به نام آنکه جان را فکرت آموخت

بخش ششم:
مفاهیم اساسی مدل داده رابطه‌ای
(گفتار دهم کتاب آقای دکتر روحانی)

دکتر عیسی زارع پور

دانشکده مهندسی کامپیوتر، دانشگاه علم و صنعت

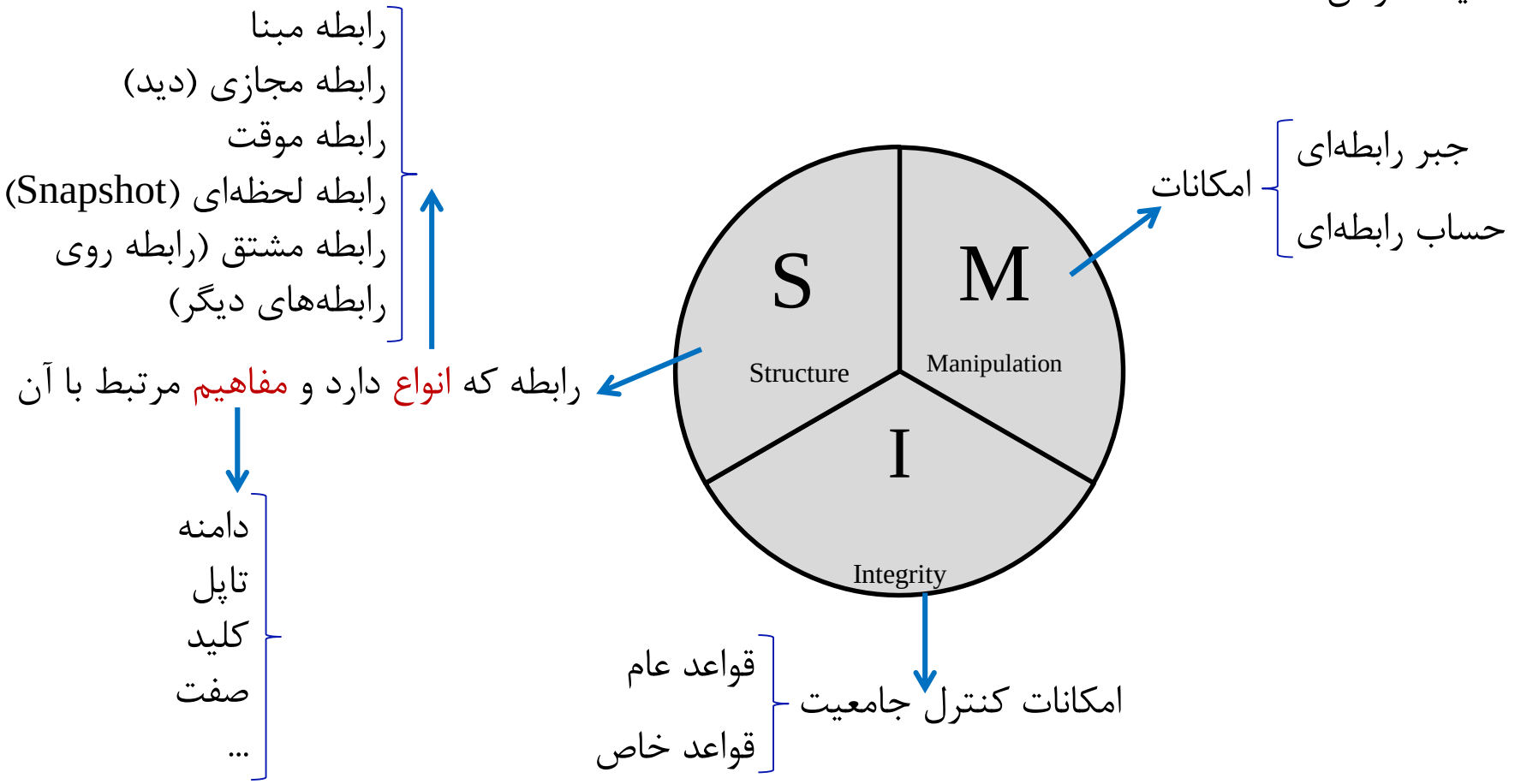
سال تحصیلی ۹۸-۹۷

محتویات اسلایدها برگرفته از یادداشت‌های کلاسی **استاد محمدتقی روحانی رانکوهی** است. اسلایدها توسط آقای **دکتر مرتضی امینی**(دانشکده مهندسی کامپیوتر دانشگاه صنعتی شریف) تهیه شده است.

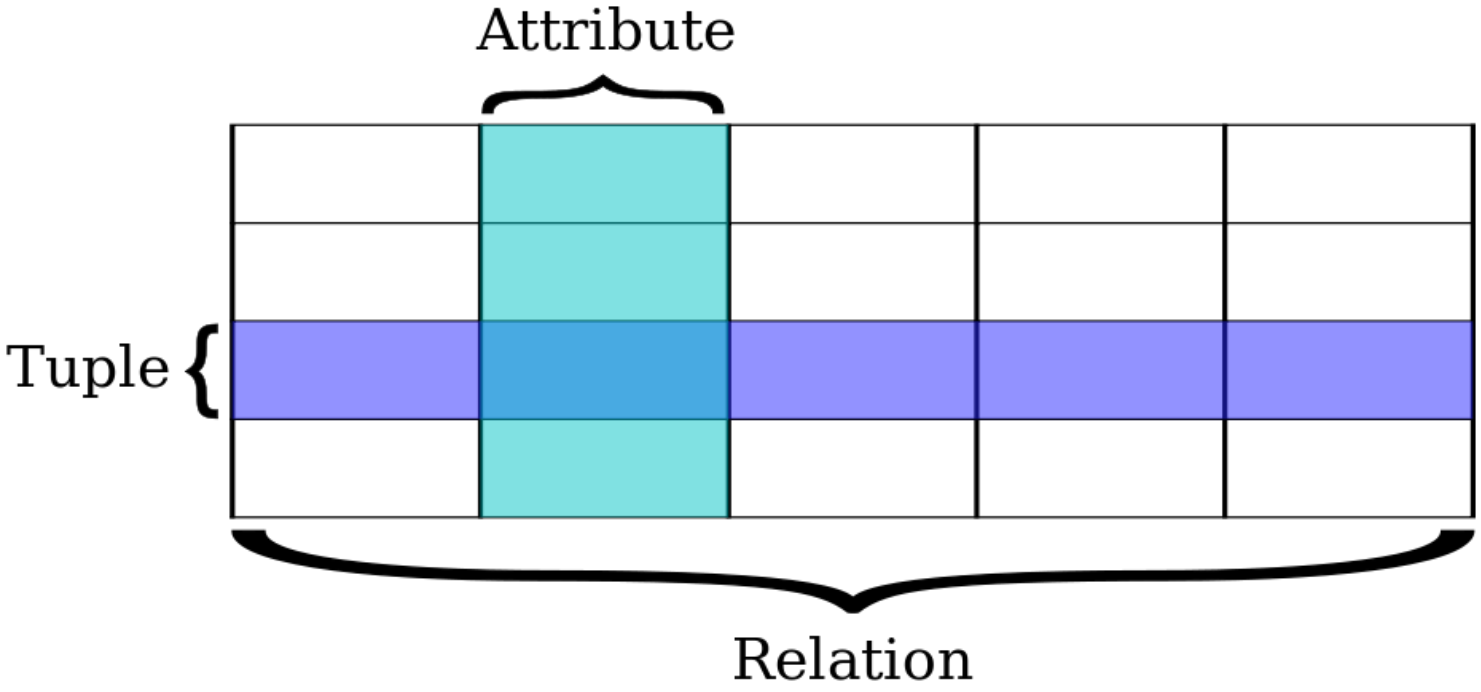
- RDM مبنای تئوریک RDB و RDBMS
- واضع مدل: F. Codd
- مفاهیم زیر در طی سه بخش باقیمانده از این درس مرور می‌شوند:
  - رابطه (Relation)
  - دامنه (میدان)
  - رابطه نرمال و غیرنرمال
  - کلید در مدل رابطه‌ای
  - قواعد جامعیت رابطه‌ای
  - عملیات در RDB ← جبر رابطه‌ای، حساب رابطه‌ای
  - طراحی RDB ← روش بالا به پایین، روش نرمال‌ترسازی (سنتز)

مدل داده‌ای

✓ **مدل داده** مجموعه‌ای است از امکانات برای طراحی منطقی و تعریف پایگاه داده‌ها، کنترل آن و نیز انجام عملیات در آن.



