

بسمه تعالی



# فصل سوم

## انواع ساختارهای داده‌ی (مدل داده‌ی)

### و

## مفاهیم بانک اطلاعاتی رابطه‌ای

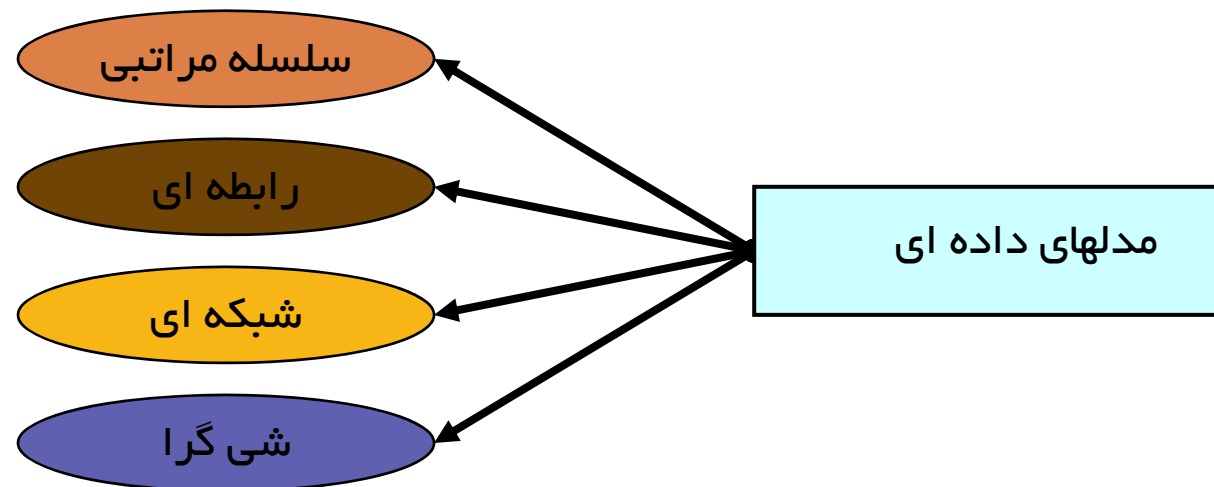
### (Relational Data Model)



## انواع ساختارهای داده‌ی

### ساختارهای داده‌ی (مدل داده‌ی) :

با آنکه ساختار داده‌ی متعددی برای طراحی سطوح ادراکی و خارجی بانک وجود دارد ما در این فصل سه ساختار رابطه‌ای، سلسله‌مراتبی و شبکه‌ای را شرح می‌دهیم. البته امروزه ساختار شی‌گرا نیز معروفیت زیادی یافته است.





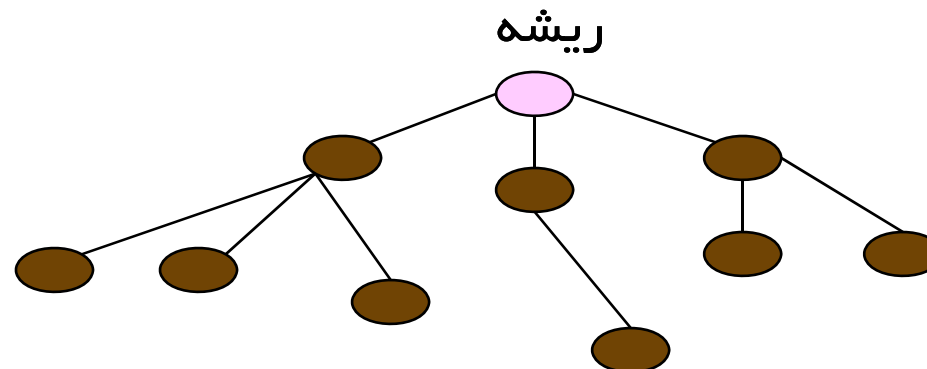
# انواع ساختارهای داده‌ی

## ۱- ساختار سلسله‌مراتبی :

این ساختار قدیمی‌ترین ساختار داده‌ی برای طرحی بانک اطلاعاتی در سطح انتزاعی است. در ساختار داده‌ها و ارتباط بین آنها به کمک یک درختواره نمایش داده می‌شوند.

درختواره گرافی است دارای یک ریشه، به هم بسته و غیر چرخشی. منظور از به هم بسته این است که بین هر دو گره پیوندی وجود دارد. غیرچرخشی یعنی مسیری از گره سطح پائین‌تر به گرهی از سطح بالاتر وجود ندارد. رابطه همواره از سطح بالاتر به سطح پائین‌تر است. هر گره پدر می‌تواند چندین فرزند داشته باشد ولی هر فرزند فقط یک پدر دارد.

در این مدل یک موجودیت بعنوان پدر (گره ریشه) و موجودیتهای دیگر بعنوان فرزند در نظر گرفته میشوند این رابطه برای حالت  $1 : n$  خوب است ولی در حالت چند به چند با مشکل مواجه می‌شود.



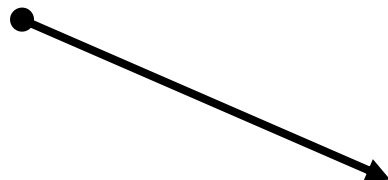


## انواع ساختارهای داده‌ی



مثالی از مدل ساختار سلسله مراتبی :

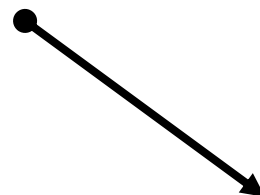
S1	نام تولیدکننده	شهر	آدرس	.....
----	----------------	-----	------	-------



