اولین همایش منطقه ایی پژوهش در فناوری برق – دانشگاه آزاد اسلامی واحد نجف آباد –30 آذر ماه 1389

ساخت مفسر زبان و محیط یکپارچه توسعه برای زبان جبر رابطه‌ای نماد گذاری کتاب C.J.Date به زبان javascript

احمد یوسفان [1] – سارا ایزدی [2]
(1) گروه مهندسی کامپیوتر - دانشگاه کاشان
yoosofan@kashanu.ac.ir , yoosofan@gmail.com
(2) دانش آموخته مهندسی کامپیوتر دانشگاه کاشان
s_izadi65@yahoo.com

خلاصه: در این مقاله چگونگی پیاده‌سازی مفسری برای جبر رابطه‌ای نماد گذاری Date توضیح داده شده است. برای قابل حمل بودن این مفسر زبان javascript برای پیاده‌سازی آن در نظر گرفته شد تا بسادگی بتوان آنرا روی هر نوع سیستم عاملی اجرا نمود. از ویژگی‌های مهم این مفسر در برابر مفسرهای کنونی این زبانِ جبرِ رابطه‌ای می‌توان به سادگی کار با آن و محیط کاملاً بصری آن و همچنین امکان دیدن جزییات اجرای آن اشاره نمود. برای بخش تحلیل‌گر نحوی این مفسر روش LALR به کار گرفته شد که انعطاف پذیری بیشتری نسبت به روش‌های دیگری تجزیه دارد و به کمک آن به سادگی می‌توان اجرا کننده‌ی این مفسر را نیز پیاده‌سازی نمود. همچنین بخش تحلیل‌گر لغوی این مفسر به صورت کاملاً جداگانه برنامه نویسی شد تا بتوان به سادگی امکانات بیشتری را نیز به آن افزود. این مفسر زیر مجموعه‌ای مهم از دستورهای زبان جبر رابطه‌ای را می‌تواند اجرا کند و برای سادگی کار با آن برخی از دستورها به شکل ساده‌تری نوشته شدند و بنابراین همه‌ی دستور زبان جبر رابطه‌ای از نو دوباره نویسی شد به گونه‌ای که هر دو ویژگی سازگاری کامل با استاندارد و همچنین سادگی را در برداشته باشد.

کلمات کلیدی: پایگاه داده‌ها، زبان جبر رابطه‌ای، مفسر، تحلیلگر نحوی LALR، رابط کاربری بصری

1 – مقدمه

پروفسور کاد (Codd) مدل رابطه‌ای را بر پایه‌ی نظریه مجموعه‌ها (از Georg Cantor) پیشنهاد کرد[1]. البته پیش از کاد فردی به نام شیلد (D.L Childs) در ۱۹۶۸ پیشنهادی همانند آن را در گزارشی فنی داده بود که به دلیل چاپ نشدن در یک مجله معتبر، پیشنهاد او نادیده گرفته شد[2] . مدل رابطه‌ای در آغاز فقط مدلی ریاضی و بدون کارآیی پنداشته می‌شد. نخستین پیاده‌سازی به نسبت کامل و کاربردی از مدل رابطه‌ای را C.J.Date آماده نمود و به این ترتیب کاربردی بودن مدل رابطه‌ای را به خوبی نشان داد. پس از آن شرکت‌های گوناگونی این مدل را برای پیاده‌سازی سامانه‌های مدیریت پایگاه داده (DBMS) برگزیدند و زبان SQL برای کار با این سامانه‌ها گسترش یافت.

C.J.Date هنوز یکی از افراد اثر گذار در مدل رابطه‌ای و پایگاه داده‌های رابطه‌ای است که پیشنهادهایی را برای بهبود مدل رابطه‌ای و سامانه‌های پایگاه داده ارائه می‌کند. کتاب مقدمه‌ای بر سامانه‌های پایگاه داده [3] از او یکی از کتاب‌های پایه و مرجع برای درس پایگاه داده‌ها است.