مدل داده رابطه ای



رابطه

در ریاضی: هر زیر مجموعه از ضرب کارتزین چند مجموعه

(۱) با فرض وجود m مجموعه از مقادیر موسوم به دامنه [میدان] D_1، ...، D_m:

رابطه R با صفات A_1، ...، A_m تعریف شده روی این m دامنه

مجموعه‌ای است از عناصر، هر یک به صورت $\langle d_{1i}, d_{2i}, \ldots, d_{mi} \rangle$ موسوم به m-تاپل (m-tuple)

به نحوی که $d_{1i} \in D_1$، ... ، $d_{ji} \in D_j$ (تاپل i ام)

مثال

STUD (STID, STNAME, STJ, STL, STD)

777	st7	bs	phys	d11
⋮	⋮	⋮	⋮	⋮
444	st4	bs	comp	d14

یک تاپل ۵-تایی

رابطه (ادامه)

(۲) [Date] با فرض وجود m مجموعه از مقادیر موسوم به دامنه [میدان] D_1، ... ،D_m نه لزوماً متمایز،

رابطه R تعریف شده روی این m دامنه:

دومجموعه
- عنوان [سرآیند] (Heading): مجموعه‌ای است نامدار از اسامی صفات یعنی $\{A_1,\ldots,A_m\}$ که با $R(A_1,\ldots,A_m)$ نمایش داده می‌شود.
- بدنه [پیکر] (Body): مجموعه ای است از تاپل‌ها که هر تاپل یک سه‌تایی ⟨دامنه، صفت، مقدار⟩ است: $\{\langle D_1: A_1:V_1\rangle, \langle D_2: A_2:V_2\rangle, \ldots\}$:TUPLES

مثال رابطه دانشجو

STUD (STID, STNAME, STJ, STL, STD)

- **درجه رابطه:** کاردینالیتی عنوان یا تعداد صفات رابطه

m	اصطلاح
۱	رابطه یگانی
۲	رابطه دوگانی
n	رابطه nگانی

رابطه (ادامه)

- مجموعه عنوان را با H_R یا R(H) نیز نمایش می‌دهیم. به R(H)، Intention (ذات، جوهر یا **چکیده**) رابطه هم گفته می‌شود.
  - R(H) ثابت در زمان است. یعنی اگر مجموعه صفات را عوض کنیم، از نظر ریاضی یک رابطه دیگر است.
  - همین R(H) برای تعریف رابطه در سیستم کافی است.

CREATE RELATEION STUD
(STID, STNAME, STJ, STL, STD)

- هررابطه یک معنا دارد، بیانگر واقعیتی از یک محیط مشخص. به عنوان مثال وقتی می‌گوییم رابطه STUD را داریم، معنایش این است که در خردجهان واقع، نوع موجودیتی با نام STUD و با صفات STID و STNAME و ... و STD وجود دارد.

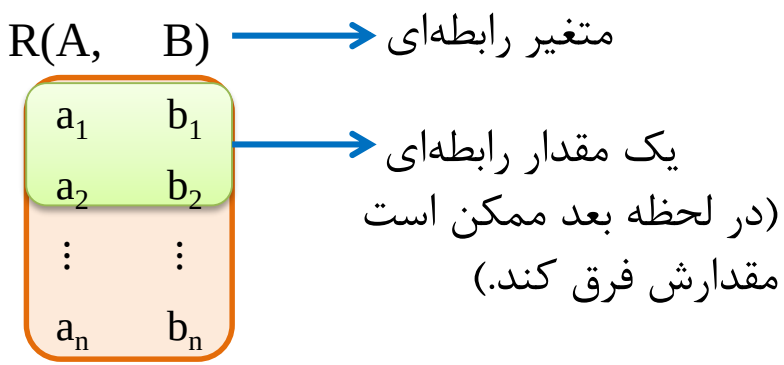
رابطه (ادامه)

- **کاردینالیتی رابطه:** همان کاردینالیتی بدنه؛ تعداد تاپل‌ها (بزرگتر مساوی صفر؛ صفر در بدو تعریف)
  - بدنه رابطه، متغیر در زمان است.
  - به یک مقدار بدنه در یک لحظه مشخص instance گویند.
  - به بدنه رابطه، Extension (**بسط** یا گسترده) یا حالت رابطه گویند.

رابطه (ادامه)

تعریف (۳) [از نظر تئوری زبان‌های برنامه‌سازی] تشکیل شده است از یک متغیر رابطه‌ای و در هر لحظه از یک مقدار رابطه‌ای.

- R(H): متغیر رابطه‌ای، متغیری از جنس رابطه Relation Variable [RELVAR]
- بدنه (r): مقدار رابطه‌ای Relation Value [RELVAL]



مدل رابطه‌ای و مدل جدولی

تناظر بین مفاهیم رابطه‌ای و اصطلاحات جدولی

مفهوم رابطه‌ای	اصطلاح
رابطه	جدول (صرفاً امکانی است برای نمایش مفهوم رابطه‌ای و تفاوت‌های متعددی با رابطه دارد.)
تاپل	سطر
صفت	ستون
دامنه	مقادیر مجاز ستون
درجه	تعداد ستون‌ها
کاردینالیتی	تعداد سطرها
کلید	؟ (به معنایی که در مدل رابطه‌ای داریم، در بحث‌های جدولی مطرح نیست.)

مدل رابطه‌ای و مدل جدولی (ادامه)

ویژگی‌های رابطه:

۱- صفات در عنوان رابطه نظم (مکانی) ندارند. [چون مجموعه است] $R(A, B) = R(B, A)$

در حالی که در جدول، ستون‌ها می‌توانند نظم مکانی داشته باشند.

در مدل رابطه‌ای، تنها راه ارجاع به صفت رابطه، نام صفت است.

۲- تاپل‌ها [در بدنه] نظم ندارند (مرتب نیستند) [چون مجموعه است].

۳- رابطه، تاپل تکراری ندارد [چون مجموعه است].

۴- تمام صفات رابطه (نرمال)، تک مقدار هستند [رجوع شود به مفهوم رابطه نرمال] (این ویژگی دلیل تکنیکی دارد و از ذات رابطه نتیجه نمی‌شود). یعنی در هر تاپل دقیقاً یک مقدار برای هر صفت وجود دارد.

- در RM هیچ یک از مفاهیم فایلینگ مطرح نیستند (مثل نظم، فیلد، رکورد، اشاره‌گر، آدرس که در سطح طراحی و فایلینگ فیزیکی مطرح است).

مدل رابطه‌ای و مدل جدولی (ادامه)

تفاوت‌های مفهوم رابطه و اصطلاح جدول

- علاوه بر سه ویژگی اول رابطه که ۳ تفاوت با مفهوم جدول نیز هست:

۴- در رابطه m>=0 (درجه)، یعنی از نظر تئوری رابطه می‌تواند از نظر درجه، صفر باشد.

۵- رابطه می‌تواند بیش از دو بُعد داشته باشد (مثلا Data Cube).

۶- نمایش دقیق عنوان رابطه به صورت زیر است حال آنکه عنوان جدول چنین نیست.

