

SQL Basics for RPG Developers

Chris Adair

Manager of Application Development – National Envelope
Vice President/Treasurer – Metro Midrange Systems Assoc.

SQL HISTORY

- Structured English Query Language (SEQUEL)
- IBM - Donald D. Chamberlin and Raymond F. Boyce in the early 1970's
- Based on Dr. E F COBB's relational model published in 1970
- Relational Software, now Oracle, 1st to have a commercially available SQL base system, Oracle V2 - 1979

SQL on OS/400

- Available on OS/400 V1R1 – 1988
- Poor Performance – Misconception!
Poorly written statements caused the performance issue and the rumors spread like wildfire.
- SQE (SQL Query Engine) – V5R2 (2002)
Major changes to improve performance

SQL: Why?

- Great for selecting/updating groups of data
- Columnar functions allow for field manipulation at record selection
- Aggregate columns in like rows for summarizing data
- IBM continues to enhance the SQL engine
- Let the engine pick the best logical view to process

Most Common Uses for System i Developers

- Interactive SQL
 - STRSQL
 - Quick ad-hoc queries
 - Remote Database Connectivity
- Embedded SQL
 - Alternative to native file I/O
 - Allows for SQL functionality in RPG
 - Remote Database Connectivity

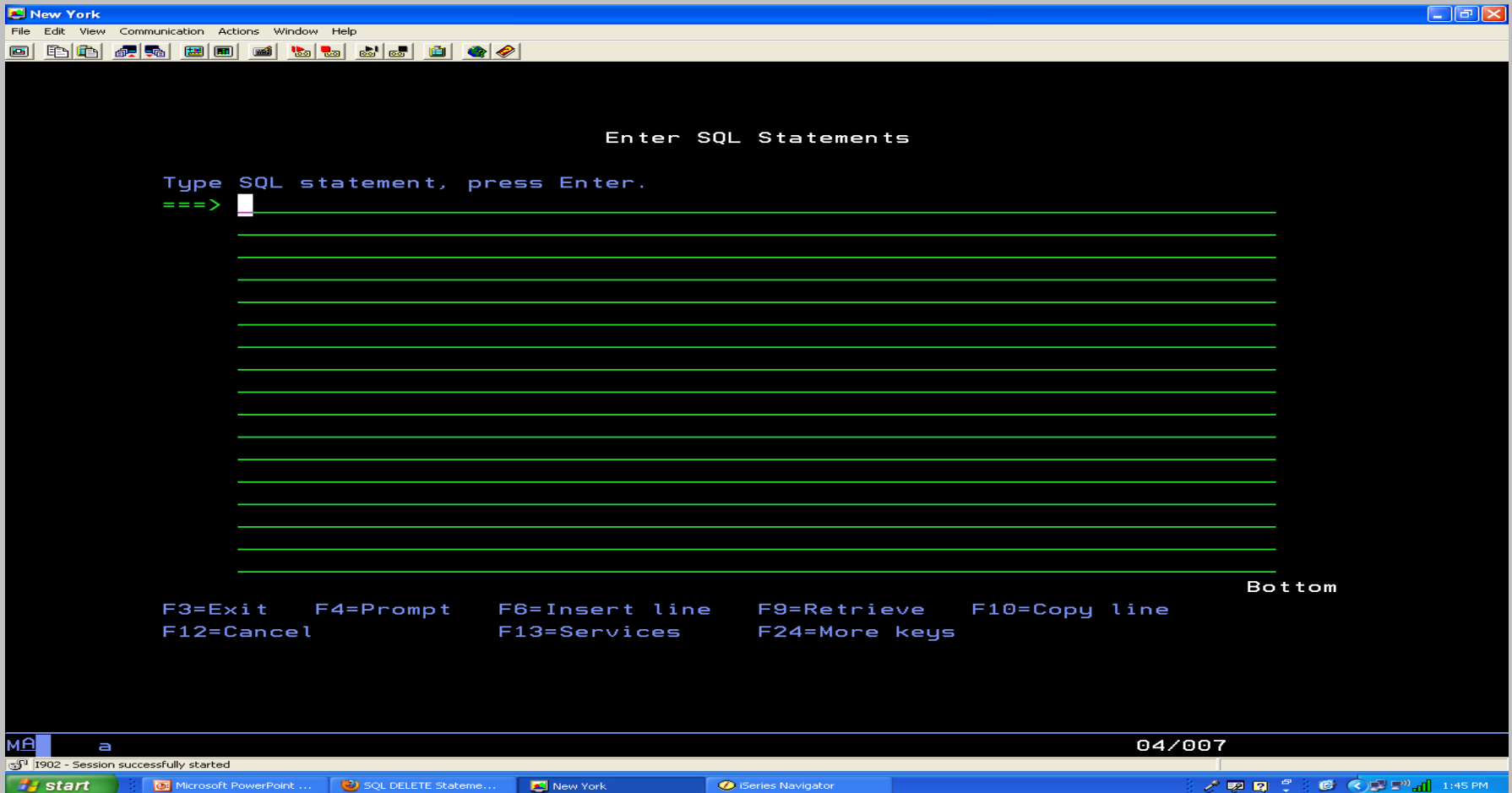
Interactive SQL

Interactive SQL

- STRSQL (Green Screen)
- System i Navigator
- Most Common SQL Statements
 - The SELECT statement is used to select data from a database. The result is stored in a result table, called the result-set.
 - The UPDATE statement is used to update existing records in a table.
 - The DELETE statement is used to delete rows in a table.