۱۵	رنگ قطعه	نام قطعه	P1
۱۲	رنگ قطعه	نام قطعه	P2

S2	نام تولیدکننده	شهر	آدرس	.....
----	----------------	-----	------	-------

S3	نام تولیدکننده	شهر	آدرس	.....
----	----------------	-----	------	-------



۱۰	رنگ قطعه	نام قطعه	P2
۳۰	رنگ قطعه	نام قطعه	P4



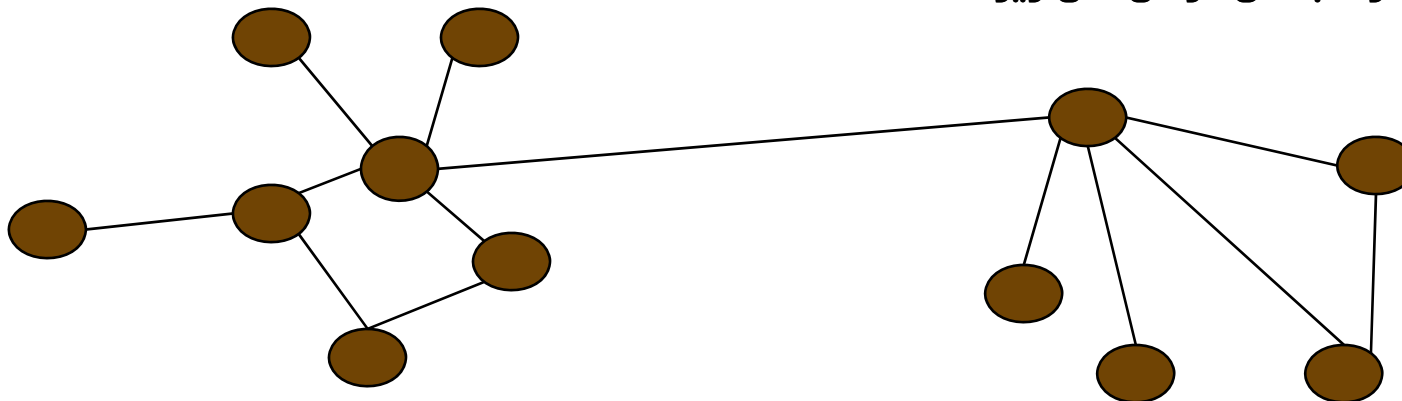
## اشکالات مدل سلسله مراتبی :

- ۱- برای جستجو باید بانک اطلاعاتی را از ابتدا تا انتها پیمایش کرد و این امر زمانبر است (مثلاً وجود P2 توسط دوتولید کننده است)
- ۲- طول رکوردها متغیر است.
- ۳- افزونگی داده‌ها زیادتر است. (اطلاعات موجودیت فرعی تکرار می‌شود)
- ۴- در عملیات ذخیره سازی مشکل دارد تغییرات اعمال شده در طک موجودیت فرعی باید در تمام موجودیتهای فرعی موجود لحاظ شود. یعنی درج یا حذف سخت تر است.
- ۵- درمورد موجودیتهای فرعی که رابطه ای با موجودیت اصلی ندارند نمیتوانیم اطلاعاتی را نگهداری کنیم. مثلاً ازقطعه P3 چون هنوز توسط هیچ تولیدکننده ای (موجودیت اصلی) ، تولید نمیشود اطلاعاتی قابل نگهداری نیست.
- ۶- تقارن ساختار جدولی را ندارد.



## ۲- ساختار شبکه‌ای :

ساختار شبکه‌ای که به آن ساختار پلکس (PLEX) نیز می‌گویند. در این ساختار هر گره فرزند می‌تواند بیش از یک گره پدر داشته باشد. این ساختار که جامع‌تر از ساختار سلسله‌مراتبی است برای نمایش ارتباطات یک به چند دوسویه مناسب است. در واقع ساختار سلسله‌مراتبی حالت خاصی از ساختار شبکه‌ای است و می‌توان با پذیرش مقداری افزونگی به ساختار سلسله‌مراتبی تبدیل کرد. در واقع ساختار شبکه‌ای گرافی مثل زیر است :

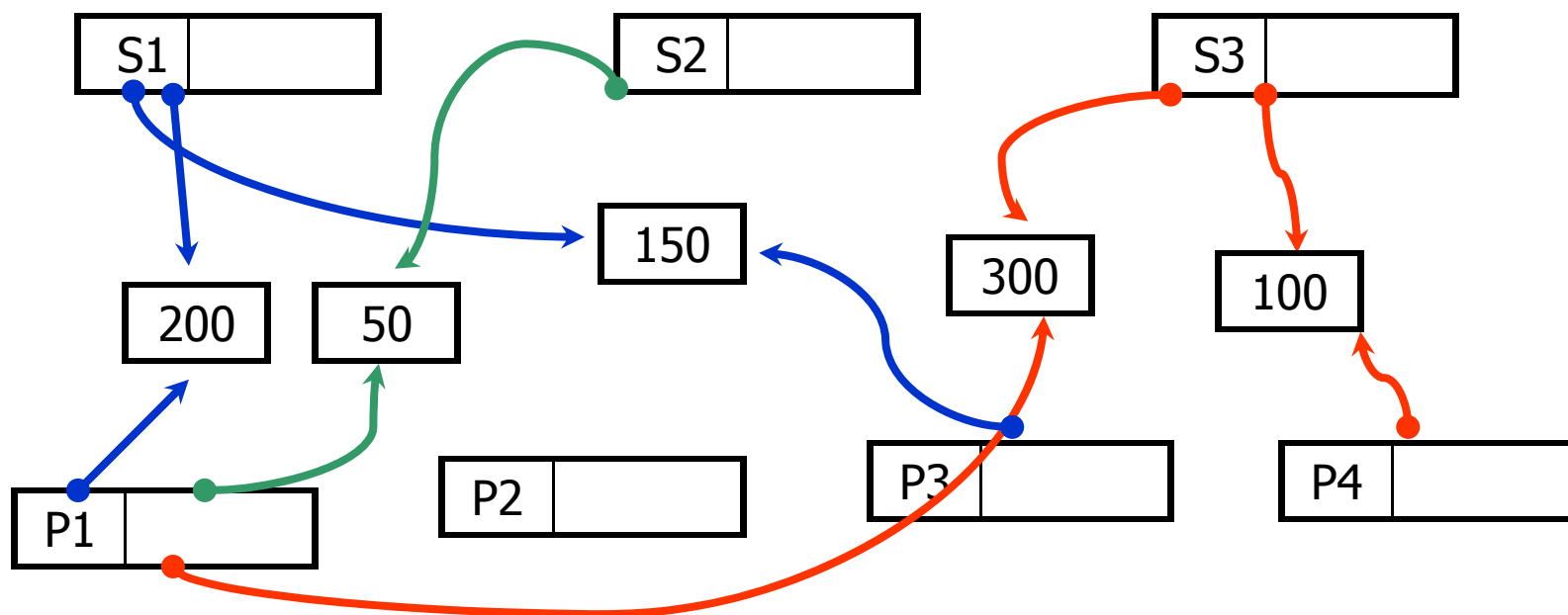


در مدل شبکه‌ای اطلاعات در گره‌های یک گراف دلخواه ذخیره می‌شوند. در این گراف هر گره دلخواه می‌تواند چند والد و چند فرزند داشته باشد و این مدل برای نمایش روابط داده‌ای چند به چند مناسب است.



## معایب روش شبکه‌ای :

- این مدل مشکل تکرار و نیز اشکال مورد ۵ سلسله مراتبی را حل کرده ولی دارای مشکلات است.
- افزودگی داده‌ها و ناسازگاری داده‌ها در این روش نسبت به سلسله مراتبی کمتر است .
- اما به علت حجم زیاد اشاره‌گرها ممکن است سبب بروز فزونکاری و همچنین افزایش پیچیدگی در سیستم بشود.