2 – جبر رابطه‌ای

جبر رابطه ای مجموعه ای از عملگرهاست که بر روی عملوند های رابطه ای اعمال می شوند و یک رابطه برمی گرداند[4]. در حقیقت جبر رابطه ای یک زبان پرس و جو است که عملیات روی پایگاه داده‌ای از نوع مدل رابطه‌ای را به کمک نمادهایی بیان می‌کند. جبر رابطه‌ای یکی از زبان‌های پرس و جوی رابطه‌ای برای مدل رابطه‌ای است که پروفسور کاد آن را پیشنهاد داده است و به کمک الگوریتم کاهشی اثبات کرده است که این ربان یک زبان کامل رابطه‌ای است. این زبان برخلاف SQL به طور کامل بر پایه‌ی تئوری رابطه‌ای است. گرچه این

زبان امروزه به صورت کاربردی به کار برده نمی‌شود ولی برای آموزش مفهوم رابطه‌ها و کار با آن‌ها بسیار سودمند است.
تعدادی از عملگرهای مهم جبر رابطه‌ای، گزینش (where)، پرتو (projection {})، ضرب (TIMES)، اجتماع(UNION)، تفاضل(MINUS)، تغییر نام ویژگی(RENAME)، اشتراک(INTERSECT)، تقسیم (DIVIDE BY) پیوند(JOIN) است. جبر رابطه‌ای به طور کامل در کتاب‌های گوناگونی توضیح داده شده است. در برخی مرجع‌ها مانند [5,2-7] نمادگذاری ریاضی برای جبر رابطه‌ای برگزیده شده است ولی C.J.Date جبر رابطه‌ای را به کمک کلمه‌های کلیدی و همانند یک زبان برنامه نویسی معمولی معرفی می‌کند.

3 – گذری بر کارهای انجام شده

تا کنون برای اجرای جبر رابطه‌ای مفسرهای گوناگونی پیاده‌سازی شده است. همچنین تعدادی از پیاده‌سازی‌هایی وجود دارند که دستور جبر رابطه‌ای را به SQL تبدیل می‌کنند و آن را بر روی یک DBMS رابطه‌ای اجرا می‌کنند و پاسخ را برمی‌گرداند[8]. در [9-13] تعدادی از مفسرهای جبر رابطه‌ای معرفی شده است. فهرستی از پیاده‌سازی‌های زبان جبر رابطه‌ای در [14] همراه با توضیح کوتاهی برای هر کدام نوشته شده است. البته فهرست آن کامل و به روز نیست. بیشتر این پیاده‌سازی‌ها می‌کوشند تا جبر رابطه‌ای تعریف شده از سوی Date و Darwen را به طور کامل پوشش دهند. این زبان Tutorial D نامیده شده است و شکلِ کاملِ دستور زبان آن و توضیح‌های آن در [15] گذاشته شده است. برخی از مشکل‌های کلی این ابزارها نیاز به نصب شدن آن‌ها یا نیاز به برنامه‌های پایه‌ی دیگری برای اجرا است و مهم‌ترین مشکل آن‌ها سخت بودن کار با آن‌ها برای فردی است که به تازگی در حال آشنایی با جبر رابطه‌ای است. بنابراین کوشش شد تا پیاده‌سازی تازه‌ای از زبان جبر رابطه‌ای انجام شود که بتوان به سادگی با آن کار کرد و فقط با نصب بودن یک مرورگر پشتیبانی کننده از زبان javascript استاندارد و بدون نیاز به هیچ ابزار اضافی این مفسر بتواند اجرا شود. همچنین کوشش شد رابط گرافیکی ساده و کارآمدی برای آن آماده شود تا بتوان به سادگی نمونه‌های گوناگون از جبر رابطه‌ای را در آن اجرا نمود.