عنوان رابطه مجموعه‌ای است از دوتایی‌های ⟨دامنه:صفت⟩ $R(H): \{\langle D_1: A_1\rangle, \langle D_2: A_2\rangle, \ldots\}$

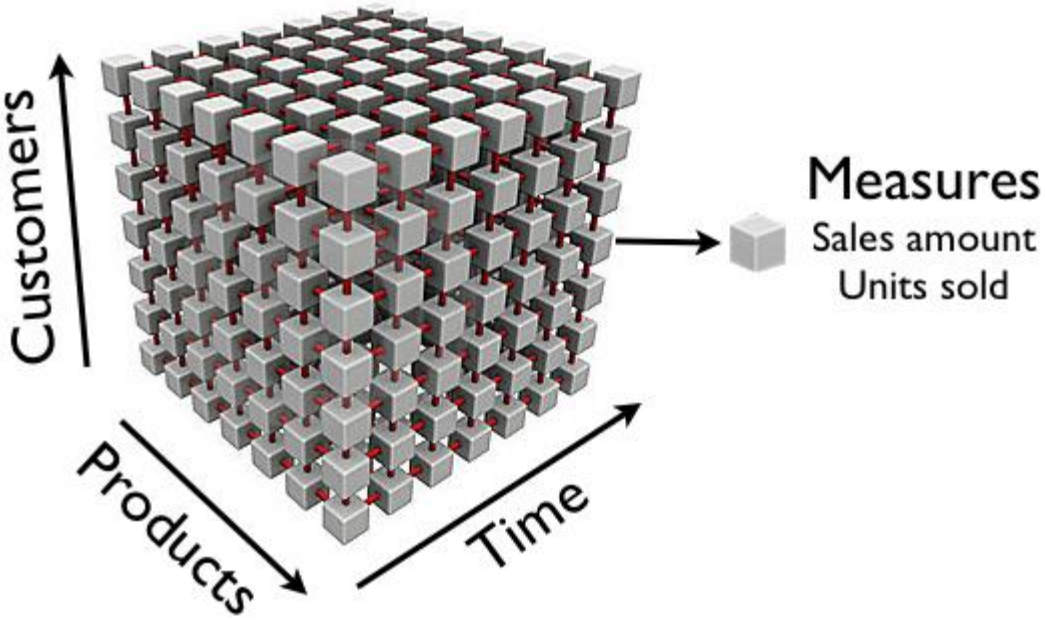
۷- نمایش دقیق تاپل رابطه به صورت زیر است حال آنکه سطر در جدول چنین نیست.

تاپل مجموعه‌ای است از سه‌تایی‌های ⟨دامنه، صفت، مقدار⟩ $TUPLE: \{\langle D_1: A_1: V_1\rangle, \langle D_2: A_2: V_2\rangle, \ldots\}$

۸- رابطه نمی‌تواند هیچ‌مقدار داشته باشد، ولی جدول می‌تواند (البته در این خصوص اختلاف نظر وجود دارد).

داده های چند بعدی

- گاهی اوقات نیاز است که به همراه خود داده، برخی از نقاط مرتبط و جالب هم ذخیره شود و یا ابعاد مختلف یک پدیده ذخیره شود. در این شرایط از داده های چند بعدی استفاده می شود. مثال زیر یک datacube یا ساختار داده سه بعدی است.



انواع رابطه

- **رابطه مبنا (Base Relation)**

نوعی رابطه ی نامدار که استقلال وجودی دارد و مشتق از رابطه های دیگر نیست و داده های ذخیره شده متناظر دارد که پایا هستند.

- **رابطه مجازی (Virtual Relation)**

نوعی رابطه ی نامدار که مشتق از سایر رابطه های دیگر است و داده های ذخیره شده خاص خود را ندارد. مثل مفهوم دید در پایگاههای رابطه ای

- **رابطه لحظه ای (Snapshot)**

نوعی رابطه ی نامدار و مشتق (شبیه به دید) ولی مجازی نیست بلکه واقعی است. یعنی داده های یک لحظه خاص از یک رابطه که به صورت مستقل ذخیره می شوند.

انواع رابطه (ادامه)

رابطه مشتق (Derived Relation)

رابطه ای است که به کمک یک عبارت رابطه ای (مثلا اشتراک، اجتماع و ...) بر روی رابطه های مبنا تعریف می شود. به این نوع رابطه رابطه عبارتی هم گفته می شود (Expression Relation) به این معنی که از مجموعه ای از رابطه های نامدار و با یک عبارت رابطه ای به دست آمده است.

رابطه نتیجه پرسش (Result Relation)

رابطه ای بی نام و مشتق که حاصل یک پرسش مشخص است که به صورت دایمی ذخیره نمی شود.

رابطه موقت (Temporal Relation)

رابطه ای است نامدار که استقلال وجودی دارد ولی داده های آن به صورت پایا در سیستم ذخیره نمی شود.

دامنه [میدان]

مفهوم دامنه (میدان)

- مجموعه‌ای است نامدار از مقادیر هم نوع، که حداقل یک صفت از رابطه، از آن **معنا**، **نوع** و **مقدار** می‌گیرد.
- معادل است با مفهوم Data Type در تئوری انواع.
- دامنه‌هایی که یک رابطه روی آن‌ها تعریف می‌شود، لزوماً متمایز نیستند.

مفروض $R(H)$

if $A_i \in H,\ A_j \in H,\ A_i \neq A_j \Rightarrow (D_i \neq D_j$ که لزوما چنین نیست$)$

دامنه [میدان] (ادامه)

- **تمرین:** مثالی از یک رابطه ۵-تایی که
  - دوصفت آن از یک دامنه باشد.
  - سه صفت آن از یک دامنه باشد.
- اگر m درجه رابطه و n تعداد دامنه‌ها باشد، داریم: $n \leq m$.
- برای تعریف یک رابطه در سیستم رابطه‌ای، از لحاظ تئوریک، ابتدا باید دامنه‌هایش را تعریف کرد.

دامنه [میدان] (ادامه)

مثالی از شمای پایگاه رابطه‌ای (در مدل تئوریک)

```
CREATE DOMAIN  SN       CHAR(8)  DEFAULT '00000000'
CREATE DOMAIN  SNAME    CHAR(20) DEFAULT 'noname'
CREATE DOMAIN  SJ       CHAR(4)  DEFAULT '?...?'
CREATE DOMAIN  SL       CHAR(3)  DEFAULT '?...?'
CREATE DOMAIN SD        CHAR(4)  DEFAULT '?...?'
CREATE DOMAIN CN        CHAR(6)  DEFAULT '?...?'
CREATE DOMAIN GRADE     DEC(2, 2) DEFAULT '?...?'

CREATE RELATEION STT
        (STID  DOMAIN  SN,
         STNAME  DOMAIN  SNAME,
         STJ  DOMAIN  SJ,
         STL  DOMAIN  STL,
         STD  DOMAIN  SD)

CREATE RELATION COT ….

CREATE RELATION STCOT …
```

در عمل در DBMS های رابطه ای، هم می توان از نوع دامنه های از پیش تعریف شده در آن DBMS استفاده کرد و هم نوع دامنه جدید ایجاد کرد.

دامنه [میدان] (ادامه)

تعریف دامنه اختصاصی در DBMS ها

- شما می توانید نوع دامنه مورد نظرتان را در برخی سمپادها تعریف کنید

```
CREATE DOMAIN addr VARCHAR(90) NOT NULL DEFAULT 'N/A';
CREATE DOMAIN idx INT CHECK (VALUE > 100 AND VALUE < 999);
```

دستورات زیر در SQL مطالعه شود.

- CREATE DOMAIN
- ALTER DOMAIN
- DROP DOMAIN

رابطه نرمال و غیرنرمال

- **رابطه نرمال (بهنجار- عادی Flat Relation):**

رابطه‌ای که تمام صفات آن تک‌مقداری (حداکثر دارای یک مقدار در هر تاپل) باشند.

- **رابطه غیرنرمال (Nested Relation):**

رابطه‌ای که حداقل یک صفت آن چندمقداری باشد.