Interactive SQL

- From a command line, STRSQL



Interactive SQL

- Options – F13, option 1, Change Session Attributes

```
New York
File Edit View Communication Actions Window Help
Change Session Attributes

Type choices, press Enter.

Statement processing . . . . . *RUN          *RUN, *VLD, *SYN
SELECT output . . . . . 1             1=Display, 2=Printer
                                           3=File
Commitment control . . . . . *NONE      *NONE, *CHG, *CS, *ALL, *RR
                                           *NC, *UR, *RS
Date format . . . . . *MDY        *JOB, *USA, *ISO, *EUR, *JIS
                                           *MDY, *DMY, *YMD, *JUL
Date separator . . . . . //          *JOB, '/', ' ', ' ', ' ', ' '
                                           ' ', *BLANK
Time format . . . . . *HMS          *HMS, *USA, *ISO
                                           *EUR, *JIS
Time separator . . . . . :.         *JOB, ':', ' ', ' ', ' ', ' '
                                           ' ', *BLANK
Data refresh . . . . . *ALWAYS     *ALWAYS, *FORWARD
Allow copy data . . . . . *YES       *YES, *OPTIMIZE, *NO
Naming convention . . . . . *SYS      *SYS, *SQL

F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  More...

05/037
1:51 PM
```

Interactive SQL - Select

- SQL SELECT Syntax
 - SELECT column_name(s)
FROM table_name
 - SELECT * FROM table_name
- SQL WHERE clause syntax
 - SELECT column_name(s)
FROM table_name
WHERE column_name operator value

Interactive SQL - Select

- Operators allowed on WHERE clause
 - = Equal
 - <> Not Equal
 - > Greater Than
 - < Less Than
 - >= Greater than or Equal
 - <= Less than or Equal
 - Between - Between an inclusive range
 - Like – Search for a pattern
 - IN – If you know the exact values you want to return

Interactive SQL - Select

- `Select * From F0116`
`Where ALCTY1 in('DALLAS','FT WORTH')`
- `Select * From F0116`
`Where ALAN8 in(112109,112117)`

Remember: Quotes around alpha fields are mandatory.

Interactive SQL - Select

```
New York
File Edit View Communication Actions Window Help
[Icons]

Display Data

Position to line . . . . . _____
Data width . . . . . : 350
Shift to column . . . . . _____
...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9...+...10...+...11...+...12...+...13.
Address   Begin   Eff   Address   Address   Address
Number   Date   Date   Line 1    Line 2    Line 3
-----
112109   @    1    Upper Allen Business Park   435 Independence Ave
112117   @    1    250 Boot Road
***** End of data *****

F3=Exit   F12=Cancel   F19=Left   F20=Right   F21=Split   F22=Width 80
First column of data.

Bottom

ME a 03/032
I902 - Session successfully started
start Microsoft PowerPoint ... SQL WHERE Clause - ... New York ISeries Navigator 2:26 PM
```

Interactive SQL - Select

- System i Navigator

The screenshot displays the iSeries Navigator application window. The title bar reads "iSeries Navigator" and includes standard window controls. Below the title bar is a menu bar with "File", "Edit", "View", and "Help". A toolbar with various icons is located below the menu bar. The main area is divided into several panes:

- Environment: My Connections:** A tree view showing a hierarchy of connections. The selected connection is "Neas400prd.nationalenvelope.com". Underneath, there are folders for "Basic Operations", "Work Management", "Configuration and Service", "Network", "Integrated Server Administration", "Security", "Users and Groups", "Databases", "File Systems", "Backup", "Application Development", and "AFP Manager". The "Databases" folder is expanded to show "S1030182", which contains "Schemas", "Database Navigator Maps", "SQL Performance Monitors", "SQL Plan Cache Snapshots", and "Transactions".
- Neas400prd.nationalenvelope.com: S1030182 Database: S1030182:** A table listing database objects. The table has two columns: "Name" and "Text".

Name	Text
Schemas	Work with DB2 UDB for iSeries objects.
Database Navigator Maps	Work with Database Navigator maps.
SQL Performance Monitors	Work with SQL performance monitors.
SQL Plan Cache Snapshots	Work with the system query plan cache entries.
Transactions	Work with transactions.

At the bottom of the window, there is a "My Tasks" pane with the following tasks:

- Add a connection
- Install additional components
- Select schemas to display
- Run an SQL script
- Map your database
- Create a new SQL Performance Monitor
- Import data into a table.
- Export data from a table or view.
- Help for related tasks

The status bar at the bottom of the application shows "1 - 5 of 5 objects". The Windows taskbar at the very bottom shows the Start button, several open applications (Microsoft PowerPoint, SQL WHERE Clause, New York), and the iSeries Navigator application. The system clock shows "2:33 PM".

Interactive SQL - Select

The screenshot shows a window titled "Untitled - Run SQL Scripts - Neas400prd.nationalenvelope.com(S1030182)". The menu bar includes "File", "Edit", "View", "Run", "VisualExplain", "Monitor", "Options", "Connection", and "Help". The toolbar contains various icons for file operations and execution. The main area is labeled "Examples" and contains the following SQL query:

```
Select * From NECPRDDTA.F0116  
Where ALAN8 in(112109,112117)
```

Below the query, a table displays the results of the query. The table has five columns: ALAN8, ALEFTB, ALEFTF, ALADD1, ALADD2, and ALADD3. The data is as follows:

ALAN8	ALEFTB	ALEFTF	ALADD1	ALADD2	ALADD3
112109		0 F1	Upper Allen Business Park	435 Independence Ave	
112117		0 F1	250 Boot Road		

At the bottom of the window, a "Messages" pane shows the executed query: "Select * From NECPRDDTA.F0116 Where ALAN8 in(112109,112117)". The Windows taskbar at the bottom shows the Start button and several open applications: Microsoft PowerPoint, SQL WHERE Clause, New York, Series Navigator, and the current application window.