4 – دستور زبان

برای ساده شدن کار با این ابزار و در اختیار گذاشتن ساده‌تر ابزار گرافیکی و همچنین کاستن از پیچیدگی برخی از دستورها دستور زبان(grammar) تازه‌ای برای زبان جبر رابطه‌ای نوشته شد. این دستور زبان بیشتر بر پایه‌ی کتاب [3] است و با Tutorial D متفاوت است. زیرا جبر رابطه‌ای توضیح داده شده در Tutorial D و دستور زبان نوشته شده در آن برای جبر رابطه‌ای پیچیده‌تر از دستور زبان درون کتاب[3] است.

در دستور زبانی که پیاده‌سازی شده است، می‌توان چند دستور(عبارت) رابطه‌ای را در یک خط نوشت و در پایان هر کدام یک ; گذاشت و همچنین نتیجه‌ی دستور روی خروجی به صورت جدول نمایش داده می‌شود. بخشی از دستور زبان در ادامه نوشته شده است.

```
statement_list → statement  | statement_list statement;
statement → assign';'  | print';';
print → expression;
assign → target':='expression |insert |delete |update;
target → variable_name;
insert → INSERT relvar_name relation_exp;
delete →DELETE relvar_name opt_where_condition;

update → UPDATE relvar_name opt_where_condition
        SET '(' attribute_assign_commalist')'
        |UPDATE relvar_name
        SET '(' attribute_assign_commalist')';
opt_where_condition → WHERE simple_exp;
attribute_assign_commalist → attribute_assign |
        attribute_assign_commalist ',' attribute_assign;
attribute_assign → attribute_name ':=' literal;
expression → relation_exp;
relation_exp → relation;
relation → relation_wononproject  | nonproject;
project → relation_wononproject'{'opt_all_but
        attribute_name_commalist'}';
nonproject → rename | union | intersect | minus | times |
where | join | divide | semijoin | semiminus |
extend | summrize;
rename → relation_wononproject RENAME renaming |
relation_wononproject RENAME
'('renaming_commalist')';
```

5 – پیاده سازی

برای پیاده‌سازی این مفسر و محیط اجرا، زبان javascript درونِ صفحه‌ی xhtml به کار گرفته شد. زبان javascript به این دلیل برای پیاده‌سازی برگزیده شد که بر روی همه‌ی مرورگرها اجرا می‌شود و برنامه‌ی نوشته شده به این زبان را به سادگی و بدونِ نیاز به نصب هیچ نرم‌افزار دیگری می‌توان بر روی مرورگر اجرا نمود و از امکانات زبان xhtml به خوبی برای ساخت یک رابط کاربری مناسب کمک گرفت. امروزه بسیاری از ابزارها به سوی پیاده‌سازی تحت وب و به کمک javascript پیش می‌رود و شرکت‌های گوناگونی بر روی برخط کردن نرم‌افزارهای خود و به کارگیری فناوری ajax سرمایه‌گذاری کرده‌اند[16-18]. هر کدام از مرحله‌های یک مفسر کامل به صورت جداگانه پیاده‌سازی شده است. روند یاد شده در کتاب‌های متداولِ طراحی و پیاده‌سازی کامپایلر و مفسر همچون [19-23] در این پیاده‌سازی در نظر گرفته شدند.

6 – تحلیل‌گر لغوی

در زبان پیاده‌سازی شده نوع‌های زیر برای نشانه‌ها وجود دارد:
۱ـ کلمه‌های کلیدی:

INSERT, DELETE, UPDATE, DROP, VAR, DECIMAL, INTEGER, CHAR, BOOLEAN, SET, WHERE, TIMES, MINUS, SEMIJOIN, SEMIMINUS, RENAME, DIVIDEBY, RELATION, TUPLE, AS, WITH, EXTEND, ADD, SUMMARIZE, BY, ALL BUT, PER, COUNT, SUM, AVG, MAX, MIN, AND, OR, XOR, UNION, INTERSECT, NOT, COUNTD, SUM, AVGD, JOIN, TRUE, FALSE, STRING

۲ـ شناسه‌ها (identifier) ، ۳ـ رشته‌های ثابت (string literals)