- **توجه:** تعریف زیر درست **نیست:**
  - رابطه‌ای نرمال است که مقادیر تمام صفات آن اتمیک (تجزیه نشدنی یا ساده) باشند.
  - **تذکر:** ساده یا مرکّب بودن صفت نقشی در نرمال بودن و نبودن آن ندارد.

رابطه نرمال و غیرنرمال (ادامه)

مثال

صفت چندمقداری ساده

NNCOPRECO (COID , PRECOID)

COID	PRECOID
c01	{c11, c17, c08}
c02	{c03, c09}
c03	c10

یک تاپل

تبدیل به رابطه نرمال

COPRECO (COID , PRECOID)

COID	PRECOID
c01	c11
c01	c17
c01	c08
c02	c03
c02	c09
c03	c10

یک تاپل

رابطه نرمال و غیرنرمال (ادامه)

مثال

صفت چندمقداری مرکب
P# , QTY

NNSP (S# , PQTY)

SP (S# , P# , QTY)

S#	PQTY
s1	p1 100 p2 90 p3 50
s2	p1 60 p2 90
s3	p1 150

یک تاپل

تبدیل به رابطه نرمال

S#	P#	QTY
s1	p1	100
s1	p2	90
s1	p3	50
s2	p1	60
s2	p2	90
s3	p1	150

یک تاپل

رابطه نرمال و غیرنرمال (ادامه)

- **دلیل نرمال بودن رابطه در RM:**
  - سادگی در
    - ۱- نمایش ظاهری رابطه
    - ۲- دستورات DSL
      - DDL
      - DML
      - DCL
    - ۳- اجرای دستورات

مثال برای درک موارد ۲ و ۳

- درج کن
  - I_1 : $< s4, p4, 40 >$: در هر دو رابطه NNSP و SP منجر می شود به درج «تاپل در رابطه» با همان دستور ساده «درج کن تاپل را».
  - I_2 : $< s2, p3, 30 >$: با همان دستور ساده درج می شود در SP و نه NNSP .

رابطه نرمال و غیرنرمال (ادامه)

- ادامه مثال

I_1 : **INSERT INTO** $\begin{Bmatrix}\text{NNSP}\\ \text{SP}\end{Bmatrix}$

TUPLE (S4 , P4 , 40**);**

I_2 : **INSERT INTO** SP

TUPLE (S2 , P3 , 30**);**

→ **امکان پذیر**

I_2 : **INSERT INTO** NNSP

TUPLE (S2 , P3 , 30**);**

→ **امکان ناپذیر**

- دلیل: تاپلی با کلید S2 وجود دارد.

برای درج I_2 در NNSP منطقا چه باید کرد؟

✓ در رابطه غیرنرمال دستورات ساده‌ی تاپلی کار نمی‌کنند.

مزایا و معایب رابطه نرمال و غیرنرمال

نوع رابطه	مزایا	معایب
نرمال	سادگی (۱- ... ۲-... ۳-...)	طولانی شدن کلید
	تقارن صفات (پیاده‌سازی در سطح فایلینگ ساده‌تر) (نقش تمام صفات در عبارت WHERE وقتی که شرط جستجو را با theta می‌دهیم، یکسان است، زیرا همه تک‌مقداری‌اند. SELECT…. FROM …. WHERE A<(=)(>) ‘Single Value’ چنین تقارنی در رابطه غیرنرمال وجود ندارد.)	افزونگی (ادراکی یا منطقی) (این نوع افزونگی که در مرحله طراحی پیدا شده ممکن است منجر به افزونگی فیزیکی بشود یا نشود؛ بستگی دارد به نحوه پیاده‌سازی رابطه در سطح فایلینگ. اگر تناظر یک به یک باشد، که هر تاپل هم با یک رکورد پیاده‌سازی شود، افزونگی فیزیکی نیز پیش می‌آید.)
		سنگین و زمانگیر کردن کار طراحی منطقی پایگاه داده‌ها
		کاهش سرعت بازیابی در بعضی از پرسش‌ها
		دشواری در نمایش طبیعی ارتباط سلسله مراتبی بین اشیاء و وراثت
غیرنرمال	[عکس معایب رابطه نرمال]	پیچیدگی (۱- ... ۲-... ۳-...)
		عدم تقارن صفات

مزایا و معایب رابطه نرمال و غیرنرمال (ادامه)

- **در عمل با کلید طولانی چه باید کرد؟**

- از یک کلید ساختگی استفاده می‌کنیم؛ یعنی یا خودمان به صورت دستی و یا خود سیستم به صورت خودکار به هر سطر یک شماره می‌دهد.

این تکنیک چه مزایا و چه معایبی دارد؟

کلید در مدل رابطه‌ای

- اصطلاح **کلید**، یک اصطلاح عام است و گونه‌هایی دارد:

۱- سوپرکلید (اَبَر کلید): SK

۲- کلید کاندید (کلید نامزد): CK

۳- کلید اصلی: PK

۴- کلید بدیل: AK

۵- کلید خارجی: FK

کلید در مدل رابطه‌ای – سوپرکلید

- رابطه $R(\underbrace{A_1, A_2, \ldots, A_m}_{H_R})$ را در نظر می‌گیریم.
- **سوپرکلید (Super Key)**

تعریف

هر زیر مجموعه $S \subseteq H_R$ که یکتایی مقدار داشته باشد.

- اگر t_i و t_j دو تاپل دلخواه و متمایز از R باشند و $t_i(S) \neq t_j(S)$، آنگاه S یک سوپرکلید است.
- اگر N تعداد SKهای رابطه R باشد، $N \geq 1$ است، زیرا در بدترین حالت خود H سوپرکلید می‌شود. چون بدنه، مجموعه است و تاپل تکراری نداریم.

$$1 \leq N \leq 2^m - 1$$

- **کاربرد سوپرکلید:**
  - در عمل، فاقد کاربرد مستقیم، در تئوری در بحث طراحی.
  - در SQL: با UNIQUE محدودیت یکتایی مقدار را اعمال می‌کنیم.

کلید در مدل رابطه‌ای – کلید کاندید

کلید کاندید (Candidate Key)

تعریف: هر زیرمجموعه $K \subseteq H_R$ که دو ویژگی داشته باشد:

۱- یکتایی مقدار

۲- کاهش‌ناپذیری (Irreducibility) یا کمینگی (Minimality)

- $K \subseteq H_R$ کاهش‌ناپذیر است هرگاه هر زیرمجموعه محض از K، خود یکتایی مقدار نداشته باشد.
- هر زیرمجموعه از H_R به نحوی که یک صفت را از آن حذف کنیم دیگر یکتایی مقدار نداشته باشد.

مثال:

رابطه	کلید کاندید
STT	STID
COT	COID
STCOT	(STID, COID)
S	S#
P	P#
SP	(S#, P#)

کلید در مدل رابطه‌ای – کلید کاندید (ادامه)

- CKها بر اساس قواعد معنایی محیط به دست می‌آیند.

دو حالت مختلف:

- هر کارمند در بیش از یک پروژه می‌تواند شرکت داشته باشد.

شماره‌ملی شماره‌پروژه شماره‌کارمند

EMPROJ (E#, J#, ENC, …)

CK CK

- هر کارمند در حداکثر یک پروژه می‌تواند شرکت داشته باشد.

EMPROJ (E#, J#, ENC, …)

CK CK

کلید در مدل رابطه‌ای – کلید کاندید (ادامه)

- **خصوصیات کلید کاندید:**
  - هر CK، SK هم هست ولی عکس این مطلب صادق نیست.
  - هر رابطه حداقل یک CK دارد، زیرا در بدترین حالت، خود H_R می‌شود CK.
  - رابطه می‌تواند بیش از یک CK داشته باشد.