Interactive SQL - Select

The screenshot shows a window titled "Untitled - Run SQL Scripts - Neas400prd.nationalenvelope.com(51030182)". The window contains a menu bar (File, Edit, View, Run, VisualExplain, Monitor, Options, Connection, Help) and a toolbar. Below the toolbar is a text area with the following SQL query:

```
Select * From NECPRDDTA.F0116  
Where ALAN8 in(112109,112117)
```

A "Save" dialog box is open in the foreground, showing the "Client Access" folder. The "File name" field contains "SampleF0116" and the "Files of type" dropdown is set to "SQL files (.sql)".

Below the SQL query, there are two tables. The left table has the following data:

ALAN8	ALEFTB	ALEFTF	ALADD
112109		0 F1	Upper A
112117		0 F1	250 Bo

The right table has the following data:

ALADD3

At the bottom of the window, a status bar displays the message: "Messages: Select * From NECPRDDTA.F0116 Where ALAN8 in(112109,112117)".

Interactive SQL - Select

The screenshot displays the IBM Client Access Interactive SQL interface. The main window title is "C:\Program Files\IBM\Client Access\SampleF0116.sql - Run SQL Scripts - Neas400prd.nationalenvelope.com(S1030182)". The menu bar includes File, Edit, View, Run, Visual Explain, Monitor, Options, Connection, and Help. The main area contains the following SQL query:

```
Select * From NECPRDDTA.F0116  
Where ALAN8 in(112109,112117)
```

An "Open" dialog box is overlaid on the main window, showing the "Client Access" directory. The "Files of type" is set to "SQL files (*.sql)". The "File name" field is empty. The "Open" and "Cancel" buttons are visible.

Below the main window, a data table is visible with the following columns and data:

ALAN8	ALEFTB	ALEFTF	ALADD
112109		0 F1	Upper A
112117		0 F1	250 Bo

The status bar at the bottom shows the current query: "Messages: Select * From NECPRDDTA.F0116 Where ALAN8 in(112109,112117)". The Windows taskbar at the bottom shows the Start button and several open applications: Microsoft PowerPoint, SQL WHERE Clause, New York, I Series Navigator, and C:\Program Files\IBM...

Interactive SQL - Update

- SQL UPDATE Syntax
- UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
- **BE CAREFUL WHEN UPDATING RECORDS!**
 - If you omit the WHERE, all records get updated. May be a good idea to backup your data to a SAVF prior to execution. Also, I make it a practice to get my WHERE clause set in a SELECT statement, then copy/paste to the UPDATE statement.

Interactive SQL - UPDATE

- All rows Updated

```
UPDATE NECCRPDTA/F3002  
SET IXQNTY = ixqnty * 10
```

- Selective Update

```
UPDATE NECPRDDTA/F0101  
SET ABAC02 = 'I05'  
WHERE abac02 = 'D04'
```

INTERACTIVE SQL - UPDATE

- Use a sub-select to update a column with data from another file
- update stlrenum/fstlrenum
set rnaitm = (select imaitm from F4101
 where rnuitm = imlitm)
where rnuitm in(select imlitm from f4101
 where rnuitm = imlitm)
- Can be confusing, but **POWERFUL!**

Interactive SQL - Update

- Like the SELECT, UPDATE can be run/saved in System i Navigator.
- For updates, **BACKUP THE FILE!**
- Be sure you know the Commitment Control

Interactive SQL - DELETE

- SQL DELETE Syntax

DELETE FROM table_name

WHERE some_column=some_value

- BE CAREFUL WHEN DELETING RECORDS!

– Like the UPDATE statement, if you omit the WHERE clause, all records get deleted.
BACK IT UP!

Interactive SQL - DELETE

- DELETE From F42119
WHERE SDTRDJ < 110000

Remote Databases

Remote Databases

- Use the CONNECT Statement to SQL other i5 OS partitions or any DB2 database supporting DRDA (Distributed Relational Database Architecture)
- What about Microsoft SQL Server or Oracle

Not 100% compliant with DRDA standard.

WebSphere Federation Server allows for connectivity to these databases.

Remote Databases

- For i5 OS Partitions
 - Add remote system to Relational Database Directory (WRKRDBDIRE)
 - STRSQL
 - Issue a CONNECT statement
 - Does require credentials
 - VIOLA! – You can now query the remote DB2 Database.