۴ـ عدد صحیح(integer) ، ۵ـ عدد اعشاری (decimal number)

۶ـ عملگرها

=:	=/	=	;	,	{	}
-	+	>	<	=<	=>	
|	#	(	)	*	/	
					$	

برپایه‌ی این نوع نشانه‌ها، رشته‌ی ورودی به نشانه‌ها شکسته می‌شوند. روند به کار گرفته شده دقیقاً بر پایه‌ی روند طراحی استاندارد تحلیل‌گر لغوی است. در آغاز نمودارهای هر کدام از این نوع نشانه‌ها رسم شد. سپس تابع اصلی تحلیل‌گر لغوی بر پایه‌ی این نمودارها پیاده‌سازی شد. برای پیاده سازی نمودار از switch_case درون یک حلقه کمک گرفته شد. هر case بخشی از نمودار را نشان می‌دهد و شرط آن بر پایه‌ی یال‌هایی که از هر حالت خارج شده است، مشخص می‌شود.

7 – تحلیل‌گر نحوی

تحلیل‌گر نحوی نشانه‌ها را یکی یکی از تحلیل‌گر لغوی درخواست می‌کند و بررسی می‌کند آیا رشته‌ی وارد شده با دستور زبان جبر رابطه‌ای همخوانی دارد یا خیر. مهم‌ترین بخش‌های این تحلیل‌گر نحوی پشته و ماشین اجرای LR است. برای اینکه بتوان به سادگی اجرا کننده را نیز به تحلیل‌گر نحوی افزود، پشته و ماشین اجرا به صورت قدیمی ،توضیح داده شده در [24]، پیاده‌سازی شد.

8 – ابزار JSCC

دشوارترین کار در روشِ LALR، ساخت جدولِ آن است. برای این منظور ابزار JSCC به کار گرفته شد[25] که یک ابزار ساخت مفسر و کامپایلر است. به کمک این ابزار به سادگی می‌توان مفسر یک زبان را طراحی کرد. این ابزار می‌تواند درون هر مرورگری که استاندارد زبان javascript را پشتیبانی می‌کند(ECMAscript) اجرا شودیا اینکه به صورت خط فرمان آن را به کمک یک مفسر استاندارد زبان javascript همچون [26,27] اجرا نمود. این ابزار کد javascript مفسر موردِ نظر برای زبان داده شده را به صورت خودکار می‌سازد و در این راه روش LALR را برای تحلیل‌گر نحوی آن به کار می‌برد. به دلیلِ توانایی‌ها و انعطاف پذیری‌هایی که نیاز بود فقط از بخشِ ساختِ خودکارِ جدولِ LALR این ابزار کمک گرفته شد و از دیگر بخش‌های آن چشم‌پوشی شد.

برای به کار گرفتن این ابزار دستور زبانِ جبر رابطه‌ای به شکل دلخواهِ این ابزار، بازنویسی و به آن داده شد. این ابزار جدولِ LALR دستور زبان داده شده را درون بخشی از صفحه‌ی خود درونِ یک جدولِ html نشان می‌دهد. با برگزیدن این بخش و کپی همه‌ی جدول، می‌توان html_ای را به دست آورد که درونِ آن جدول به همراه tag_های html گذاشته شده است. برای تبدیل این جدول به به کدِ آرایه‌ای در javascript یک برنامه‌ی میانی نوشته شد. این برنامه دو آرایه برای action و goto از جدولِ LALR می‌سازد. در بخشِ action برای درایه‌های خالی صفر، برای کاهش‌ها عدد منفی، برای حالتِ پذیرش عدد یک و برای انتقال‌ها عدد مثبتِ حالت به اضافه‌ی یک گذاشته شد. در بخش goto درونِ درایه‌های خالی صفر و شماره‌ی حالتِ موردِ نظر برای دیگر درایه‌های این بخش گذاشته شد. هر داده‌ی درونِ پشته همراه با نوعِ آن گذاشته می‌شود. شماره‌ی حالت، نام غیر پایانه (به صورت رشته) و پایانه در پشته گذاشته می‌شود که هر کدام از این سه نوع دربردارنده‌ی داده‌های دیگری نیز هستند. نام پایانه در جدول گذاشته نمی‌شود بلکه خودِ رشته‌ی پایانه همراه با نوع و مقدارِ آن در پشته گذاشته می‌شود.