رابطه R حداکثر چند CK دارد؟

- بیشترین تعداد CK زمانی است که به اندازه نصف تعداد صفات رابطه در CK شرکت کنند.
- CKهای رابطه می‌توانند همپوشا باشند، یعنی حداقل در یک صفت مشترک باشند.

کلید در مدل رابطه‌ای – کلید کاندید (ادامه)

- **نقش کلید کاندید:** تضمین‌کننده عملیات تاپلی (و نه مجموعه‌ای) یا امکان ارجاع به تک تاپل در رابطه را فراهم می‌نماید.
- هر **زیر**مجموعه از CK، یک SK است.
- CK(های) رابطه باید به سیستم معرفی شوند.

CREATE RELATEION EMPROJ
(E# … NOT NULL,
J# … NOT NULL,
ENC … NOT NULL)

CANDIDATE KEY (E#, J#)
CANDIDATE KEY (J#, ENC)

- تئوری این را می‌گوید ولی در عمل، پکیج‌ها نمی‌پذیرند.

کلید در مدل رابطه‌ای – کلید اصلی

کلید اصلی (Primary Key)

کلید اصلی (PK) یکی از CKها است به انتخاب طراح.

- در عمل با عبارت PRIMARY KEY تعریف می‌شود.

ضوابط انتخاب کلید اصلی:

۱- شناسه رایج در محیط باشد.

۲- مقادیرش همیشه معلوم باشد (نه هر CK، آنکه به عنوان PK انتخاب می‌شود)

۳- کوتاه‌تر بودن طول

۴- حتی‌الامکان مقادیرش تغییر نکند.

کلید در مدل رابطه‌ای – کلید اصلی (ادامه)

- **دلایل لزوم انتخاب کلید اصلی:**

۱- دلیل تاریخی: PK مفهوم آشناتر برای طراحان است.

۲- ایجاد شاخص اتوماتیک روی PK.

۳- در بحث جامعیت DB: چون محدودیت هیچ‌مقدارناپذیری را اگر به همه CKها بدهیم خیلی محدود کننده است. کلید CKای که این محدودیت را روی آن اعمال می‌کنند می‌شود PK.

- **اصالت مفهومی در مدل رابطه‌ای با کلید کاندید (CK) است.**

کلید در مدل رابطه‌ای – کلید بدیل

کلید بدیل (Alternate Key)

به هر کلید کاندید (CK) غیر از کلید اصلی (PK)، کلید بدیل (AK) گویند.

- در عمل متناظر ندارد.
- اگر N تعداد AKهای رابطه R باشد، داریم $N >= 0$.

ممکن است فقط یک CK داشته باشیم که آن هم می‌شود PK و دیگر AK نداریم.

کلید در مدل رابطه‌ای – کلید خارجی

کلید خارجی (Foreign Key)

- *در عمل:* $T_2.C$ در T_2، کلید خارجی است هرگاه در T_1، کلید اصلی باشد.
- *در تئوری:* صفت (ساده یا مرکب) $R_2.A_i$ در R_2 کلید خارجی است، هرگاه در R_1، نه لزوماً متمایز از R_2، کلید کاندید (CK) باشد.
- صفت (صفات) کلید خارجی باید هم‌میدان با صفت (صفات) کلید کاندید باشد و معمولا هم‌نام با کلید کاندید است، ولی گاه لازم می‌شود که نام دیگری داشته باشد.

مثال

رابطه	کلید خارجی	دلیل: CK در
STCOT	STID	STT
STCOT	COID	COT
SPJ	S#	S
SPJ	P#	P
SPJ	J#	J

کلید در مدل رابطه‌ای – کلید خارجی (ادامه)

- اگر N تعداد FKهای رابطه R باشد، داریم $N \geq 0$.
- معرفی کلید خارجی با عبارت FOREIGN KEY انجام می‌شود.
- **نقش کلید خارجی:** برای نمایش ارتباطهای صریح بین نوع موجودیت‌ها (و در نتیجه بین نمونه‌های آنها) به کار می‌رود. **منظور از ارتباط صریح، ارتباطی است که در مدل ER با لوزی مشخص شده است.**

مثال

S (S#, …) — CK

P (P#, …) — CK

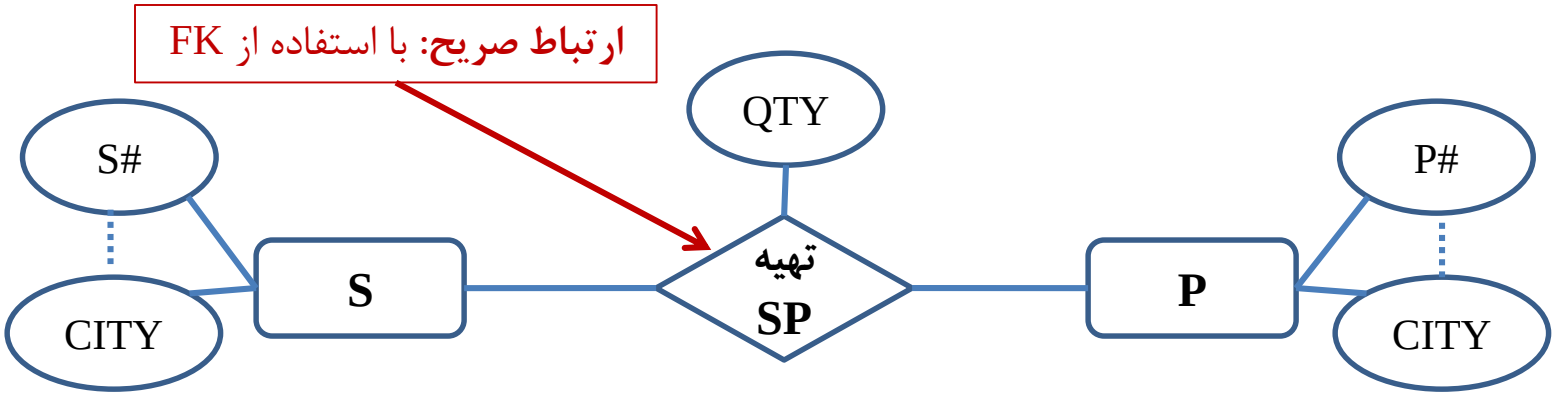
SP (S#, P#, …) — FK, FK — CK

SCOT (STID, COID, …) — FK, FK — CK

کلید در مدل رابطه‌ای – کلید خارجی (ادامه)

- آیا FK تنها امکان نمایش ارتباط است یا امکان دیگری هم وجود دارد؟
- FK تنها امکان نیست.
- وجود هر صفت مشترک [هم دامنه و در عمل، هم‌نام (نه لزوماً)]، در عنوان مثلاً دو رابطه، نمایشگر نوعی ارتباط است بین دو نوع موجودیت که با آن دو رابطه نمایش داده‌ایم.

مثال



S (S#, … , CITY)

P (P#, … , CITY)

SP (S#, P#, …)

ارجاع

ارتباط ضمنی: از طریق هر صفت مشترک؛ صفت هم‌معنا (از یک میدان) و نه لزوماً هم‌نام

بحث تکمیلی: کلید خارجی – گراف ارجاع

مفهوم گراف ارجاع

- FK امکانی است برای ارجاع از یک رابطه به رابطه‌ای دیگر
- هر مقدار معلوم FK، امکانی است برای ارجاع مقداری از تاپل(هایی) از رابطه(هایی) به تاپلی از رابطه(هایی).

تعریف **گراف ارجاع** امکانی است برای نمایش ارجاعات بین رابطه‌ها که در آن هر گره، نمایانگر یک رابطه و هر یالِ جهت‌دار، نمایانگر ارجاع از یک رابطه (حاوی کلید خارجی) به رابطه دیگر (حاوی کلید کاندید) است.

مثال

$$P \longleftarrow SP \longrightarrow S$$

- **شکل کلی مسیر ارجاع:**

$$R_m \rightarrow R_{m-1} \rightarrow \cdots \rightarrow R_2 \rightarrow R_1$$

با این ارجاع می شود **چرخه ارجاع**

- مسیر ارجاع می‌تواند **چرخه‌ای** باشد.