Remote Databases

```
New York
File Edit View Communication Actions Window Help
Work with Relational Database Directory Entries

Position to . . . . .

Type options, press Enter.
1=Add 2=Change 4=Remove 5=Display details 6=Print details

Option  Entry                Remote Location                Text
-----  -----
-       DB2PDTCP                    192.168.230.17                 Frisco 570 - Informix/ >
-       DB2QATCP                    192.168.230.18                 NY 570 - Baan QA
-       MX_MIMIX_PKE                MXPROD                         MIMIX MAINTAINED ENTRY
-       S1030182                    *LOCAL                         Frisco 570 - NY
-       S65F0B9D                    S65F0B9D

F3=Exit  F5=Refresh  F6=Print list  F12=Cancel  F22=Display entire field
(C) COPYRIGHT IBM CORP. 1980, 2009.

Bottom

MA a 11/004
1902 - Session successfully started
```


Embedded SQL

Types of Embedded SQL

- Static SQL
 - The simplest form of embedding SQL in RPG
 - Essentially, the SQL statement is hard coded in your program
- Dynamic SQL
 - The SQL statement is assembled at run-time
 - Requires more resource at run-time for preparing the statement
 - Can become very sophisticated

Variables

- Most variables defined in RPG can be used in a SQL statement
- Variables are preceded by a colon, ‘ : ‘.
- Some non-supported variables include
 - UDATE, UDAY, UMONTH, UYEAR
 - Pointer
 - Tables
 - Named Constants

Source Members

- New Source Types
 - SQLRPG
 - SQLRPGLE
 - Note: RPG II and Auto Report does not support SQL

Static SQL

Getting Started

- Lets create a new SQLRPGLE Module –
Get_ALPH
- We'll pass a Customer Number
- Create a Static SQL That Will Query the
Address Book
- Return the Name

Get_ALPH

```
Remote System Explorer - GET_ALPH.SQLRPGLE - Eclipse Platform
File Edit Source Compile Navigate Search Project Run Window Help
GET_ALPH.SQLRPGLE x SAMPLEA.RPGLE
Line 24      Column 20      Replace
.....PName+++++++..B.....Keywords+++++++
000100      HNOMAIN
000200      DGet_ALPH      PR      30
000300      DAN8      8 0
000400
000500      PGet_ALPH      B      EXPORT
000600
000700      DGet_ALPH      PI      30
000800      DAN8      8 0
000900
001000      DALPH      S      30
001100      DW_ALPH      S      30
001200
001300      C/Exec SQL
001400      C+ Select ABALPH Into :W_ALPH from F0101 Where ABAN8 = :AN8
001500      C/End-Exec
001600
001700      C      If      SQLCOD = *Zero
001800      C      Eval      ALPH = W_ALPH
001900      C      Else
002000      C      Eval      ALPH = 'Not Found In Master'
002100      C      ENDIF
002200
002300      C      Return      ALPH
002400      PGet_ALPH      E
```

Let's Make This A Service Program

- Compile Get_ALPH with CRTRPGSQLI
 - Opt. 15 From SEU
- Create the Service Program – SP_F0101
 - CRTSRVPGM
 - Specify Get_Alph as a module

Let's Give it a Test Ride

- First, We'll need a sample program to call our new module that's in the service program.
- Let's name that code SampleA

SampleA Program

Remote System Explorer - SAMPLEA.RPGLE - Eclipse Platform

File Edit Source Compile Navigate Search Project Run Window Help

GET_ALPH.SQLRPGLE SAMPLEA.RPGLE X

Line 1	Column 6	Replace
.....	DName+++++++ETDsFrom+++To/L+++IDc.Keywords+++++++	
000100	DGet_ALPH	PR 30
000200	DAN8	8 0
000300		
000400	DABAN8	S 8 0
000500	DNAME	S 30
000600		
000700	C	EVAL ABAN8 = 112109
000800		
000900	C	EVAL NAME = GET_ALPH(ABAN8)
001000		
001100	C	EVAL *INLR = *ON

Let's run that in GUI Debug

Running in GUI Debugger

The screenshot displays the iSeries System Debug Manager application. The main window has a menu bar (File, Edit, Debug, Help) and a toolbar. Below is a table with columns for Debug, System, User, and Status.

Debug	System	User	Status
<input checked="" type="checkbox"/>	192.168.230.10	Cadair	ready
<input checked="" type="checkbox"/>	192.168.230.13	Cadair	ready

A 'Start Debug' dialog box is open, showing the following configuration:

- System: 192.168.230.10
- User: Cadair
- Select debug method:
 - Submit and debug program in batch job
 - Debug existing job on system
 - Debug OS/400 PASE
- Program to debug:
 - Type: Program
 - Name: Samplea
 - Library: Chrislib
 - Parameters: (empty)
 - Initialization command: (empty)
 - Standard I/O: Send to server spool file
 - Classpath: (empty)
 - Host source path: (empty)
 - Client source path: (empty)
- Existing job to debug:
 - Job name: C1
 - User: Cadair
 - Job number: 137127

Buttons at the bottom of the dialog are OK, Cancel, and Help.

A Messages window in the bottom-left corner shows the following log:

```
<2/23/07 4:26 PM> Attempting connect
<2/23/07 4:26 PM> Attempting connect
<2/23/07 4:26 PM> Connected success
<2/23/07 4:26 PM> Connected success
<2/23/07 4:27 PM> starting iSeries
```

The Windows taskbar at the bottom shows the Start button and several open applications, including iSeries System Debug Manager.