9 – اجرا کننده

برای پیاده‌سازی اجرا کننده، برای هر کاهشِ معادل با هر عملگرِ جبر رابطه‌ای یک تابعِ اجرا کننده‌ی آن عملگر نوشته شد که در هنگام کاهش فراخوانی می‌شود. تعدادِ حالت‌ها و نمادهایی که باید از پشته برداشته شود، به کمک دستورِ کاهشِ موردِ نظر مشخص می‌شود. چون نمادهای درون پشته دارای نوع و مقدار هستند بنابراین به سادگی در هنگام کاهش می‌توان عملِ دلخواه را روی آن‌ها درونِ تابعِ موردِ نظرِ آن انجام داد.

10 – چند نوع اصلی پیاده‌سازی شده

برای پیاده‌سازی شیءگرای مفسر، نیاز به چند شیء است که هر کدام کاری را انجام می‌دهند و دیگر بخش‌های اجرا اغلب به آن‌ها نیاز دارند.

شیء Relation

این شیء برای پیاده‌سازی رابطه در مفسر پیاده‌سازی شده است. تابع سازنده‌ی پیاده‌سازی شده برای این شیء تعدادِ سطرها و ستون‌های یک رابطه را به عنوان آرگومان‌های ورودی دریافت می‌کند و درون Relation.row وRelation.column می‌گذارد. آرایه‌ی دو بُعدی Relation.data برای ذخیره کردن مقدارهای رابطه، آرایه‌یRelation.schema برای ذخیره کردن مقدارهای عنوان(سرآیندheader) و آرایه Relation.type برای ذخیره کردن نوع در سطر عنوان (با مقدار پیش فرض رشته) تعریف شده است. تابع‌های Relation_setalldata و Relation_setallschema و Relation_setalltype به ترتیب برای مقدار دهی کامل داده‌ها، عنوان و نوع عنوان پیاده‌سازی شده‌اند و تابع‌های Relation_getalldata و Relation_getallschema وrelation_getalltype برای برگرداندن

این مقدارها پیاده‌سازی شده‌اند. برای مقدار دهی تک تک عنصرها در رابطه تابع‌های Relation_setsingledata، Relation_setsingleschema و Relation_setsingletype پیاده‌سازی شده‌اند و برای بازیابی مقدارهای درون رابطه Relation_getsingledata وRelation_getsingleschema و Relation_getsingletype پیاده‌سازی شده است. تابع‌های Relation_setcolumn وRelation_setrow وRelation_getcolumn وRelation_getrow برای مقداردهی و بازیابی تعداد سطر و ستون‌های رابطه پیاده‌سازی شده‌اند.

جدول نماد(شیء value_table)

این شیء برای نگهداری جدولِ نماد پیاده‌سازی شده است. تابع سازنده‌ی این شیء تعداد خانه‌های جدولِ نماد را به عنوان آرگومان ورودی دریافت می‌کند. تابع‌های VT_setname و VT_setrelation به ترتیب برای گرفتن نام رابطه و مقدار رابطه و تابع‌های VT_getname و VT_getrelation نیز برای برگرداندن این مقدارها پیاده‌سازی شده‌اند.

