به نام آنکه جان را فکرت آموخت

چراغ دل به نور جان برافروخت

پایگاه داده ها (بانک اطلاعاتی)

نوع درس : تئوری

دوره : کارشناسی

رشته : مهندسی کامپیوتر

تعداد واحد : ۳

مدرس : علیرضا عظیمی

3/4- حساب رابطه ای دامنه ای (***domain relational calculus***)

آقای کاد دوسال پس از چاپ مقاله اصلی خود و ارائه جبر رابطه ای ، در مقاله دیگری در سال 1972 تئوری خود را به صورت دیگری نیز ارائه داد . او این روش را حساب رابطه ای تاپلی (***tuple relational calculus***) نامید. پنج سال پس از آن دانشمندی دیگر روش سومی به نام حساب رابطه ای دامنه ای (***domain relational calculus***) ارائه داد . این روش از قدرت محاسباتی مساوی برخوردار است و از روش آقای کاد ساده تر می نماید. در این بخش این روش ارائه می شود . خوانندگان علاقمند به حساب رابطه

ای تاپلی را به ارجاع مـی دهـیم و یـادآورد مـی شـویم کـه ایـن روش فـرق زیـادی بـا ***predicate calculus*** ندارد .

شکل کلی عبارت در حساب رابطه ای دامنه ای چنین است :

$$\{< c_1, c_2, .. c_n > | \, p(c_1, c_2, ..., c_n, c_{n+1}, ...)\}$$

و معنـــی آن ایـــن اســـت : ســـتون هـــای cn,…c1 را بـــده بـــه طوریکـــه شرط $P(c_1, c_2, ..., c_n, c_{n+1}, ...)$ برقرار باشد .

قواعد زیر باید رعایت شود :

- برای ارتباط متغیر ها به جداول ، از تعلق $(\in)$ استفاده می شود و شـرط همتـایی (***same arity***) باید رعایت شود .
- ترکیب شرط ها با $\Rightarrow, \neg, \vee, \wedge$ (معادل " و " ، " یا " ، " نه " ، " نتیجه می دهد ") انجام می شود .
- عدم تعلق با منفی کردن شرط (استفاده از $\neg, \in$) انجام می شود و علامت عدم تعلق ($\notin$) تعریف نشده است (زیرا با طبیعت بانک اطلاعات همخوانی ندارد) .
- خروجی دستور شامل همه ستونهای $< c_1, c_2, ..., c_n >$ خواهد بود .
- در شرط $P(c_1, c_2, ..., c_n, c_{n+1}, ...)$ می توان از متغیر های دیگر نیـز اسـتفاده کـرد ولی باید این متغیرها قبلاً با استفاده از $\exists$ (وجود دارد ، هر مقـداری) یـا $\forall$ (بـرای همه مقادیر) تعریف شده باشند . استفاده از ثابتها نیز در صورتی که متعلق به دامنـه متغیر های مربوطه باشند ایرادی ندارد .
- برای پیوند جداول از متغیر های همنام در آنها استفاده می شود .

- گاهی می توان با تغییر جملات پرس و جو ، پاسخ گویی به آن را آسان تر کرد .
- از فرمول های تئوری مجموعه ها برای پیدا کردن معادل های ساده تر استفاده می وشد . مثلاً برای پیاده سازی A-B از $A \cap \neg B$ و برای پیاده سازی $A \to B$ از $\neg A \cup B$ استفاده می شود . در این معادل سازی ، عملگر هایی مانند تفریق که مربوط به جبر رابطه ای هستند و در حساب رابطه ای جائی ندارند حذف می شوند و به جای آنها عملگر هایی می آیند که به سادگی قابل معادل سازی هستند .

مثال : دانشجویانی که معدل کل آنها بالاتر از 18 است .

دستور : $\{< s, sn, c, a, cl > | < s, sn, c, a, cl > \in stud \wedge a > 18\}$

مثال : نام دانشجویانی که معدل کل آنها بالاتر از 18 است .

دستور : $\{< sn > | \exists s, c, a, cl(< s, sn, c, a, cl > \in stud \wedge a > 18)\}$

توجه : شرط همتایی (یعنی تعلق سطر های پنج ستونی به جدول stud) با معرفی متغیر های جدید رعایت شده است .

مثال : اساتید متخصصین بانک اطلاعات که مدرک دکتری دارند .

دستور :

$\{< p, o, e, d, c > | < p, o, e, d, c > \in prof \wedge e =$ " بانک " $\wedge d =$ " دکتری "

مثال : نام و تخصص روسای دانشکده ها

حل : این پرس و جو به دو جدول استاد و دانشکده نیـاز دارد . متغیـر مـشترک P در جدول را به هم پیوند می دهد . متغیر های کمکی هر دو جدول جداگانه تعریف شـده اند . به محل باز و بسته شدن پرانتز ها دقت شـود . متغیـر هـا را بایـد در نزدیکتـرین محلی که مورد استفاده قرار می گیرند ، تعریف کرد و حـوزه عملکـرد آنهـا را بـاپرانتز مشخص نمود . حوزه عملکرد متغیر های خروجی سراسر دستور است .

دستور :

$$\{< p,e >| \exists c,cn,ci(< c,cn,ci,p > \in c\lg) \wedge \exists o,d,cl(< p,o,e,d,cl > \in prof$$

نکته : اگر متغیر c را هم مشترک می گرفتیم ، آنگاه روسای دانشکده را خواهد داد نه اساتید .

دستور :

$$\{< p,o,e,d,cl >|< p,p,e,d,cl > \in prof \wedge \exists cn,ci,pn(< cl,cn,ci,pn \in c\lg \wedge ci = \text{"تهران"}\}$$

مثال : نام دانشجویانی که هیچ درس چهار واحدی نگرفته اند .