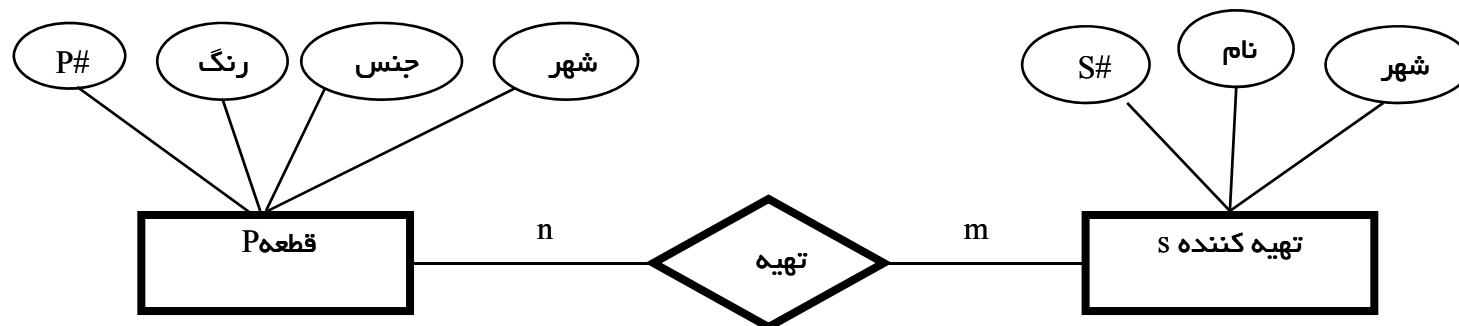




## ۳- ساختار رابطه‌ای (Relation Data Base Management System) :

در سال ۱۹۷۰ دکتر کاد عضو آزمایشگاه تحقیقاتی سن جوز IBM تئوری ریاضی پایگاه داده‌ای رابطه‌ای را که اینکه چگونه میتوان داده‌ها را با استفاده از ساختار جدولی ذخیره و مدیریت نمود را ارائه کرد. از دید کاربر بانک اطلاعاتی تشکیل شده است از تعدادی جدول.

رابطه، مفهومی ریاضی است. اما از دید کاربر، رابطه نمایشی جدولی دارد. جدول ساختاری است نامدار که از تعدادی سطر و ستون تشکیل یافته است. هر ستون نمایشگر یک صفت خاصه از یک نوع موجودیت است و هر سطر نمایشگر یک نمونه از یک نوع موجودیت می‌باشد. در این مدل یک سری قوانین وجود دارد که اگر جداول آن قوانین را رعایت کنند مدل رابطه‌ای هستند و گرنه جدول هستند. در واقع هر رابطه‌ای جدول است ولی هر جدولی یک رابطه نیست. مفاهیم ساختار جدولی عبارتند از: جدول - سطر - ستون موجودیتهای قطعه و تولید کننده را در نظر گرفته و نمودار EER آن را ترسیم می‌کنیم.







## انواع ساختارهای داده‌ی

برای تشریح نمودار فوق در بانک رابطه‌ای یک جدول برای هر یک از دو موجودیت و جدولی نیز برای بیان ارتباط آنها، استفاده می‌شود:

S#	Name	City
S1	فن آوران	تهران
S2	ایران قطعه	تبریز
S3	پولادین	تبریز

جدول تهیه کنندگان (S)

P#	Name	Color	جنس	City
P1	محصول ۱	قرمز	آهن	تهران
P2	محصول ۲	سبز	مس	تبریز
P3	محصول ۳	آبی	برنج	شیراز
P4	محصول ۴	قرمز	آهن	تهران

جدول قطعه (P)

S#	P#	تعداد Qty
S1	P1	300
S1	P2	200
S1	P3	400
S2	P1	300
S2	P2	400
S3	P2	200

جدول محموله (SP)

❖ یکی از مزایای مهم مدل رابطه‌ای سادگی زیاد و درک راحت این ساختار است. همچنین این مدل از پشتوانه تئوری ریاضی و قوی برخوردار می‌باشد.



## تعاریف پایه و نکاتی در باره اصطلاحات کاربردی مدل رابطه ای

10

### ■ مفهوم میدان (Domain) :

مجموعه‌ای است که مقادیر یک صفت خاصه از آن بر گرفته می‌شود. مثلاً میدان نام تهیه کنندگان و شهرها به صورت زیر است:

$D_{\text{name}} = \{\text{آلومین ، پولادین ، ایران قطعه ، فن‌آوران}\}$

$D_{\text{city}} = \{\text{شیراز، تهران، تبریز}\}$

### ■ تعریف رابطه:

رابطه زیر مجموعه‌ای از ضرب دکارتی چند دامنه.

$$\{1,2,3\} \times \{4,5\} = \{(1,4), (1,5), (2,4), (2,5), (3,4), (3,5)\}$$

مثال ۱ :

$R = \{(1,5), (2,4), (3,4)\}$  یک رابطه است:

نکته: هرگز رابطه را با ارتباط یکی نگیرید. ارتباط (*Relationship*) به معنی ارتباط ۲ موجودیت است که در بخش قبل دیدیم. در حالیکه رابطه مجموعه‌ای از تاپل‌هاست

### ■ تاپل (Tuple):

به عضو  $(3,4)$  از رابطه  $R$  یک تاپل گویند. پس تاپل به اعضاء رابطه گفته می‌شود.

به عبارتی دیگر تاپل مجموعه‌ای است از مقادیر صفات خاصه.



## نمایش رابطه با استفاده از جدول

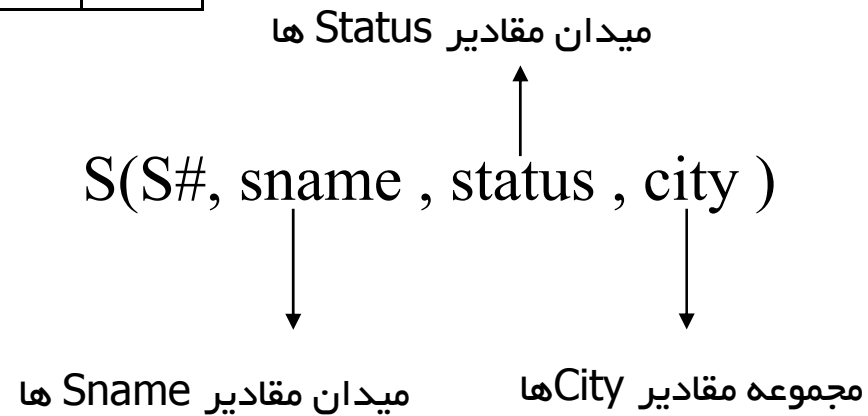
11

1	5
2	4
3	4

$$R = \{(1,5), (2,4), (3,4)\}$$

مثال ۱: رابطه R را می‌توان بصورت جدول زیر نمایش داد:

مثال ۲: رابطه تهیه کنندگان را در نظر می‌گیریم:



یک تاپل (Tuple)

S(S#	Sname	Status	City)
S1	Sn1	10	C1
S2	sn2	20	C2
S3	sn3	15	C2
S4	sn4	14	C3



## جداول نمونه مربوطه به سیستم بانک اطلاعاتی تولید کننده و محصول

12

S#	P#	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

جدول (SP)

تعداد تولید هر محصول توسط تولیدکننده

S#	Sname	Status	City
S1	Sn1	20	C2
S2	Sn2	10	C3
S3	Sn3	30	C2
S4	Sn4	20	C2
S5	Sn5	30	C1

جدول (S)  
تولیدکنندگان

P#	PName	Color	Wight	City
P1	Nut	Red	12	C2
P2	Bolt	Green	17	C3
P3	Screw	Blue	17	C4
P4	Screw	Red	14	C2
P5	Cam	Blue	12	C3
P6	Coy	red	19	c2

جدول (P)  
محصولات



## تعریف مفاهیم رابطه

13

با فرض داشتن  $n$  میدان  $D_1, D_2, \dots, D_n$  رابطه از دو مجموعه تشکیل شده است :

مجموعه عنوان : Heading یا سررابطه  
مجموعه بدنه : Body یا پیکر رابطه

مجموعه اسامی صفات خاصه است. (Attribute names)

این صفات خاصه از میدانهایی که صفات خاصه روی آنها تعریف می شود ، مقدار می گیرد.  
هر رابطه دارای یک نام است و با داشتن نام و مجموعه عنوان رابطه **جوهر رابطه** معلوم است.

نام رابطه

$R (A_1 , A_2 , \dots , A_n)$

مجموعه عنوان رابطه

ذات یا جوهر رابطه که ثابت در زمان است .



## تعریف مفاهیم رابطه

14

**درجه رابطه:** کاردینالیتی مجموعه عنوان ، یعنی تعداد صفات خاصه عنوان رابطه (تعداد ستونها)

مثال :

اصطلاح	درجه
رابطه Unary	۱
رابطه Binary	۲
رابطه Ternary	۳
رابطه n-ary	.

■ S از درجه ۴ است.

■ SP از درجه ۳ است.

■ P از درجه ۵ است.

تعداد ستونهای جدول همان درجه رابطه است.

## کاردینالیتی رابطه:

تعداد تاپل‌های رابطه در یک لحظه از حیات آن، کاردینالیتی رابطه نام دارد و در طول حیات رابطه متغیر است.

S#	Sname	City
S1	فن‌آوران	تهران
S1	فن‌آوران	تبریز
S1	فن‌آوران	شیراز
S2	فن‌آوران	تهران
S2	فن‌آوران	تبریز
.	.	.
.	.	.



مفهوم رابطه ای	مفهوم جدولی
رابطه	جدول
مجموعه عنوان	عنوان جدول
تاپل	سطر
صفت خاصه	ستون
درجه	تعداد ستونها
کاردینالیتی	تعداد سطرها
میدان	مجموعه مقادیر ستون
کلید	(خود جدول چیزی به نام کلید ندارد؟)



## ۱- در رابطه تاپل تکراری وجود ندارد.

زیرا پیکر رابطه یک مجموعه است و مجموعه در ریاضیات طبق تعریف عناصر تکراری ندارد. به همین ترتیب در بانک جدولی نیز رکوردهای تکراری نداریم.

## ۲- تاپل‌ها در رابطه نظم ندارند.

این خصوصیت نیز از مجموعه بودن پیکر رابطه نتیجه می‌شود. به همین ترتیب در بانک اطلاعاتی جدولی نیز ترتیب رکوردها در جدول مهم نیست. هر چند که معمولاً سطرها با یک نظم خاص نشان داده می‌شوند.

## ۳- صفات خاصه نظم ندارند.

این خاصیت نیز از مجموعه بودن عنوان رابطه نتیجه می‌شود. به همین ترتیب فیلدهای یک جدول نیز نظم ندارند و می‌توان آنها را جابجا کرد. هنگام تعریف یک جدول مهم نیست ترتیب فیلدها چگونه باشد.





## خصوصیات رابطه

17

۴- همه مقادیر صفات خاصه تجزیه ناپذیرند.

به عبارتی دیگر در رابطه، یک تاپل نمی‌تواند حاوی تاپل‌های دیگری باشد. مثلاً مجموعه  $R$  زیر، رابطه نیست، چون  $(1, (4, 5))$  قابل قبول نمی‌باشد:

$$R = \{(1, (2, 3)), (1, (4, 5)), \dots\}$$

به همین ترتیب در بانک اطلاعاتی رابطه‌ای نمی‌توان مثل زیر تعریف کرد که فیلدی از آن مرکب بوده و از فیلدهای ساده‌تری تشکیل شده باشد. به عبارت دیگر از تقاطع هر سطر و ستون باید یک مقدار بدست آید.