گراف ارجاع (ادامه)

- چرخه ارجاع می‌تواند تک‌رابطه‌ای باشد و این در صورتی است که یک رابطه خود ارجاع (Self-Referencing) داشته باشیم.
- هنگامی که FK تعریف می‌کنیم باید معنایش را نیز بگوییم.

مثال

چرخه ارجاع بین دو رابطه کارمند و اداره.

DEPT ⇄ **EMPL**

شماره کارمند مدیر اداره

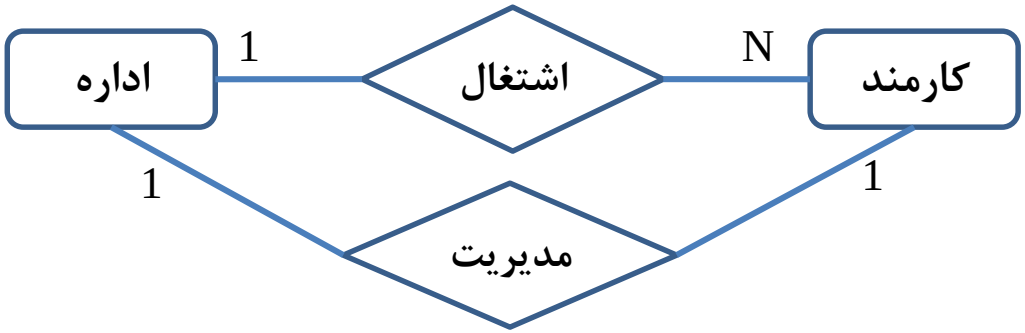
DEPT (D#, DTITLE, … , E#)

Unique

شماره اداره محل کار

EMPL (E#, ENAME, …, D#)

- بر اساس کدام مدلسازی این طراحی انجام شده است؟



گراف ارجاع (ادامه)

چرخه ارجاع تک‌رابطه‌ای کارمند با خودش.

EMPL (E#, ENAME, ENC, …, EPHONE, EMANAGER#)

شماره مدیر

EMPL

- **نکته‌های مثال اخیر:**
  - مثالی است از حالتی که در آن R1 و R2 در تعریف FK، لزوماً متمایز نیستند.
  - رابطه EMPL به خود رجوع کننده (خود ارجاع) است.
  - اگر m درجه EMPL باشد و n تعداد دامنه‌هایش باشد، داریم: $n \leq m-1$
  - لزوم دگر نامی شماره کارمندی مدیر، چون عنوان رابطه (Heading)، مجموعه‌ای از نام صفات است.
  - **تمرین:** این طراحی بر اساس کدام مدلسازی انجام شده است؟

گراف ارجاع (ادامه)

مثال چرخه ارجاع سه رابطه‌ای

دانشکده استاد

PROF (PRID, PRNAME, …, DEID)

DEPT(DEID, DTITLE, …., UNID)

UNIV(UNID, UNAME, …, UNPRESNUM)

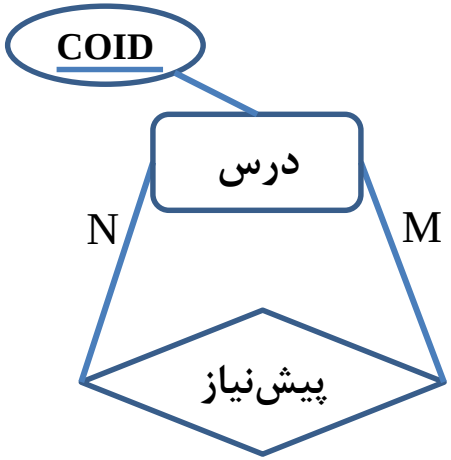
شماره استادی رئیس دانشگاه

تمرین: این طراحی بر اساس کدام مدلسازی انجام شده است؟

گراف ارجاع (ادامه)

نکته صرف وجود دور در ERD چرخه ارجاع ایجاد نمی‌شود.

مثال



COT (COID, …)

COPRECO(COID, PRECO)

COPRECO ⟶ COT

- **در چه وضعی چرخه ارجاع پدید می‌آید؟**
- باید به چندی ارتباط‌ها توجه شود.

جامعیت در مدل رابطه‌ای

جامعیت پایگاه داده‌ها (DB Integrity)

تعریف: صحت، سازگاری [، دقت و اعتبار] داده‌های ذخیره شده در پایگاه داده‌ها

جنبه‌های کیفی داده (Data Quality Features)

- مسئولیت کنترل جامعیت DB با RDBMS است.
- بر اساس اطلاعاتی که کاربر [تیم طراح – پیاده‌ساز] به سیستم می‌دهد.

← **قواعد یا محدودیت‌های جامعیتی** (Integrity Rules/Constraints)

- IRها [ICها] با استفاده از دستورات زبان پایگاهی به سیستم داده می‌شوند.

← **اعلانی:** قواعد به نحوی اعلان می‌شوند.

← **اجرایی:** قواعد در یک رویه به سیستم داده می‌شوند.

جامعیت در مدل رابطه‌ای (ادامه)

- هر DBMSای باید بتواند جامعیت پایگاه داده‌ها را کنترل و تضمین کند.
- **دلیل:** زیرا همیشه ممکن است عواملی سبب نقض جامعیت شوند. از جمله:
  - اشتباه در برنامه‌های کاربردی
  - اشتباه در وارد کردن داده‌ها
  - وجود افزونگی کنترل نشده
  - اجرای همروند تراکنشها به گونه‌ای که داده نامعتبر ایجاد شود.
  - خرابی‌های سخت‌افزاری و نرم‌افزاری

قواعد جامعیت در مدل رابطه‌ای

IRها [ICها] در مدل رابطه‌ای

۱- **قواعد [محدودیت‌های] عام:** ناوابسته به داده‌های محیط: فراقواعد (MetaRules)

۲- **قواعد [محدودیت‌های] خاص:** وابسته به داده‌های محیط: قواعد کاربری (User Defined)

یا قواعد فعالیت‌های محیط (Business Rules)

قواعد عام در مدل رابطه‌ای

- قاعده C1: جامعیت موجودیتی
- قاعده C2: جامعیت ارجاعی

قواعد عام در مدل رابطه‌ای – قاعده جامعیت موجودیتی C1

قاعده (محدودیت) C1 – قاعده جامعیت موجودیتی (Entity IR)

- ناظر است به PK.
- هیچ جزء تشکیل‌دهنده PK نباید هیچ‌مقدار (Null) داشته باشد.
- دلیل:
  - ✓ PK عامل تمییز تاپل‌ها است.
  - ✓ تاپل در مدل رابطه‌ای نمایشگر نمونه موجودیت است.
  - ✓ PK عامل تمییز نمونه موجودیت‌ها است.

  } عامل تمییز خود نمی‌تواند ناشناخته باشد.

- مکانیزم اِعمال C1: اعلان PK به سیستم ← کنترل می‌کند
  - ۱- محدودیت یکتایی مقدار (با UNIQUE
  - فقط این محدودیت کنترل می‌شود)
  - ۲- محدودیت هیچ‌مقدارناپذیری

قواعد عام در مدل رابطه‌ای – قاعده جامعیت ارجاعی C2

- **قاعده (محدودیت) C2 – قاعده جامعیت ارجاعی (Referential IR)**
  - ناظر است به FK.
  - اگر $R_2.A_i$ در R_2، کلید خارجی باشد، مقدار A_i در هر تاپل از R_2 باید در R_1 مقدار قابل انطباق (Matchable Value) داشته باشد.
  - به عبارت دیگر باید هر مقدار معلوم A_i در R_2، در R_1 نیز وجود داشته باشد. یعنی در عمل می‌تواند در R_2 مقدار آن Null باشد (البته اگر جزء تشکیل‌دهنده کلید R_2 نباشد).
  - دلیل:
    - FK عامل ارجاع است؛ ارجاع به نمونه موجودیت (ارجاع مقداری و نه ارجاع از طریق اشاره‌گر).
    - در واقعیت نمی‌توان به نمونه موجودیت ناموجود ارجاع داد.

