها اعمال کرد. در واقع نقش having بر روی گروه ها همانند نقش where بر روی سطر ها می باشد. اما باید توجه داشته باشید که having با group by می آید. مثال: شماره درسهایی که در ترم دوم ۷۹-۷۸ کمتر از ۱۰ دانشجو در آنها ثبت نام کرده اند

```
Select STCOT.COID
From STCOT
Where TR = '2'
And YRYR = '78-79'
Group by COID
Having count(*) < 10;
```

نکته: امکانات Group by و امکانات Having از نظر برخی از پژوهشگران بویژه دیتل زیادی و نالازم می باشد چون می توانیم آنها را استفاده از برخی امکانات SQL بدست آوریم. بعنوان مثال می توانیم مثال قبل را بصورت زیر نیز بنویسیم:

```
Select (distinct COID)
From STCOT
Where ( Select count(*)
From STCOT As T
Where T.COID = STCOT.COID
And T.TR = '2'
And T.YRYR = '78-79' ) < 10
```

### Between - ۶-۹-۱۰

شکل کلی این دستور بصورت زیر می باشد:

```
Scalar-expression [NOT] BETWEEN Scalar-expression AND scalar-expression
```

مثال: شماره دانشجویانی را بدهید که نمره آنها در درس HIS444 در ترم اول ۷۸-۷۷ بین ۱۵ و ۱۹ باشد.

```
Select STCOT.SID
From STCOT
Where COID = 'HIS444'
And YRYR = '77-78'
And GRADE Between '15' And '19';
```

## ۱۱-۹-۶- پیوند در Select

ممکن است برای حل یک مساله به بیش از یک جدول نیاز داشته باشیم، در چنین حالتی باید این جداول را با هم پیوند دهیم، به این منظور نام جداول را در بخش From خواهیم نوشت. اکثراً در چنین مواردی باید شرط برابر بودن کلید خارجی و کلید اصلی را در بخش Where بنویسیم.

**مثال:** نام دانشجویانی را مشخص کنید که درس SOC6820 را انتخاب کرده اند؟

مشخصات دانشجو در جدول STT و مشخصات دروس انتخابی در جدول STCOT قرار دارد. لذا برای حل این مثال به دو جدول نیاز خواهیم داشت. همچنین این دو جدول از طریق STID باهم در ارتباط هستند پس باید شرط برابری این صفت در دو جدول در بخش where نوشته شود.

```
Select STT.STname
From STT,STCOT
Where STT.STID = STCOT.STID
And STCOT.COID = 'SOC6820';
```

**مثال:** دستوری بنویسید که شماره جفت دانشجویان از یک گروه آموزشی را بدهد؟

برای حل این مساله جدول دانشجو را با خودش Join خواهیم کرد. در چنین حالتی باید نام یکی از جداول یا هر دو را تغییر دهیم. برای تغییر نام موقت یک جدول در دستور Select در بخش From بعد از نام جدول مستقیماً نام دیگر آن را می آوریم.

```
Select Ftab.STID,Stab.STID
From STT Ftab,STT Stab
Where Ftab.STDEID=Stab.STDEID
And Ftab.STID < Stab.STID;
```

۱۲-۹-۶- زیر دستور<sup>۱</sup>

ممکن است داخل دستور Select از یک دستور Select دیگر استفاده شود. به اینگونه دستورات پرسش تودرتو یا چند سطحی گفته می شود. در چنین حالتی ابتدا جواب دستور Select داخلی سپس دستور Select بیرونی مشخص می گردد.

**مثال:** نام دانشجویانی که درس SOC43972 را انتخاب کرده اند؟

```
Select Sname
From STT
Where STID in (
Select STID
From STCOT
Where COID = 'SOC43972' );
```

**مثال:** نام دانشجویان هم رشته با دانشجوی شماره ۹۵۲۰۶۸ را بدست آورید؟

<sup>1</sup> Subquery

```

Select STNAME
From STT
Where STMJR = (
    Select STMJR
    From STT
    Where STID = '952068');

```

## ۱۳-۹-۶- سور وجودی Exists و Not Exists

مثال: نام دانشجویانی که درس SOC333 را انتخاب کرده باشند.

```

Select STNAME
From STT
Where Exists ( Select *
                From STCOT
                Where STCOT.STID = STT.STID
                And COID = 'SOC333');

```

به ازاء هر سطر از جدول STT سیستم بررسی می کند: آیا وجود دارد سطری در جدول STCOT، به نحوی که STID آن برابر باشد با STID آن سطر از جدول STT و COID آن برابر با 'SOC333' باشد؟ اگر پاسخ ارزیابی سیستم درست باشد STNAME از آن سطر جدول STT، پاسخ پرسش است.