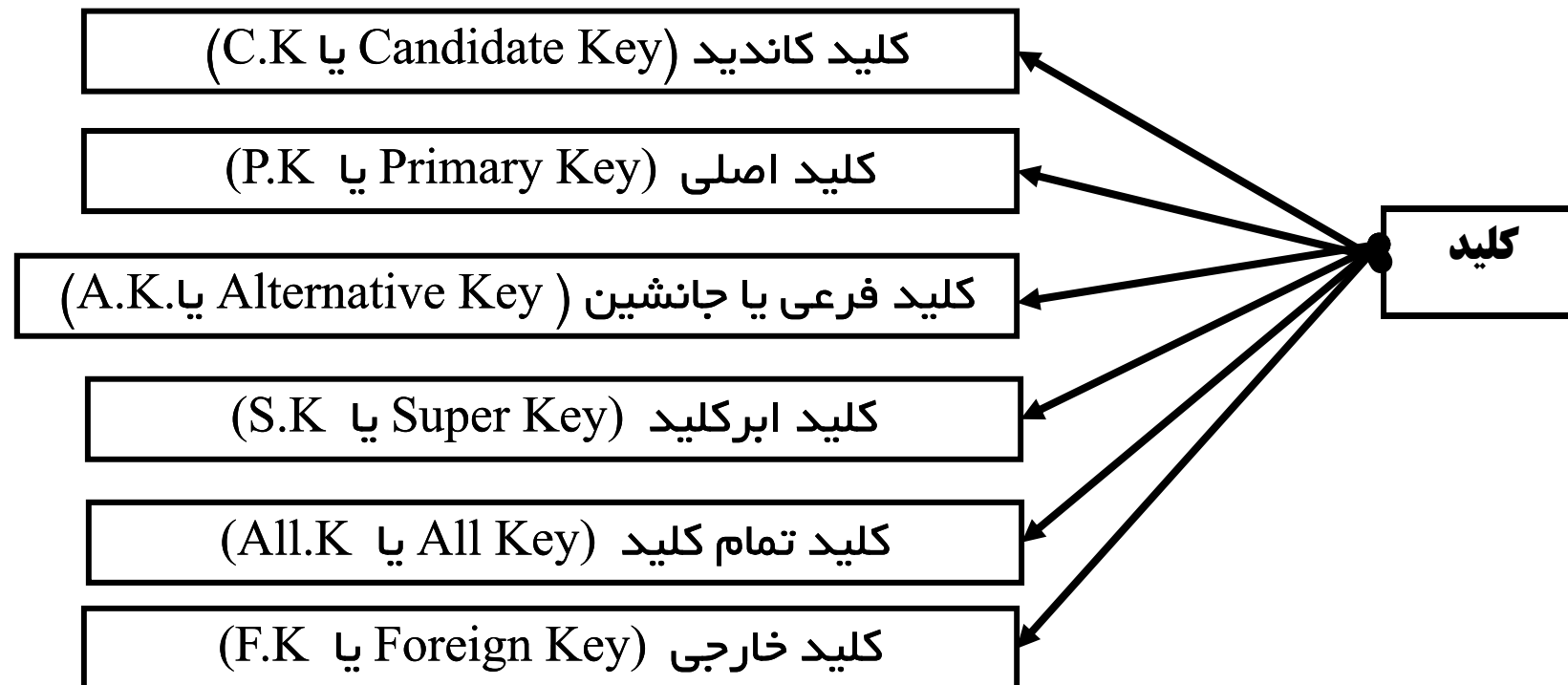
تاریخ			نام		معدل
سال	روز	ماه	فامیلی	نام کوچک	



# انواع کلید ها



از مهمترین ابزارها در حفظ جامعیت بانک اطلاعاتی کلیدها Keys هستند.





# انواع کلید ها

19

## ۱- تعریف کلید کاندید (Candidate Key) :

هر زیر مجموعه از مجموعه عنوان رابطه به صورت  $A_1 A_2 \dots A_k$  که دارای دوخاصیت زیر باشد کلید کاندید است :

۱- در طول حیات رابطه یکتایی مقدار داشته باشد (Uniqueness) یعنی در هیچ دو Tuple ای از رابطه این ترکیب مقدار یکسان نداشته باشد.

۲- کاهش ناپذیری Irreducibility یا Minimality (کمینگی) داشته باشد. یعنی اگر هر یک از اجزای تشکیل دهنده این ترکیب را ازش بگیریم خاصیت یکتایی مقدار آن از بین برود.

هر رابطه ای در سیستم بانک اطلاعاتی رابطه ای حتما " حداقل یک کلید کاندید دارد.

نام رابطه	کلید کاندید
S	S#
P	p#
SP	(S#,P#)



## ۲- تعریف کلید اصلی ( Primary Key ) :

کلید کاندیدی است که طراح انتخاب می کند بنابر ملاحظات محیطی (یعنی یکی از کلید های کاندید) مثلا رابطه S دارای کلیدهای کاندید S# و SNAME است ، اما طراح ممکن است S# را به عنوان کلید اصلی در نظر بگیرد. آن کلید کاندیدی که توسط آن Access بیشتری به Data انجام می شود به عنوان کلید اصلی مطرح می شود که بستگی به نظر طراح دارد.

**در هر رابطه (جدول) ، فقط و فقط یک کلید اصلی وجود دارد.**

کلید اصلی :شناسه تاپل است در رابطه ، امکان آدرس دهی به تاپل در رابطه است در محیط انتزاعی (الزاما صرف معرفی کلید اصلی ایجاب نمی کند حتما سیستم روی کلید اصلی Index بزند بلکه باید درخواست کنید)

CREATE TABLE S

(S# CHAR(6) NOT NULL ,

SNAME VARCHAR(50) NOT NULL ,

CITY VARCHAR(30) ,

STATUS NUMBER(3) NOT NULL ,

CONSTRAINT PK\_S PRIMARY KEY (S# );

کلید اصلی حتما در تعریف رابطه باید قید شود

یعنی جزئی از شمای اصلی پایگاه است.



## انواع کلید ها

### ۳- کلید جانشین (Alternate Key) :

در یک رابطه (جدول) امکان وجود بیش از کلیدکاندید وجود دارد که یکی از آنها حتماً "کلید اصلی و باقیمانده کلیدهای کاندید، کلید جانشین هستند. به عبارت دیگر : کلید جانشین = کلید کاندید منهای کلید اصلی .

### ۴- تعریف ابر کلید (Supper Key) :

اگر خاصیت دوم در کلید کاندیدا برقرار نباشد، یعنی زیر مجموعه ای از مجموعه Heading وجود داشته باشد که یکتایی مقدار داشته باشد ولی کاهش ناپذیری نداشته باشد، به آن Supper Key می گویند (یعنی درون خود حد اقل یک کلید کاندید دارد) اما Supper Key، Codd را به عنوان یک مفهوم اساسی در مدل رابطه ای مطرح نمی کند.

آیا وجود حد اقل یک کلید کاندید در رابطه محرز است ؟

بله . هر رابطه کلید کاندید دارد. زیرا در بدترین حالت خود مجموعه عنوان می تواند کلید کاندید باشد ، زیرا مجموعه Body ، Tuple تکراری نمی تواند داشته باشد.

### ۵- تعریف تمام کلید (All-Key) :

به رابطه ای که مجموعه عنوانش کلید کاندیدش باشد رابطه تمام کلید All-Key می گویند.

(سیستم مطلوبی نیست زیرا حداقل Overhead ایجاد Index زیاد می شود)

کلید Identifier تشخیص Tuple ها است و هرچه کلید طولانی تر باشد برای یافتن Tuple باید مسیر پیچیده تری را رفت



## انواع کلید ها

### ۶- تعریف کلید خارجی (Foreign Key) :

Ai در R2 کلید خارجی است اگر در رابطه ای مثلا R1 کلید اصلی باشد .  
می توان گفت Ai در R2 کلید خارجی است اگر از یک میدان اصلی مقدار بگیرد.

مثال : صفت خاصه S# در SP کلید خارجی است زیرا S# در S کلید اصلی است.  
مثال : P# در SP کلید خارجی است زیرا P# در P کلید اصلی است .

نکته : در تعریف کلید خارجی R1 و R2 لزوما مجزا نیستند

مثال :

EMP ( EMP# , EMPNAME ,.... , EMPMGR#)

E1 ,      En1 , ..... , E13

E2 ,      En2 , ..... , E27

E3 ,      En3 , ..... , E13

EMP# کلید اصلی است . EMGR # در همین رابطه کلید خارجی است .