قواعد عام در مدل رابطه‌ای – قاعده جامعیت ارجاعی C2 (ادامه)

STT (STID, …)

777
888
444

STCOT (STID, COID, …)

777	CO1
…	…
444	CO4

INSERT INTO STCOT
VALUES (‘999’, ‘CO9’, …)

- چون برای 999 مقدار قابل انطباق در STT وجود ندارد، پس این درخواست رد می‌شود.

قواعد عام در مدل رابطه‌ای – قاعده جامعیت ارجاعی C2 (ادامه)

- برای اِعمال قاعده C2 در مدل رابطه‌ای لازم است:

۱- معرفی FKها به سیستم

۲- دادن گراف ارجاع

۳- مشخص کردن روش اِعمال در عملیات حذف و به‌هنگام‌سازی مقدار کلید اصلی

(در درج روش خاصی لازم نیست و در صورت عدم وجود تاپل مرجع، درخواست رد می‌شود.)

```
CREATE TABLE STCOT
        (STID     CHAR(6) NOT NULL
         COID     CHAR(6) NOT NULL
         TR       CHAR(1)
         YR       CHAR(5)
         GR       DEC(2, 2))
        CHECK (0 <= GR <= 20)
        PRIMARY KEY (STID, COID)
        FOREIGN KEY STID REFERENCES STT (STID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
        FOREIGN KEY COID REFERENCES COT (COID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
```

۱- معرفی FK

۲- گراف ارجاع

۳- روش اِعمال (انتشار عمل)

قواعد عام در مدل رابطه‌ای – قاعده جامعیت ارجاعی C2 (ادامه)

روش‌های اِعمال C2 در حذف (بعضاً در به‌هنگام‌سازی):

۱- روش CASCADE: انتشاری یا تسلسلی

در این روش با حذف تاپل مرجع، تمام تاپل‌های رجوع کننده به آن حذف می‌شوند.

هر چه گراف ارجاع سنگین‌تر باشد، کار سیستم در اینجا بیشتر است.

DELETE FROM STT
WHERE STID=‘444’

DELETE FROM STCOT
WHERE STID=‘444’

۲- روش RESTRICTED: روش منوط به ... (یا مشروط به ...) یا روش تعویقی

- در این روش اگر بخواهیم تاپل مرجع را حذف کنیم، ابتدا باید تاپل‌های ارجاع کننده به آن حذف شوند.
- کلمه کلیدی در SQL: Restrict

قواعد عام در مدل رابطه‌ای – قاعده جامعیت ارجاعی C2 (ادامه)

- روش‌های اِعمال C2 در حذف (و بعضاً در به‌هنگام‌سازی):

۳- روش SET TO NULL: روش هیچ‌مقدارگذاری یا Nullifying

- در این روش باحذف تاپل مرجع، FK در تاپل‌های رجوع کننده Null می‌شود به شرط آنکه FK جزء سازنده PK نباشد.
- دستور SQL: Set Null

۴- روش SET TO DEFAULT: روش درج پیش‌فرض

- در این روش، با حذف تاپل مرجع، FK با مقدار پیش‌فرض جاگذاری می‌شود به شرط آنکه FK جزء سازنده PK نباشد.
- دستور SQL: Set default

قواعد عام در مدل رابطه‌ای – قاعده جامعیت ارجاعی C2 (ادامه)

- **روش‌های اِعمال C2 در حذف (و بعضاً در به‌هنگام‌سازی):**

۵- روش NO ACTION: عدم اقدام

برای این روش دو پیشنهاد داده شده است:

۵-۱- عدم اقدام مطلق: مثلاً مجاز نبودن عمل حذف تاپل مرجع و نمایش خطا.

۵-۲- انجام عمل خواسته شده و نه اقدام دیگر: تاپل مرجع حذف بشود ولی اقدام دیگری انجام نشود. در این مورد طراح-پیاده‌ساز می‌پذیرد که موقتاً (معمولا تا پایان یک تراکنش و نه بعد از آن) محدودیت C2 نقض شود.

*No Action گاها معادل Restriction دیده می شود.

- در حالت وجود چرخه ارجاع کدام روش انجام شدنی است؟
  - نمی‌توان روش RESTRICTED را در حالت کلی اِعمال کرد. با روش CASCADE هم ممکن است تاپل‌های ناخواسته حذف شود.
  - در این مواقع NO ACTION را انتخاب می‌کنیم.

قواعد خاص در مدل رابطه‌ای

- **قواعد خاص در مدل رابطه‌ای:**
  - محدودیت دامنه‌ای (میدانی)
  - محدودیت صفتی
  - محدودیت رابطه‌ای
  - محدودیت پایگاهی

قواعد خاص در مدل رابطه‌ای (ادامه)

محدودیت دامنه‌ای (میدانی)

- این محدودیت ناظر است به دامنه، مشخص‌کننده نوع و طیف مقادیر دامنه
- در همان دستور CREATE DOMAIN اعلان می‌شود.

مثال

CREATE DOMAIN GRADE DEC(2, 2) **DEFAULT** '?...?' دستور ایجاد دامنه

CONSTRAINT GRADECONST نام محدودیت (اختیاری)

CHECK VALUE BETWEEN (0, 20)

DROP DOMAIN GRADE دستور حذف دامنه

قواعد خاص در مدل رابطه‌ای (ادامه)

محدودیت صفتی [ستونی]

- این محدودیت ناشی می‌شود از محدودیت دامنه‌اش
- صفت می‌تواند محدودیت‌های دیگری هم داشته باشد، به شرطی که ناقض محدودیت دامنه‌ای‌اش نباشد.

محدودیت‌های ناظر به صفت:

۱- صفت نمره باید بین ۰ تا ۲۰ باشد.

۲- صفت سن کاهش نمی‌یابد (محدودیت پردازشی).

محدودیت ۱، یک **محدودیت وضعیتی** است ولی محدودیت ۲، یک **محدودیت گذاری** است.

قواعد خاص در مدل رابطه‌ای (ادامه)

- محدودیت صفتی را چگونه می‌توان به سیستم اعلان کرد؟

۱- با تعریف دامنه‌اش اعلان می‌شود.

۲- در همان دستور CREATE TABLE با عبارت CHECK اعلان می‌شود.

جدول انتخاب درس

```
CREATE TABLE STCOT
        (STID …
         COID …
         TR …
         GR …)
        CHECK (0 <= GR <= 20)
        CHECK (0 <= GR <= 20 and …)
```

۳- با ASSERTION اعلان می‌شود.

۴- با TRIGGER به سیستم داده می‌شود (اجرایی).

قواعد خاص در مدل رابطه‌ای (ادامه)

محدودیت رابطه‌ای

- ناظر است به تاپل‌های یک رابطه (درون رابطه‌ای Intra-relational).
- حیطه اِعمالش یک رابطه است و مقادیر مجاز یک متغیر رابطه‌ای را مشخص می‌کند.
- باید در هر عملی که بر روی رابطه انجام می‌شود (که منجر به تغییر در متغیر رابطه‌ای می‌گردد) کنترل شود.

تعداد واحد درس‌های عملی قابل اخذ برای هر فرد در هر ترم، حداکثر ۲ واحد است.

تهیه‌کنندگان ساکن شهر C2 نمی‌توانند مقدار وضعیت بیش از ۱۵ داشته باشند.

قواعد خاص در مدل رابطه‌ای (ادامه)

محدودیت پایگاهی

- ناظر است به تاپل‌های بیش از یک رابطه که به نحوی با هم ارتباط معنایی [منطقی] دارند.