Specified Break Points

The screenshot displays the iSeries System Debugger interface. The main window shows the source code of the program `Samplea.pgm` with several break points specified. The break points are:

- Line 1: `DGet_ALPH` (PR) at column 30
- Line 2: `DAN8` (S) at column 8
- Line 3: `DABAN8` (S) at column 8
- Line 4: `DNAME` (S) at column 30
- Line 7: `C` (EVAL) with condition `ABAN8 = 112109`
- Line 9: `C` (EVAL) with condition `NAME = GET_ALPH(ABAN8)`
- Line 11: `C` (EVAL) with condition `*INLR = *ON`

The debugger also shows a Call Stack window at the bottom right, which lists the following stack frames:

Method	Module	Line	Statement	Program	Library
		459		Qt3reqio	Qsys
		1584		Qwsget	Qsys
		2283		Quylist	Qsys
		112		Quomain	Qpda
		1910		Quocpp	Qpda
		297		Qcmdexc	Qsys
		200		J00cmda	Jdfobj
		759		F00menu	Jdfobj
		211		J98initv	Jdfobj
		231		J98inita	Jdfobj
		1174		Qcmd	Qsys

The taskbar at the bottom shows the system time as 5:51 PM.

Call from Command Line

New York

File Edit View Communication Actions Window Help

Work with Members Using PDM S65F0B9D

File QRPGSRC
Library CHRISLIB Position to _____

Type options, press Enter.
2=Edit 3=Copy 4=Delete 5=Display 6=Print 7=Rename
8=Display description 9=Save 13=Change text 14=Compile 15=Create module...

Opt	Member	Type	Text
___	SAI450S_WH	SQLRPGLE	Create shipments trans. by Acct, order and item
___	SAMPLEA	RPGLE	Retrieve ABALH from Service Program - Get-Alph
___	TEMPLATE	SQLRPGLE	Template to convert MCU
___	TEMPLATE2	TXT	Template to convert MCU
___	UPCCHK	RPGLE	F4101 vs IT400P F/G Item Imbalance
___	UPCCHK2	RPGLE	F4101 vs IT400P F/G Item Imbalance
___	VS_CREDITS	SQLRPGLE	COs keyed in 2005 for SOs keyed in 2004

Bottom

Parameters or command
==> call chrislib/samplea

F3=Exit F4=Prompt F5=Refresh F6=Create
F9=Retrieve F10=Command entry F23=More options F24=More keys

MA c 21/007

EPSON Stylus CX3800 Series on USB001

1902 - Session successfully started

Start In... Da... Da... N... M... Pa... D... E... Re... C:... iSe... Ne... iSe... 5:52 PM

GUI Debugger

The screenshot displays the iSeries System Debugger interface. The main window shows the source code for the program `Samplea.pgm`. The current execution point is at line 7, which is highlighted in yellow. The code includes several variables and statements:

```
1 DGet_ALPH PR 30
2 DAN8 8 0
3
4 DABAN8 S 8 0
5 DNAME S 30
6
7 C EVAL ABAN8 = 112109
8
9 C EVAL NAME = GET_ALPH(ABAN8)
10
11 C EVAL *INLR = *0N
```

The **Locals** window shows the following variables:

Name	Value
*IN	
ABAN8	00000000.
NAME	"

The **Call Stack** window shows the current call stack:

Method	Module	Line	Statement	Program	Library
SAMPLEA	Samplea	7		Samplea	Chrislib
_QRNP_PEP_SAMPLEA	Samplea	481		Samplea	Chrislib
		481		Quocmd	Qsys
		253		Quomain	Qpda
		1910		Quocpp	Qpda
		297		Qcmdexc	Qsys
		200		J00cmda	Jdfobj
		759		P00menu	Jdfobj
		211		J98initv	Jdfobj
		231		J98inita	Jdfobj
		1174		Qcmd	Qsys

The status bar at the bottom indicates the current breakpoint is at line 7, and the system clock shows 6:08 PM.

Programs Breakpoints

/Qsys.lib/Chrislib.lib/Samplea.pgm/Samplea

```

1   DGet_ALPH      PR          30
2   DAN8           8  0
3
4   DABAN8         S          8  0
5   DNAME          S          30
6
7   C              EVAL      ABAN8 = 112109
8
9   C              EVAL      NAME = GET_ALPH(ABAN8)
10
11  C              EVAL      *INLR = *U
    
```

Variable: NAME = 'RIS Paper Co-Mechanicsburg'

Locals Monitors Console

Name	Value
*IN	
ABAN8	00112109.
NAME	'RIS Paper Co-Mechanicsburg'

Call Stack Threads Memory Standard I/O

Method	Module	Line	Statement	Program	Library
SAMPLEA	Samplea	11	11	Samplea	Chrislib
_QRNP_PEP_SAMPLEA	Samplea		481	Samplea	Chrislib
			481	Quocmd	Qsys
			253	Quomain	Qpda
			1910	Quocpp	Qpda
			297	Qcmdexc	Qsys
			200	J00cmda	Jdfobj
			759	P00menu	Jdfobj
			211	J98initv	Jdfobj
			231	J98inita	Jdfobj
			1174	Qcmd	Qsys

Variable: NAME = 'RIS Paper Co-Mechanicsburg'

What If I Need To Return Multiple Rows

- Result Sets – The multiple rows returned by a `SELECT` statement
- Think of it as a temporary work file in memory
- Access to each individual row is achieved by the SQL cursor.
- The cursor is a pointer that keeps track of which row you're working with

More on Cursors

- Working with the cursor requires several new SQL statements.
 - Declare – Analyzes the Select statement and names the buffer area the result set is sent to
 - Open – After the Declare, the Open statement actually runs the SQL and returns the result set
 - Fetch – Reads the row that the cursor is on and advances the cursor
 - Close – Essentially, removes the cursor and result set.