**فایده کلید خارجی چیست ؟** امکانی است برای نشان دادن ارتباط بین موجودیتها . مثل اینکه در رابطه SP از طریق کلید

خارجی S# و P# ارتباط دو موجودیت S و P را نشان می دهیم. و نیز امکان ارجاع به موجودیتهاست . به عنوان مثال

وجود S1 در رابطه SP ارجاعی است به نمونه موجودیت S1 در رابطه S.

**آیا تنها امکان نشان دادن ارتباط بین موجودیتها کلید خارجی است ؟**

خیر. در حالت کلی وجود هر صفت خاصه مشترک در مجموعه عنوان دو رابطه نشاندهنده نوعی ارتباط بین دو نوع

موجودیتی است که آن دو رابطه نمایشگر آنها هستند.

مثال :

S(S#,SNAME, . . . ,CITY)

P(P# ,PNAME,COLOR,WEIGHT ,CITY)

J(J#,JNAME,CITY)

SPJ(S#,P#,J#, CITY)

CITY در دو رابطه نوعی ارتباط را نشان می دهد مثلا "در یک شهر بودن" "در یک شهر نبودن" و CITY در هیچیک از دو

رابطه کلید خارجی نیست. زیرا هیچ جا کلید اصلی نیست.

کلید خارجی بیانگر یک ارتباط محوری و اساسی می باشد اما صفت مشترک چنین نیست.

CITY بیانگر نوعی ارتباط است ولی J# و P# و S# بیانگر یک ارتباط محوری و اساسی می باشد.



## مثالی از کلید ها

شماره دانشجویی	ش پرونده	نام	فامیلی	ش.ش	نام پدر	سال تولد
۲	۲۰۴۱	علی	محمدی	۱۱	محمد	۱۳۶۳
۵	۱۲۷۵	محمد	ناصری	۱۹	رضا	۱۳۶۲
۱۴	۱۱۰۲	علی	محمدی	۴۵۶	احمد	۱۳۶۵
۶	۱۵۹۰	حمید	کاظمی	۸۶۷	حامد	۱۳۶۶
۳	۱۵۶۱	رضا	رضایی	۴۵۶	احمد	۱۳۶۳

جدول  
student

ش.دانشجویی و  
کد درس در  
جدول term  
کدام به تنهایی  
کلید خارجی  
هستند.

کد ترم	ش دانشجویی	کد درس	نمره
۸۲۱	۱۴	۴	۱۲
۸۲۱	۵	۲	۱۷
۸۲۱	۱۴	۲	۱۶
۸۲۱	۵	۴	۹
۸۲۱	۲	۱	۱۶
۸۲۱	۱۴	۶	۱۴
۸۲۲	۶	۶	۱۳
۸۲۲	۵	۴	۱۱
۸۲۲	۳	۱	۱۰
۸۲۲	۵	۳	۱۹
۸۲۲	۳	۲	۱۸

جدول term

ش.دانشجویی و  
کد درس و  
کدترم در جدول  
term با یکدیگر  
کلید کاندید و  
کلید اصلی  
هستند.

در جدول بالا "شماره دانشجویی" و "ش پرونده" هر کدام به تنهایی کلید کاندید و ش دانشجویی به تنهایی کلید اصلی است.  
در جدول بالا "شماره دانشجویی" به عنوان کلید اصلی انتخاب شده است.

کد درس	نام درس	-----	-----
۱	فیزیک	---	---
۲	مبانی کامپیوتر		
۳	ریاضی ۱		
۴	معادلات		
۶	شیمی		

جدول  
dars





## مثالی از کلید ها

جدول S

S#	S.name	status	City
S1	Sn1	20	C2
S2	Sn2	10	C3
S3	Sn3	30	C2
S4	Sn4	20	C2
S5	sn5	30	C1

جدول SP

S#	P#	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

P#	P.Name	Color	Wight	City
P1	Nut	Red	12	C2
P2	Bolt	Green	17	C3
P3	Screw	Blue	17	C4
P4	Screw	Red	14	C2
P5	Cam	Blue	12	C3
P6	coy	red	19	c2

جدول P

در جدول S، کلید اصلی S# است و در جدول P کلید اصلی P# است و در جدول SP، فیلدهای S#+P# "توأم" کلید اصلی هستند  
 جدول SP دارای ۲ کلید خارجی است :  
 • فیلد S# در جدول SP به عنوان کلید خارجی است  
 • فیلد P# در جدول SP به عنوان کلید خارجی است



## قواعد جامعیت (INTEGRITY RULES)

مجموعه قواعدی که به کمک آنها سیستم صحت و دقت داده های ذخیره شده در پایگاه و نیز ارتباطات بین انواع موجودیتها را کنترل می کند.

### ۱- قواعد جامعیت کاربری (UDR=User Defined Rules):

تعریف شده توسط کاربر (منظور شخص DBA) است. کاربران خارجی باید درخواست خود را به DBA بدهند.

■ میدانی : قاعده ای است ناظر به مقادیر یک میدان

■ ستونی : برای صفات خاصه بخصوصی در رابطه بخصوصی محدودیتی قایل بشویم

#### قواعد خاص

### ۲- قواعد جامعیت عام (Meta-Rule) :

مستقل از Data های خاص مطرحند ، Independent از Data های محیط خاص

■ C1 : قاعده جامعیت موجودیتی (Entity Integrity Rule)

ناظر به کلید اصلی است

■ C2 : قاعده جامعیت ارجاعی (Refrential Integrity Rule)

ناظر به کلید خارجی است

#### قواعد عام

#### قواعد جامعیت



## قواعد جامعیت (INTEGRITY RULES)

27

مثال میدانی : شماره تهیه کنندگان حتما به این صورت است :

$R1 = sdddd$

$R2 = 1111 < dddd < 8888$

$R3 = \text{City In } \{ C1, C2, C3, C4, C5 \}$

ستونی : علاوه بر قواعد میدانی که بر روی مجموعه مقادیر میدان ناظر است ، قواعد جامعیت ستونی اخص از محدودیت میدانی است ، یعنی برای صفات خاصه بخصوصی در رابطه بخصوصی محدودیتی قایل بشویم .

$R4 = 0 < QTY < 100$

مثال :

$R5 =$

$QTY$  برای تهیه کنندگان ساکن  $C1$  و  $C2$  که قطعات  $P1$  و  $P2$  را تهیه می کنند نمی تواند بیشتر از ۵۰ باشد

■ یک DBMS واقعی باید امکان بدهد تا DBA انواع قواعد جامعیت را بیان و ایجاد کند و سیستم باید این قواعد را پیاده و عمل کند

■ شمای ادراکی فقط از تعریف رابطه تشکیل نمی شود بلکه قواعد جامعیت مربوط به آن رابطه جزئی از شمای ادراکی است.