شیء Table

این شیء برای نمایش رابطه‌ها به صورتِ جدول بر روی مرورگر برپایه‌ی [28] پیاده‌سازی شده است. مقدارهای ورودی تابعِ سازنده‌ی این شیء تعداد سطرها و ستون‌های جدول است. مقدار Table.border برای تنظیمِ ضخامت کادرِ جدول و مقدار Table.cellspace برای تنظیمِ فاصله‌ی میان خانه‌های جدول به کار می‌روند. تابع Table_getValue برای برگرداندنِ مقدارِ داده‌ی سطر و ستونِ ویژه‌ای پیاده سازی شده است. تابع Table_setValue برای مقدار دهی خانه‌ی ویژه‌ای پیاده سازی شده است. تابع Table_set برای مقداردهی همه‌ی خانه‌ها به صورت یکجا پیاده سازی شده است. تابع Table_isHeader برای این پیاده سازی شده است که اگر خانه‌ی ویژه‌ای جزء عنوان رابطه باشد، مقدار true را برگرداند. تابع Table_write برای کشیدنِ رابطه به صورت جدول بر روی مروگر پیاده‌سازی شده است.

شیء ColoredTable

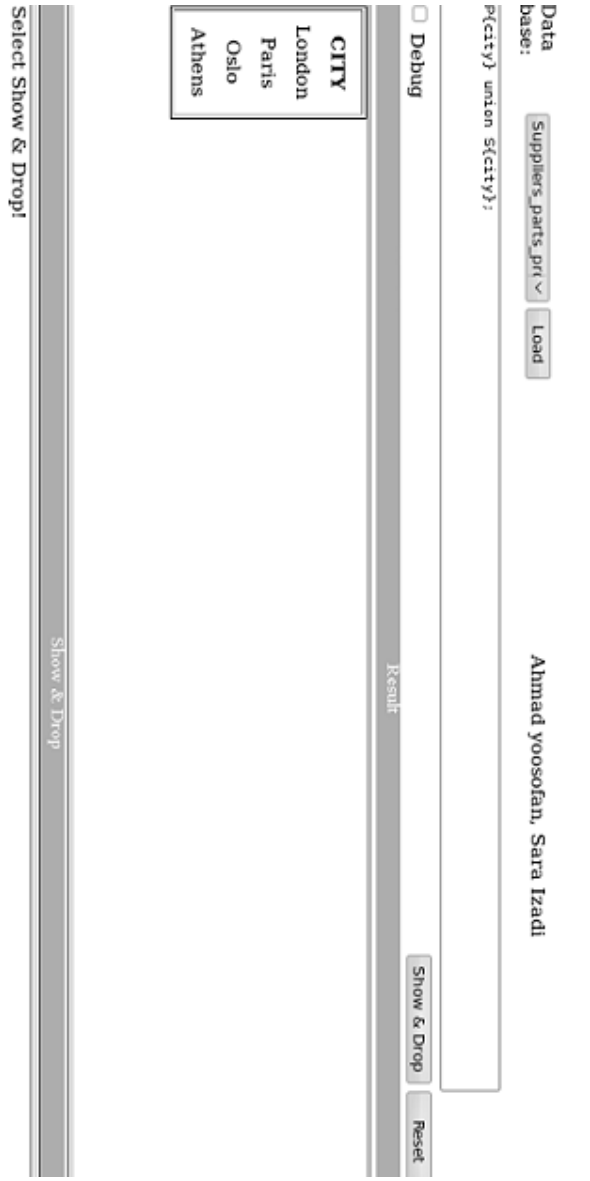
این شیء گسترش یافته‌ی شیء Table است که امکان تعیین طول هر خانه، رنگ پس زمینه‌ی جدول، رنگ مرز جدول و عنوان جدول را فراهم می‌کند. تابع سازنده‌ی تعداد سطرها و ستون‌ها و همچنین رنگ پس زمینه‌ی جدول را به عنوان آرگومان ورودی دریافت می کند. تابع Table_colorWrite بازنویسی تابع Table_write است که ویژگی‌های گفته شده را نیز در نظر می گیرد و جدول را رسم می‌کند. تابع Table_colorsave مقدار رشته‌ای جدول تولید شده (تگ‌های html و داده‌های آن)را برمی‌گرداند ولی جدول را رسم نمی کند .تابع Table_setwidth برای مقداردهی طول خانه‌ها و تابع Table_setcaption برای تعیین عنوان جدول پیاده‌سازی شده است.