Lets Take a Look at Some Cursors In Action

- P591801 – This module is used to report the daily bookings from the NY JDE system
- We pass in some parameters, query the F4211 and F42119
- Process the data
- Return the Daily Bookings

The Declare and Open

```
004500 C
004600 C/EXEC SQL
004700 C+ DECLARE EntOrd01 Cursor For
004800 C+     SELECT SDTRDJ, SDSOQS, SDSOCN, SDAEXP, SDITM, SDUOM
004900 C+     FROM NECPRDDTA/F4211,NECPRDDTA/F0101
005000 C+     WHERE SDTRDJ > :Date and
005001 C+     SDDCTO NOT IN('ST','SB','SI','CO') and
005002 C+     SDLNTY NOT IN('F ','FA','MI','MC','SW') and
005100 C+     SDAN8 = ABAN8 and
005101 C+     ABAC05 <> 'IC' and
005102 C+     SDMCU = :KOMCU
005200 C+     ORDER BY SDDOCO
005300 C/END-Exec
005400 C/Exec SQL
005500 C+ OPEN EntOrd01
005600 C/End-Exec
005700
```

The Fetch

```
005700
005800      C              DOW      SQLCOD = *ZERO
005900
006000      C/Exec SQL
006100      C+ FETCH FROM EntOrd01 INTO
006200      C+ :W_SDTRDJ, :W_SDSOQS, :W_SDSOCN, :W_SDAEXP, :W_SDITM, :W_SDUOM
006300      C/End-Exec
006400
006500      C              If      SQLCOD = 0
006600
006700      C      *Jul      Move    W_SDTRDJ      EntDate
006800
006801      C              Eval    UFactor = 1
006802      C              If      W_SDUOM <> 'M '
006803      C              CallP   CVTUOM(KOMCU:W_SDITM:W_SDUOM:UMRUM:UMFactor)
006804      C              ENDIF
006805
006900      C              If      EntDate = Cur_ISO
007000      C              Eval    @$DE = @$DE + (W_SDAEXP * .01)
007100      C              Eval    @QDE = @QDE +
007101      C                      ((W_SDSOQS - W_SDSOCN) * .01) * UFactor)
007200      C              EndIf
007300      C              If      EntDate = Cur_ISO
```

The Close

```
Remote System Explorer - P591801.SQLRPGLE - Eclipse Platform
File Edit Source Compile Navigate Search Project Run Window Help
P591801.SQLRPGLE x
Line 137      Column 78      Replace
.....1.....2.....3.....4.....5.....6.....7.....8.....9.....0
008500      C                      Eval          @$YE = @$YE + (W_SDAEXP * .01)
008600      C                      Eval          @QYE = @QYE +
008601      C                      ((W_SDSOQS - W_SDSOCN) * .01) * UMFactor)
008700      C                      EndIf
010400
010500      C                      EndIf
010600      C                      EndDo
010700
010701      C/Exec SQL
010702      C+ Close EntOrd01
010703      C/End-Exec
010704
010800      C/EXEC SQL
010900      C+ DECLARE EntOrd02 Cursor For
011000      C+      SELECT SDTRDJ, SDSOQS, SDSOCN, SDAEXP, SDITM, SDUOM
011100      C+      FROM NECPRDDTA/F42119,NECPRDDTA/F0101
011200      C+      WHERE SDTRDJ > :Date and
011201      C+      SDDCTO NOT IN('ST','SB','SI','CO') and
011202      C+      SDLNTY NOT IN('F ','FA','MI','MC','SW') and
011203      C+      SDAN8 = ABAN8 and
011204      C+      ABAC05 <> 'IC' and
011205      C+      SDMCU = :KOMCU
011400      C+      ORDER BY SDDOCO
011500      C/END-Exec
011600      C/Exec SQL
011700      C+ OPEN EntOrd02
011800      C/End-Exec
011900
012000      C                      DOW          SQLCOD = *ZERO
012100
012200      C/Exec SQL
012300      C+ FETCH FROM EntOrd02 INTO
012400      C+ :W_SDTRDJ, :W_SDSOQS, :W_SDSOCN, :W_SDAEXP, :W_SDITM, :W_SDUOM
012500      C/End-Exec
012600
012700      C                      If          SQLCOD = 0
```

The Scroll Cursor

- If you have a need to process the result set more than once, you'll need to specify the 'Dynamic Scroll Cursor'
- The Scroll cursor let's you control how the records in the results are read

Example: Dynamic Scroll Cursor

```
004300 C/EXEC SQL
004400 C+ DECLARE Prod01 Dynamic Scroll Cursor For
004500 C+ SELECT PSMO, PSDA, PSYR, PSQTY1, PSQTY2, PSQTY3
004600 C+ FROM OCDTALIB/PRDSUM
004700 C+ WHERE PSYR = :Wrk_YY and PSTYPE = 'D' and
004800 C+ PSMCH# IN('1','2','3','4','5','6','7','8','9','10','11',
004900 C+ '12','14','15','16','17','20','21','22','23','24','25',
005000 C+ '30','32','38','39','40','41','42','44','47','48','49',
005100 C+ '50','51','52','54','56','59','60','61')
005200 C+ ORDER BY PSMO
005300 C/END-Exec
005400 C/Exec SQL
005500 C+ OPEN Prod01
005600 C/End-Exec
005700
005800
005900
006000 C/Exec SQL
006100 C+ FETCH First FROM Prod01 INTO
006200 C+ :W_PSMO, :W_PSDA, :W_PSYR, :W_PSQTY1, :W_PSQTY2, :W_PSQTY3
006300 C/End-Exec
006400
006500 C          DOW          SQLCOD = *ZERO
006600
006700 C          *MDY         Move      W_PSMO      MfgDate
006800
006900
007000
007100
007200
007300
007400
007500
007600
007700
007800
007900
008000
008100
008200
008300
008400
008500 C          Eval      @UY = @UY + ((W_PSQTY1+W_PSQTY2+W_PSQTY3)
008600 C          * .001)
008700 C          EndIf
008800 C/Exec SQL
008900 C+ FETCH Next FROM Prod01 INTO
009000 C+ :W_PSMO, :W_PSDA, :W_PSYR, :W_PSQTY1, :W_PSQTY2, :W_PSQTY3
009100 C/End-Exec
009200
009300 C          EndDo
009400
009500 C/Exec SQL
009600 C+ CLOSE Prod01
009700 C/End-Exec
009800
009900
010000 C          Eval      Production = Amounts
010100 C          Return    Production
010200 PP591804_OC      E
```