■ بخش عمده Overhead عملیات DBMS مربوط به این قواعد جامعیت است .



## قواعد جامعیت (INTEGRITY RULES)

28

مثال:

قاعده ۱: شماره دانشجویی به صورت dddddd است که در آن دو رقم اول عددی بزرگتر از ۶۵ است.

قاعده ۲: مقادیر میدان STDEGR: 'bs', 'ms' و 'doc' است.

قاعده ۳: دانشجویی با معدل کمتر از ۱۲ در یک ترم، نمی‌تواند در ترم بعد بیش از ۱۴ واحد انتخاب کند.

قاعده ۴: مدیر گروه آموزشی حداکثر می‌تواند دو دوره دو ساله، مدیر گروه باشد.

قاعده ۵: هر درس حتماً باید یک منبع اصلی و دو منبع فرعی داشته باشد.

قاعده ۶: رعایت پیش نیاز (ها) در انتخاب درس، الزامی است.

قاعده ۷: در میدان مقادیر GRADE، نمره «ناتمام» وجود ندارد.

قاعده ۸: دانشجوی دوره کارشناسی باید ۱۴۲ واحد بگذراند تا فارغ التحصیل شود.

قاعده ۹: دانشجو نمی‌تواند درس آزمایشگاه را حذف کند.

قاعده ۱۰: تعداد دانشجویان درس‌های تخصصی در هر گروه نباید بیش از ۳۰ نفر باشد.

قاعده ۱۱: حداقل نمره قبولی برای دانشجوی دوره کارشناسی ارشد ۱۲ است.

قاعده ۱۲: تعداد واحدهای اخذ شده دانشجوی فعال نمی‌تواند «هیچمقدار» باشد.



## قواعد جامعیت (INTEGRITY RULES)

29

■ C1 : قاعده جامعیت موجودیتی Entity Integrity Rule ناظر به کلید اصلی است.

■ C2 : قاعده جامعیت ارجاعی Refrential Integrity Rule ناظر به کلید خارجی است.

قواعد جامعیت عام

قاعده C1 : می گوید :

هیچ جز تشکیل دهنده کلید اصلی نمی تواند Null باشد. ( هیچ مقدار Null Value داشته باشد )

■ مقدار ناشناخته ( Unknown ) : مثلا STATUS تهیه کننده ای مشخص نباشد .

■ مقدار غیر قابل اعمال (Inapplicable) : مثلا شماره شناسنامه همسر برای شخص مجرد غیر قابل اعمال است

Null Value

Null Value با صفر یا Blank فرق می کند زیرا آنها مقدارند . هر چند سیستم باید امکانی برای بازنمایی Null-Value داشته باشد. به عنوان مثال در مقایسه دو null value چه می توان گفت ؟

Null ( < = > ) Null

لحظه به لحظه Database باید صحیح باشد.



### دلیل قاعده C1:

زیرا کلید اصلی شناسه Tuple است. شناسه یک نمونه مشخص از یک نمونه موجودیت به معنای عام و متمایز از هر نمونه دیگر، و کلید اصلی عامل تمیز یک نمونه با نمونه دیگر است و عامل تمیز خود نمی تواند ناشناخته باشد.

باید در تعریف رابطه ، Null-Value پذیری یا ناپذیری صفت خاصه را بیان کرد (DBMS باید این امکان را بدهد)

```
CREATE TABLE S
```

```
(S# CHAR(6) NOT NULL ,
```

```
SNAME VARCHAR(50) NOT NULL ,
```

```
CITY VARCHAR(30) ,
```

```
STATUS NUMBER(3) NOT NULL ,
```

```
CONSTRAINT PK_S PRIMARY KEY (S# );
```

یعنی در طول حیات این رابطه  
Sname نمی تواند Null باشد  
و باید مقدار مشخصی داشته باشد.



## قواعد جامعیت (INTEGRITY RULES)

31

**C2 : قاعده جامعیت ارجاعی (Refrential Integrity Rule) ناظر به کلید خارجی است.**

اگر  $A_i$  در  $R_2$  کلید خارجی باشد (یعنی  $A_i$  در رابطه  $R_1$  کلید اصلی است) در این صورت  $R_2.A_i$  :

۱- می تواند null باشد اگر جزء تشکیل دهنده کلید اصلی در  $R_2$  نباشد در غیر اینصورت ،

۲- باید حتما مقداری داشته باشد که در  $R_1$  وجود دارد.

**رعایت قانون جامعیت ارجاعی باعث :**

۱-عدم نقض روابط کلید خارجی (اگر در رکوردهای جدول پدر تغییر بدهیم در جدوال فرزند نیز تغییر اعمال شود)

۲-عدم وجود رکورد یتیم ( رکورد یتیم (Orphan) یعنی یک رکوردی که در جدول فرزند وجود دارد ولی در جدول پدر وجود ندارد )

ترجیحا "قانون جامعیت ارجاعی توسط DBMS اعمال میشود چون حتی DBA امکان نقض آن را بر اثر فراموشی یا اشتباه دارد.

$R_1: S(S\#, SNAME, CITY, STATUS)$  \_\_\_\_\_  $R_2: Sp (S\#, P\#, QTY)$

S1	S1	P1
S2	S1	P2
S3	S2	P1
	S3	P2
	S3	P1
	S4	P1

در S وجود ندارد . (Unmatchable مقدار)



- ❑ CREATE TABLE SP
- ❑ (snum character(3) ,
- ❑ pnum character(3) ,
- ❑ qty integer ,
- ❑ CONSTRAINT pk\_snumpnum PRIMARY KEY (snum,pnum),
- ❑ CONSTRAINT fk\_snum
- ❑ FOREIGN KEY (snum)
- ❑ REFERENCES s(snum)
- ❑ ON DELETE CASCADE ,
- ❑ CONSTRAINT fk\_pnum
- ❑ FOREIGN KEY (pnum)
- ❑ REFERENCES p(pnum)
- ❑ ON DELETE CASCADE)





## مکانیسم های رعایت قاعده C2

33

چگونه قاعده C2 را رعایت کنیم :

سه مکانیسم برای رعایت C2 متصور است :

نکته : قاعده C2 باید در جمیع عملیات روی پایگاه رعایت شود.

مثال : فرض می کنیم در رابطه S تهیه کننده 'S7' وجود ندارد.

```
INSERT INTO SP
```

```
VALUES (S7,P1,60) ;
```

قاعده C2 می گوید این عمل باید Reject شود

درج یک تاپل

مثال : فرض می کنیم در رابطه S موجودیت S3 را داریم :

```
DELETE FROM S
```

```
WHERE S#='S3' ;
```

آیا باید اجرا شود؟ اگر بله با تاپلهای Referencing چه باید کرد؟؟ و اگر نه ...