حل : این پرس و جو به سه جدول دانشجو ، درس و گروه درس نیاز دارد . با متغیر هـای مشترک آنها را دو به دو پیوند طبیعی می دهیم . توجه شود که دانـشجو و درس ممکـن است از دو دانشکده مختلف باشند . به دو متغیر cl1 و cl2 نیاز است) .

دستور :

$$\{< sn >| \exists s,ci,a,cl1(< s,sn,ci,a,cl1 >\in stud \wedge \neg(\exists se,c,t,p,s,c(< se,c,s,t,p,sc >\in \sec \wedge \exists cn,u,cl2$$
$$(< c,cn,u,cl2 >\in crs \wedge u = 4))))\}$$

نکته : امتیاز حاسب رابطه ای دامنه ای این است که ما حرف می زند ! باید مطمئن شویم

که دقیقا همان حرف پرس و جو را زده است نه مشابه آن را ممکن است غلط باشد .

مثال : دانشکده هایی که همه اساتید آنها مدرک دکتر دارند .

حل : کلمه " همه "در شرط ، این پرس و جو را متمـایز مـی کنـد . در اینجـا بایـد از $\forall$

استفاده کرد .

دستور :

$$\{> cl,cn,ct,p >|< cl,cn,ct,p >\in c\lg \wedge \forall p2,o,e,d(< p2,o,e,d,cl >\in prof \wedge d =$$ " دکتري "$)\}$

مثال : شماره دانشجویانی که همه دروس افتاده خود را مجدداً گرفته اند .

حل : این پرس و جو را می توان به صورت زیر تغییر داد :

شماره دانشجوئی که اگر در درسی افتاده اند آنگاه مجددا آنرا گرفته اند .

برای پاسخگوئی به این پرس و جو از فرمول $\neg A \cup B$ استفاده می شود .

دستور :

$$\{< s >| \forall se,c,t,p,sc(< se,c,s,t,p,sc >\in \sec \wedge (\neg(sc < 10) \vee \exists se1,t1,p1,sc1$$
$$(< se,c,s,t1,p1,sc1 >\in \sec \wedge t1 \neq t)))\}$$

نکته : اگر به جای $t1 \neq t$ از $t1 > t$ استفاده کنیم ، غلـط مـی شـود زیـرا ممکـن اسـت

دانشجو برای بار دوم بیفتد . در این صورت شماره او داده نخواهد شد (چرا ؟)

1-4/3- خطر حلقه بی پایان

عبارتهای حساب رابطه ای دامنه ای که در آنها از $\neg$ استفاده شود می توانند خطرناک باشند یعنی تولید حلقه بی انتها کنند و خروجی آنها بی پایان باشد . به عبارت زیر توجه کنید :

$$\{< c,cn,u,cl >| \neg(< c,cn,u,cl > \in crs)\}$$

این عبارت می گوید " تمام جداول چهار ستونی ممکن ، به غیر از جدول crs را بده که ستون های آنها با جدول crs هم دامنه باشند " . خروجی این دستور بی پایان است .

در فرمول هائی که در آنها از $\forall,\exists$ استفاده می شود دو خطر وجود دارد :

1. تعداد بی پایان مقایسه و ...
2. خروجی بی پایان

به عبارت زیر توجه کنید :

$$\{< c,u,cl > \exists cn(\neg(< c,cn,u,cl > \in crs))\}$$

این عبارت علاوه بر اینکه خطر خروجی بی پایان را شبیه مثال فوق دارد ، باید برای تعداد بی پایانی رشته (***string***) که می توانند به جای نام درس بنشینند ، مقایسه را انجام دهد . در نوشتن عبارت های حاسب رابطه ای دامنه ای باید به این مهم توجه کرد .

برای پیاده سازی حساب رابطه ای دامنه ای کوشش هایی صورت گرفته که قدیمی ترین آنها ***QBE*** (***Query By Example***) و موفقترین آنها زبان پرس و جوی ***Datalog***(***Database Logic***) است . در زبان ***Datalog*** ، رابطه ها و ارتباط بین آنها آنچه در این بخش گفته شد ، با فرمی متفاوت ، بیان می شوند . برای نمونه یک مثال می آوریم . تشابه دو فرم در این مثال دیده می شود .

مثال : دانشجویان ممتاز (معدل بالای 18)

$$good_stud(s,sn,c,a,cl) \leftarrow stud(s,sn,c,a,cl) AND a > 18$$

بسیاری از پرس و جوها را ، در جبر و حساب رابطه ای ، نمی توان پاسخ داد . بـه همـین دلیل آنها از نظر محاسباتی کامل نیستند ، یعنی اطلاعاتی در جـداول هـستند کـه نمـی توان استخارج نمود . مهمترین آنها محاسبه مقادیری چون میانگین و گروهبندی داده ها است . در پیاده سازی این مفاهیم نظری ، به تدریج کمبود ها مرتفـع گردیـده ، و امـروز SQL3 از نظر محاسباتی کامل است .

مثال : دانشجویانی که میانگین دروس یک واحدی آنها زیر 10 است .

برای پاسخگوئی به این پرس و جو به دو مورد زیر نیاز است :

1- گروهبندی دروس یک واحدی به تفکیک دانشجو

2- محاسبه میانگین

هیچکدام از این دو مورد در جبر و حساب رابطه ای مستقیما پـیش بینـی نـشده اسـت . البته می توان طرحی نو در انداخت و توانایی های جدیدی (مانند توابع محاسـباتی) بـه آنها افزود .