Dynamic SQL

- The SQL statement is built within the RPG
- The SQL statement is verified at run time, not compile time
- Takes more overhead during run-time
- Makes an application very dynamic

Let's Take a Look at Dynamic SQL

- JDEdwards P032002 – Customer Ledger Inquiry
- A SQLRPG Program written in 1990
- Based on user input, program either uses a LF or builds a SQL statement using the CAT opcode. Yikes!!!
- A Prepare statement is used to minimize the overhead for repeat executions

P032002 – Building the String

```
..... *..1.....+.....2.....+.....3.....+.....4.....+.....5.....+.....6.....+.....7.....+.....8
048290      C* 20
048300 B1   CSR      PSAN8J      IFNE *BLANK
048310 B2   CSR      $WHERE      IFEQ '1'
048320      CSR      MOVE *BLANKS  $WHERE
048330      CSR      STRING      CAT  'WHERE':1  STRING
048340 X2   CSR      ELSE
048350      CSR      STRING      CAT  'and':1   STRING
048360 E2   CSR      ENDIF
048370      C*
048380      CSR      STRING      CAT  'RPN8J':1 STRING
048390      CSR      STRING      CAT  '=':1    STRING
048400      CSR      STRING      CAT  PSAN8J:1  STRING
048410 E1   CSR      ENDIF
048420      C*
048430      C*      Document number.
048440      C* 40
048450 B1   CSR      $DOC        IFEQ '1'
048460 B2   CSR      $WHERE      IFEQ '1'
048470      CSR      MOVE *BLANKS  $WHERE
048480      CSR      STRING      CAT  'WHERE':1  STRING
048490 X2   CSR      ELSE
048500      CSR      STRING      CAT  'and':1   STRING
048510 E2   CSR      ENDIF
048520      C*
048530      CSR      STRING      CAT  'RDOC':1  STRING
048540      C*
048550 B2   CSR      $#DOC       IFEQ '1'
048560      CSR      MOVE 'between' $BTWN 7
048570      CSR      STRING      CAT  $BTWN:1  STRING
048580 B3   CSR      VDDOC       IFEQ *BLANK
048590      CSR      STRING      CAT  '0':1    STRING
048600 X3   CSR      ELSE
048610      CSR      STRING      CAT  VDDOC:1  STRING
048620 E3   CSR      ENDIF
048630      CSR      STRING      CAT  'and':1   STRING
048640      CSR      STRING      CAT  VD#DOC:1  STRING
048650 X2   CSR      ELSE
048660      CSR      STRING      CAT  '=':1    STRING
```

P032002 – Running the Prepare

```
P032002.SQLRPG x
Line 5216      Column 1      Replace
.....CLON01N02N03Factor1+++OpcdeFactor2+++ResultLenDEHiLoEqComments++++*****
052230 X2     CSR                ELSE
052240       CSR          STRING      CAT 'RPDGJ ':1STRING
052250 E2     CSR                ENDIF
052260       CSR          MOVE *BLANK   §ORDER
052270 E1     CSR                ENDIF
052280       C*
052290       C*-----
052300       C*
052310       C*      Prepare the statement.
052320       C*
052330       C/EXEC SQL
052340       C+      PREPARE RPSELECT FROM :STRING
052350       C/END-EXEC
052360       C*-----
052370       C*
052380       C*      Declare the cursor.
052390       C*
052400       C/EXEC SQL
052410       C+      DECLARE RPCURSOR CURSOR FOR RPSELECT
052420       C/END-EXEC
052430       C*-----
052440       C*
052450       C*      Open the cursor.
052460       C*
052470       C/EXEC SQL
052480       C+      OPEN RPCURSOR
052490       C/END-EXEC
052500       C*
052510       C*-----
052520       C/EXEC SQL
052530       C+      WHENEVER SQLERROR CONTINUE
052540       C/END-EXEC
052550       C*-----
052560       CSR          ENDSQL        ENDSR
052630       CSR          S00CLC       BEGSR
052740       CSR          ENDCLC      ENDSR
052930       CSR          S004        BEGSR
055380       CSR          END004       ENDSR
```