## مکانیسم های رعایت قاعده C2

34

در عمل Delete سه مکانیسم برای رعایت قاعده C2 وجود دارد:

۱- روش آبشاری (Cascade): انتشار عملیات به رابطه Referencing وقتی که عمل در رابطه Referenced انجام شود.

مثال: با حذف تاپلهای S3 از S، تمام تاپلهای رجوع کننده به S3 در رابطه های دیگر نیز باید حذف گردد.

1- DELETE FROM S WHERE S#='S3' ;

2- DELETE FROM SP WHERE S#='S3';

این روش نوعی Propagating Update است. این روش در بعضی از سیستمهای موجود وجود دارد.

۲- روش Restricted (انجام عمل بطور مشروط): در این روش تاپل مرجع (Referenced) حذف نمی شود و عبارتی دیگر

تا زمانی که تاپل مرجع تاپل رجوع کننده ای دارد، عمل حذف انجام نمی شود.

۳- روش هیچمقدارگذاری (Nullifying): در این روش با حذف تاپل مرجع مقدار کلید خارجی در تاپل رجوع کننده Null

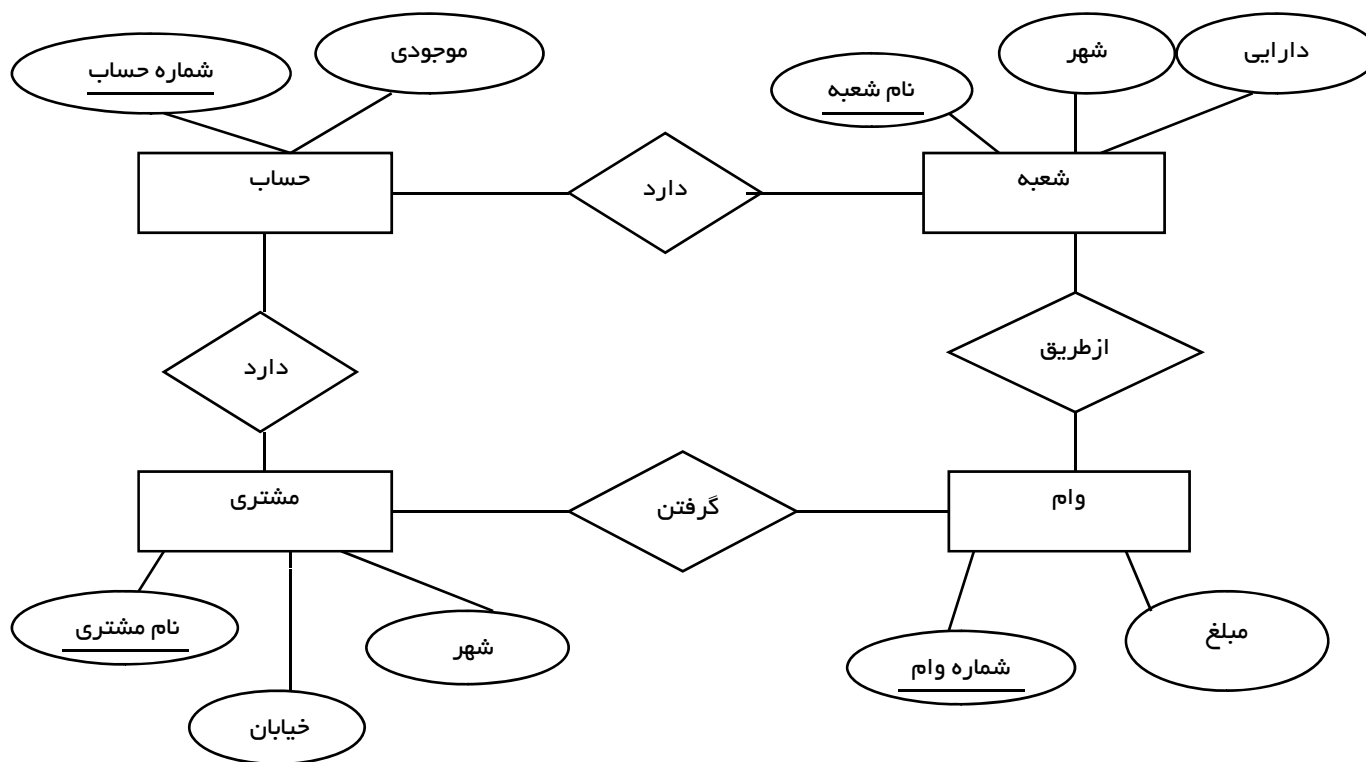
می شود و زمانی امکانپذیر است که کلید خارجی جزء کلید اصلی نباشد.



## تبدیل نمودار E-R به جداول مدل رابطه ای

35

تا اینجا شما طراحی لایه اول بانک اطلاعات یعنی تصویر ادراکی عام را آموخته اید. در این قسمت می خواهیم تبدیل مدل عام به مدل منطقی را در قالب جدول ببینیم. در این بخش می خواهیم روشی مطرح کنیم که توسط آن نمودار E/R به جداول مدل رابطه ای تبدیل کنیم. در شکل زیر کل بانک اطلاعات یک خصوصی را می بینید. می خواهیم این بانک را پیاده سازی کنیم :





## تبدیل نمودار E-R به جداول مدل رابطه ای

36

**پیاده سازی موجودیت ها:** هر موجودیت باتمام صفات آن یک جدول راتشکیل میدهد.کلید این جدول کلید همان موجودیت است.

**نکته:** برای پیاده سازی موجودیت ضعیف ، کلیدپدرش را باید به عنوان یکی از فیلدهای جدول در نظر بگیریم وکلید این جدول تلفیقی از کلیدپدر و صفت های موجودیت ضعیف است.

**پیاده سازی ارتباط :** ارتباط از ۲ طریق قابل پیاده سازی است : ۱ - کلید خارجی ۲ - جدول ارتباط (Junction Table)  
اگر ارتباط یک به چند باشد، از طریق قراردادن کلید اصلی موجودیت یک در جدول موجودیت چند به عنوان کلید خارجی ، ارتباط را پیاده سازی میکنیم.

اگر ارتباط چند به چند باشد، از جدولی موسوم به جداول ارتباط (Junction Table) استفاده میکنیم. بدین ترتیب که کلید هر موجودیت را در مقابل کلید موجودیت مرتبط با آن قرار میدهیم.

**نکته:** صفات یک رابطه نیز در یک جدول ارتباط قرار می گیرند.



## تبدیل نمودار E-R به جداول مدل رابطه ای

37