مثال رابطه بین جداول STT و STCOT

یا رابطه بین جداول S و SP

مثال دانشجوی رشته کامپیوتر نمی‌تواند درس آمار و احتمال را از گروه آموزشی D13 (دانشکده ریاضی) انتخاب کند. رابطه‌های دخیل: STT، COT و STCOT

مثال تهیه‌کننده ساکن شهر C7 با وضعیت کمتر از ۱۵، نمی‌تواند قطعه آبی رنگ با وزن بیش از ۱۰ گرم به تعداد بیش از ۱۰۰ عدد تهیه کند.

- محدودیت‌های رابطه‌ای و پایگاهی چگونه اِعمال می‌شوند؟
  - با ASSERTION (اِعلانی)
  - با TRIGGER (اجرایی)

امکانات بیان محدودیت‌ها – اِظهار

اِظهار – ASSERTION

- امکانی است اِعلانی برای بیان محدودیت‌های رابطه‌ای و پایگاهی [و صفتی]

CREATE ASSERTION *name*
[BEFORE|AFTER *action*
ON *tablename* **]**
CHECK *condition(s)*

- در قسمت *condition(s)* می‌توان یک شرط ساده، یک عبارت بولی شامل چند شرط و نیز یک عبارت SELECT معتبر نوشت (همانطور که بعد از عبارت WHERE نوشته می‌شود).

- دستور حذف اِظهار

DROP ASSERTION ***name***

امکانات بیان محدودیت‌ها – اِظهار (ادامه)

با این اظهار، محدودیت یکتایی مقادیر صفت کد ملی STNATID اعلان می‌شود.

```
CREATE ASSERTION  UNC-CHECK
        CHECK (UNIQUE(SELECT  STNATID  FROM  STT))
```

با این اظهار این محدودیت که «جمع واحدهای انتخابی دانشجو در هر ترم–سال نباید بیش از ۲۰ واحد باشد»، اعلان می‌شود.

```
CREATE ASSERTION  TOTCRED-CHECK
        CHECK (NOT EXISTS (SELECT STID
                                FROM COT  JOIN  STCOT
                                GROUP BY  (STID, TR, YR)
                                HAVING SUM(CREDIT) > 20) )
```

امکانات بیان محدودیت‌ها – اِظهار (ادامه)

همه دانشجویان دانشکده مهندسی کامپیوتر (CE) باید درس مبانی برنامه‌سازی (با کد ۴۰۱۱۱) را اخذ کرده باشند.

```
CREATE ASSERTION   ELEM-CHECK
      CHECK (NOT EXISTS
                  ( SELECT  *  FROM  STT
                        WHERE  DEPT='CE'  AND
                              NOT EXISTS
                              ( SELECT * FROM  STCOT
                                 WHERE  STCOT.STID = STT.STID
                                       AND  STCOT.COID='40111'  ) )
```

امکانات بیان محدودیت‌ها – رهانا

رهانا [راه‌انداز] – TRIGGER

- امکانی است اجرایی برای اِعمال محدودیت‌های [صفتی،] رابطه‌ای و پایگاهی قبل یا بعد از بروز یک رویداد و یا به جای یک رویداد (معمولا تغییر دهنده داده‌ها).

```
CREATE TRIGGER name
        {BEFORE | AFTER | INSTEAD OF}
        {INSERT | DELETE | UPDATE OF columnlist
        ON tablename
        [REFERENCING { OLD ROW | NEW ROW | OLD TABLE | NEW TABLE} AS name ]
        [FOR EACH {ROW | STATEMENT}]
        {(WHEN condition(s)
                SQL 2003 Procedure
        )}
```

- مفهوم نظری TRIGGER: مفهوم قاعده فعال [مفهوم محوری است در ADBMSها]

ساختار (قاعده ECA): if **E**vent on **C**ondition, then **A**ction

- Insert
- Delete
- Update

امکانات بیان محدودیت‌ها – رهانا (ادامه)

- با FOR EACH ROW بعد از بروز رویداد در هر سطر عبارت رهانا اجرا شود.
- با FOR EACH STATEMENT فقط یک بار پس از بروز رویداد (با هر تعداد سطر متاثر از آن)، عبارت رهانا اجرا شود.

این رهانا این محدودیت را که «حقوق کارمند هیچگاه کاهش نمی‌یابد» اِعمال می‌کند.

```
CREATE TRIGGER EMP-PAY-TRIG
        BEFORE UPDATE OF EMPSAL
        ON EMPL
        REFERENCING OLD AS OEMPL,  NEW AS  NEMPL
        FOR EACH ROW
        (WHEN OEMPL.EMPSAL > NEMPL.EMPSAL
                -- ROLLBACK TRANSACTION
                -- Standard SQL: Print ‘salary cannot be decreased’;
                -- PostgreSQL: RAISE NOTICE ‘salary cannot be decreased ’;
        )
```

امکانات بیان محدودیت‌ها – رهانا (ادامه)

این رهانا باعث حفظ سازگاری در جدول PROF می‌شود تا همواره صفت SALAUG حاوی آخرین میزان افزایش حقوق استاد باشد.

```
CREATE TRIGGER EMP-PAY-TRIG
        AFTER UPDATE OF  PSALARY
        ON  PROF
        REFERENCING OLD AS OPROF,  NEW AS  NPROF
        FOR EACH ROW
        (UPADATE   PROF
                SET   SALAUG=NPROF.PSALARY – OPROF.PSALARY
                WHERE   PROF.PID=OPROF.PID
        )
```

اگر بیش از یک عبارت باشد، آنها را داخل BEGIN و END قرار می‌دهیم.

امکانات بیان محدودیت‌ها – رهانا (ادامه)

از کاربردهای رهانا، استفاده از آن در انجام عملیات ذخیره‌سازی از دید خارجی است (به خصوص در سمپادهایی که از عملیات در دید خارجی پشتیبانی نمی‌کنند).

STT1 (STID, NAME, MAJOR, LEVEL**)**
STT2 (STID, DEPT, BDATE, NATID**)**

```
CREATE VIEW CE-STT
        AS SELECT STID, NAME, MAJOR
        FROM STT1 JOIN STT2
        WHERE DEPT='CE' AND LEVEL ='BS'

CREATE TRIGGER INS-VIEW-TRIG
        INSTEAD OF INSERT ON CE-STT
        REFERENCING NEW AS NST
        FOR EACH ROW
        BEGIN
            INSERT INTO STT1 VALUES ( NST.STID, NST.NAME, NST.MAJOR, 'BS')
            INSERT INTO STT1 VALUES ( NST.STID, 'CE', NULL, NULL)
        END
```

امکانات بیان محدودیت‌ها – رهانا (ادامه)

این رهانا باعث اعمال قاعده C2 در عمل حذف می‌شود.

```
CREATE TRIGGER  DEL-TRIG
        BEFORE   DELETE
        ON  COT
        REFERENCING OLD AS OCOT
        FOR EACH ROW
        (DELETE   FROM   STCOT
                WHERE   STCOT.COID=OCOT.COID  )
```

امکانات بیان محدودیت‌ها – رهانا (ادامه)

با فرض داشتن جدول زیر دو رهانا برای اهداف زیر بنویسید:

Profs(PID, Pname, Grade, Salary, Score, Rank)

- یک رهانا بنویسید که با هر واحد افزایش در پایه (Grade) استاد، به صورت خودکار، ۵ درصد به حقوق (Salary) او اضافه شود و ده امتیاز هم به مجموع امتیازات او (Score) اضافه گردد.
- یک رهانا بنویسید که با رسیدن امتیاز یک استاد ۱۰۰۰۰ ، مرتبه علمی (Rank) او به صورت خودکار به Professor تغییر کند.

پرسش و پاسخ . . .

ایمیل : zarepour@iust.ac.ir

ارتباط حضوری: ساعت مشخص شده در برنامه هفتگی به عنوان رفع اشکال دانشجویی

www.ezarepour.ir