11 – صفحه اصلی مفسر

صفحه‌ی اصلی این مفسر و نتیجه‌ی اجرای دستور

S{city} union P{city}

در شکل ۱ نشان داده شده است.



شکل 1: صفحه‌ی اصلی مفسر و محیط توسعه

این صفحه دارای چند بخشِ اصلی زیر است.

۱ـ بخش برگزیدنِ پایگاه داده: در این مفسر تعدادی از پایگاه داده‌های متداول که در کتاب‌های آموزشی زیاد به کار برده شده است به صورت داخلی پیاده‌سازی شده است. کاربر می‌تواند یکی از این پایگاه داده‌های آماده را برگزیند و دستورهای جبر رابطه‌ای را بر روی آن آزمایش کند. دکمه‌ی Load برای بار کردن پایگاه داده‌ی برگزیده شده گذاشته شده است.

۲ـ یک textarea برای گرفتن پرس و جوهای کاربر گذاشته شده

است. کاربر می‌تواند چند دستور جبر رابطه‌ای را درون آن بنویسد و میان آن‌ها ; بگذارد.

۳ـ دکمه‌ی Run : برای اجرای پرس و جوی داده شده بر روی پایگاه داده به کار برده می‌شود. اگر پرس و جوی داده شده خطایی نداشته باشد آن‌گاه نتیجه به صورت یک یا چند جدول در کادر پایین صفحه نمایش داده می‌شود.

۴ـ جعبه‌ی علامت Debug: اگر این جعبه را کاربر علامت بزند آن‌گاه نتیجه‌ی اجرای هر عملِ درونیِ پرس و جو به صورت جدول نشان داده می‌شود. برای نمونه برای پرس و جوی { S{city} union P{city } در حالت اجرا به صورت debug سه جدول نشان داده می‌شود یک جدولِ نتیجه‌ی P{city} و جدول S{city} و در پایان جدول اجتماع دو جدول پیشین در جدول دیگری نشان داده می‌شود. در حالی که اگر این جعبه را کاربر علامت نزده باشد فقط جدولِ نتیجه‌ی نهایی نشان داده می‌شود. در شکل ۲ نتیجه‌ی اجرای این دستور به صورت debug نشان داده شده است.

11 – نتیجه گیری

زبان جبر رابطه‌ای گرچه یک زبان به نسبت قدیمی می‌باشد ولی بسیاری از کتاب‌های مرجع در زمینه‌ی آموزش مبانی پایگاه داده‌ها این زبان را برای آموزش به کار می‌برند. نیاز به مفسری توانمند و همزمان بسیار ساده و با رابط کاربری شایسته برای به کارگیری این زبان بسیار نیاز است و کوشش‌هایی که تا کنون انجام شده است، هر کدام دارای کمبودهایی است. به کارگیری زبان قابل حمل javascript، سکوی اجرایی و بصری تحت وب، سرعت اجرای به نسبت خوب، نیاز نداشتن به حافظه‌ی زیاد، نیاز نداشتن به نصب یا تنظیم‌های ویژه و امکان اجرا بر روی هر سیستم عاملی و بر روی سخت افزارهای گوناگون که فقط دربردارنده‌ی مرورگری سازگار با استانداردهای وب باشند، از هدف‌های ساخت این مفسر و محیط اجرای یکپارچه بود. با وجود دگرگونی‌ها و دشواری‌هایی که در طولِ ساختِ این ابزار به دلیل‌های گوناگون پیش آمد، ابزارِ ساخته شده همه‌ی توانایی‌های دلخواه و مورد نظر را داراست و ابزار بسیار شایسته‌ای برای آموزش جبر رابطه‌ای در درس پایگاه داده‌ها می‌باشد. البته این ابزار نیاز به گسترش‌ها و دگرگونی‌هایی دارد که در نسخه‌های بعدی این ابزار در آن پیش‌بینی خواهد شد.