نکته: به شرط داده شده در بخش Where از دستور select درونی دقت کنید. شرط اول یعنی STCOT.STID = STT.STID ظاهراً شبیه شری است که در عمل پیوند دیدیم، اما سیستم عمل پیوند انجام نمی دهد. نکته: در همین گزاره ذکر نام STT به عنوان قید ستون STID الزامی است ولی ذکر نام جدول STCOT برای همین ستون اختیاری و برای وضوح بیشتر است. مثال: نام اساتیدی که حد اقل یک اثر منتشره دارند.

```

Select PROF.PNAME
From PROF
WHERE Exists (Select *
              From PRPUB
              Where PRPUB.PRID = PROF.PRID);

```

```

Select STNAME
From STT
Where Not Exists (Select *
                  From STCOT
                  Where STCOT.STID = STT.STID
                  And STCOT.COID = 'SOC333');

```

مثال: نام دانشجویانی را بدهد که درس SOC333 را انتخاب نکرده اند.



## ۱۰-۶- دستورات عملیات ذخیره سازی

## ۱-۱۰-۶- دستور Update

```
Update table-name
Set assignment-commalist
[Where condition(s)]
```

شکل کلی این دستور بصورت زیر می باشد:

```
Update PROF
Set RANK = 'Asso.Prof'
Where PRID = 'PR7777';
```

مثال: بهنگام سازی تک سطر. مرتبه دانشگاهی استادی با شماره PR7777 را از استادیاری به دانشیاری تغییر دهید.

مثال: بهنگام سازی چند سطر. تعداد واحدهای درسهای عملی گروه آموزشی 'D111' را برابر با "یک" کنید.

```
Update COT
Set CREDIT = '1'
Where COTYPE = 'p'
And CODEID = 'D111';
```

## ۲-۱۰-۶- دستور Delete

شکل کلی این دستور بصورت زیر می باشد:

```
Delete
From table-name
[Where condition(s)]
```

مثال: حذف تک سطر: درس شماره COM111 را برای دانشجویی با شماره 78110555 حذف کنید.

```
Delete
From STCOT
Where STID = '78110555'
And COID = 'COM111';
```

مثال: حذف چند سطر: گروه آموزشی با شماره D333 را حذف کنید.

```
Delete
From DEPT
Where DEID = 'D333'
```

**۳-۱۰-۶- دستور Insert**

این دستور دو شکل کلی دارد:

```
Insert Into table-name
Values (one row);
```

```
Insert Into table-name
Sbquery;
```

مثال: درج تک سطر: درج اطلاعات درس زیر:

```
< 78110888 , COM888 , 2 , 77-78 , 12 >
```

```
Insert Into STCOT
Values < '78110888' , 'COM888' , '2' , '77-78' , '12' >
```






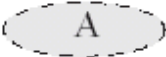

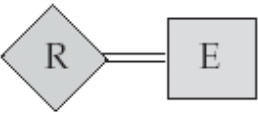

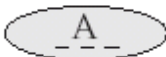



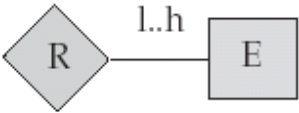
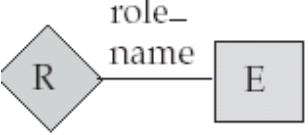


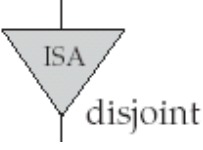
نکته: سطر درج شدنی می تواند کامل یا ناقص باشد.

مثال: درج چند سطر:

```
Insert Into WORKTAB
Select COID, Avg(GRADE)
From STCOT
Group by COID;
```

ضمیمه A

مجموعه نمادهای استفاده شده برای نودهای ER در کتاب "Database System Concept" تالیف Silberschatz.

	entity set		attribute
	weak entity set		multivalued attribute
	relationship set		derived attribute
	identifying relationship set for weak entity set		total participation of entity set in relationship
	primary key		discriminating attribute of weak entity set
	many_to_many relationship		many_to_one relationship
	one_to_one relationship		cardinality limits
	role indicator		ISA (specialization or generalization)
	total generalization		disjoint generalization

## ضمیمه B

## انواع داده‌ای SQL

1. **char(n)**. Fixed length character string, with user-specified length  $n$ .
2. **varchar(n)**. Variable length character strings, with user-specified maximum length  $n$ .
3. **int**. Integer (a finite subset of the integers that is machine-dependent).
4. **smallint**. Small integer (a machine-dependent subset of the integer domain type).
5. **numeric(p,d)**. Fixed point number, with user-specified precision of  $p$  digits, with  $n$  digits to the right of decimal point.
6. **real, double precision**. Floating point and double-precision floating point numbers, with machine-dependent precision.
7. **float(n)**. Floating point number, with user-specified precision of at least  $n$  digits.
8. **date**: Dates, containing a (4 digit) year, month and date
  - Example: **date** '2005-7-27'
9. **time**: Time of day, in hours, minutes and seconds.
  - Example: **time** '09:00:30'      **time** '09:00:30.75'
10. **timestamp**: date plus time of day
  - Example: **timestamp** '2005-7-27 09:00:30.75'
11. **interval**: period of time
  - Example: interval '1' day
  - Subtracting a date/time/timestamp value from another gives an interval value
  - Interval values can be added to date/time/timestamp values
  - Can extract values of individual fields from date/time/timestamp
    - Example: **extract (year from r.starttime)**
  - Can cast string types to date/time/timestamp
    - Example: **cast** <string-valued-expression> **as date**
    - Example: **cast** <string-valued-expression> **as time**

n