شکل 2: نتیجه‌ی اجرای دستور در حالت debug

مراجع

[1] E. Codd, “A relational model of data for large shared data banks,” Commun. ACM, vol. 13, Jun. 1970, pp. 387, 377.

[2] R. Ramakrishnan and J. Gehrke, Database Management Systems, McGraw Hill Higher Education, 2002.

[3] C. Date, An Introduction to Database Systems, Addison-Wesley Longman Publishing Co., Inc., 2003.

[4] C. Date, Date on database : writings 2000 - 2006, Berkeley, Calif: Apress, 2006.

[5] J. Ullman and J. Widom, A First Course in Database Systems, Prentice Hall PTR, 2001.

[6] A. Silberschatz, H. Korth, and S. Sudarshan, Database System Concepts, McGraw-Hill Science/Engineering/Math, 2005.

[7] H. Garcia-Molina, J. Ullman, and J. Widom, Database Systems: The Complete Book, Pearson Education (US), 2008.

[8] J. Yang, “Using the ra Relational Algebra Interpreter,” http://www.cs.duke.edu/~junyang/.

[9] G. Gaughan, “Dee, makes Python relational ,” http://www.quicksort.co.uk, 2009.

[10] R. Hartmann, “Duro is a relational open-source database management system,” http://duro.sourceforge.net.

[11] R. Leyton, “LEAP is a relational database management system (RDBMS),” http://leap.sourceforge.net, 2005.

[12] D.Voorhis, “Rel: An Implementation of Date and Darwen's Tutorial D database language,” http://dbappbuilder.sourceforge.net/Rel.php .

[13] S. Tomaselli, “Relational is an educational tool for relational algebra,” http://galileo.dmi.unict.it/wiki/relational/, 2009.

[14] D. Hugh and C. Date, “List of Projects of the third manifesto,” http://www.dcs.warwick.ac.uk/~hugh/TTM/Projects.html, 2009.

[15] H. Darwen and C. Date, “The Third Manifesto,” http://www.thethirdmanifesto.com, 2009.

[16] J. Resig, Pro JavaScript techniques, Berkeley, Calif.; New York: Apress ; Distributed to the book trade worldwide by Springer-Verlag New York, 2006.

[17] P. Wilton and J. McPeak, Beginning JavaScript., Indianapolis, Ind.: Wiley Publishing, Inc., 2007.

[18] E. Woychowsky, Ajax : creating Web pages with asynchronous JavaScript and XML, Upper Saddle River, NJ: Prentice Hall, 2007.

[19] A. Aho, M. Lam, R. Sethi, and J. Ullman, Compilers: Principles, Techniques, and Tools, Addison Wesley, 2006.

[20] D. Grune, H. Bal, C. Jacobs, and K. Langendoen, Modern Compiler Design, Wiley, 2000.

[21] R. Mak, Writing compilers and interpreters, New York: Wiley Computer Pub., 1996.

[22] R. Mak, Writing compilers and interpreters : An applied approach using C++., New York: John Wiley & Sons, 1996.

[23] Writing Compilers and Interpreters Using Java., John Wiley & Sons Inc, 2009.

[24] A. Aho, R. Sethi, and J. Ullman, Compilers: principles, techniques, and tools, Addison-Wesley Longman Publishing Co., Inc., 1986.

[25] J. Meyer and Santillan, “JS/CC Parser Generator Project Homepage,” http://jscc.jmksf.com/.

[26] mozilla, “SpiderMonkey is the code-name for the Mozilla's C implementation of JavaScript,” http://www.mozilla.org/js/spidermonkey/ .

[27] mozilla, “Rhino is an open-source implementation of JavaScript written entirely in Java,” http://www.mozilla.org/rhino/.

[28] عین اله جعفرنژاد قمی، «آموزش گام به گام جاوااسکریپت»، بابل:علوم رایانه، ۱۳۸۴.