P032002 – ISDB Debug

```
Debug          Goto          Program          Options          Help
NECPRD0BJ/P032002:52330          ISDB/400          More: - + >
052210        CSR          STRING          CAT 'RPDGJ ':1STRING
052220        CSR          STRING          CAT 'DESC ':1STRING
052230 X2      CSR          ELSE
052240        CSR          STRING          CAT 'RPDGJ ':1STRING
052250 E2      CSR          ENDIF
052260        CSR          MOVE *BLANK    $ORDER
052270 E1      CSR          ENDIF
052280        C*
052290        C* -----
052300        C*
052310        C*      Prepare the statement.
052320        C*
052330        C/EXEC SQL
052340        C+    PREPARE RPSELECT FROM :STRING
052350        C/END-EXEC
052360        C* -----
052370        C*
052380        C*      Declare the cursor.
ISDB ===>
F3=Exit      F5=Step     F6=Break    F11=Display variable
F12=Cancel   F17=Run     F23=Change variable  F24=More keys
```

Display Program Variables

Job . . . : F1 User . . . : CADAIR Number . . . : 158044

Program : P032002

Recursion level : 1

Start position : 1

Format : *CHAR

Length : *DCL

Variable : STRING

 Type : CHARACTER

 Length : 475

 *...+...1...+...2...+...3...+...4...+...5

'SELECT * FROM F0311SQL WHERE RPAN8 = 00112109 ORDE'

'R BY RPAN8 , RPP0'

'

'

Press Enter to continue.

F3=Exit F12=Cancel

Columnar Functions

Some SQL Columnar Functions

- SUM – This function returns the total of the supplied field values within the data set
- AVG – This function returns an average of the field.
- COUNT – How many rows meet the selection criteria
- Min or Max – Returns Min or Max

Some Examples

- Select SDAN8, SUM(SDAEXP) as Total
From F42119
Where SDIVD > 107000
Group BY SDAN8
Order By Total Desc

Display Data

Data width : 31
Shift to column _____

Position to line █
.....1.....2.....3.

Address TOTAL
Number

88831	41,754.59
64	31,430.65
203641	19,936.80
27	14,595.08
173446	10,854.89
168920	10,824.85
114841	9,638.40
123211	8,537.48
106403	8,509.34
173462	7,324.80
90510	7,270.96
117129	6,017.50
193172	5,828.20
234000	3,969.00
182060	3,712.50
147521	3,200.00
22	3,097.48

More...

F3=Exit F12=Cancel F19=Left F20=Right F21=Split F22=Width 80

Things to Watch For

- Commitment Control

Query Manager

Query Manager

- Use QUERY MANAGER
 - Build SQL statement in CL using variables
 - Using STRQMQRV in your CL, call a QM Query passing the variables to execute the SQL
 - Can be somewhat cryptic with the CAT(catenation) in the CL, but works well.
 - Like building the OPNQRYF
 - Only character data allowed

Query Manager

```
Columns . . . : 1 79
QM . . .
Type SQL Statement
***** Beginning of Data *****
0001.00 &STATEMENT1
0002.00 &STATEMENT2
0003.00 &STATEMENT3
0004.00 &STATEMENT4
0005.00 &STATEMENT5
0006.00 &STATEMENT6
0007.00 &STATEMENT7
0008.00 &STATEMENT8
0009.00 &STATEMENT9
0010.00 &STATEMENT10
***** End of Data *****

F2=Alternate keys  F3=Exit  F4=Prompt  F5=Run report  F6=Run sample
F9=Retrieve        F24=More keys
(C) COPYRIGHT IBM CORP.
```

ME a 07/010

1902 - Session successfully started

Query Manager

```
New York
File Edit View Communication Actions Window Help
Columns . . . : 1 80
SEU=> Edit
NECPRDSRC/JDESRC
J5541SQL01
***** Beginning of data *****
0017.00 PGM 050030
0018.00 DCL VAR(&SQL) TYPE(*CHAR) LEN(40) 000117
0019.00 DCL VAR(&SQL1) TYPE(*CHAR) LEN(55) 000117
0020.00 DCL VAR(&SQL2) TYPE(*CHAR) LEN(55) 000117
0021.00 DCL VAR(&SQL3) TYPE(*CHAR) LEN(55) 000117
0022.00 DCL VAR(&SQL4) TYPE(*CHAR) LEN(55) 000117
0023.00 DCL VAR(&SQL5) TYPE(*CHAR) LEN(55) 000117
0024.00 DCL VAR(&SQL6) TYPE(*CHAR) LEN(55) 000117
0025.00 DCL VAR(&SQL7) TYPE(*CHAR) LEN(55) 000117
0026.00 DCL VAR(&SQL8) TYPE(*CHAR) LEN(55) 000117
0027.00 DCL VAR(&SQL9) TYPE(*CHAR) LEN(55) 000117
0028.00 DCL VAR(&SQL10) TYPE(*CHAR) LEN(55) 000117
0029.00 DCL VAR(&B1) TYPE(*CHAR) LEN(1) VALUE('') 000118
0030.00 DCL VAR(&JOBNAME) TYPE(*CHAR) LEN(10) 000118
0031.00 DCL VAR(&JOBUSER) TYPE(*CHAR) LEN(10) 000118
0032.00 DCL VAR(&JOBNBR) TYPE(*CHAR) LEN(6) 000118
0033.00 DCL VAR(&SYSDAT ) TYPE(*CHAR) LEN(6) 000118
0034.00 DCL VAR(&SYSTEM ) TYPE(*CHAR) LEN(6) 000118
0035.00 DCL VAR(&JULDAT ) TYPE(*CHAR) LEN(5) 000118
0036.00 DCL VAR(&JDEJULDAT ) TYPE(*CHAR) LEN(6) 000118
0037.00 000118
0037.01 120116
0044.00 000118
```

ME a 25/001
1902 - Session successfully started

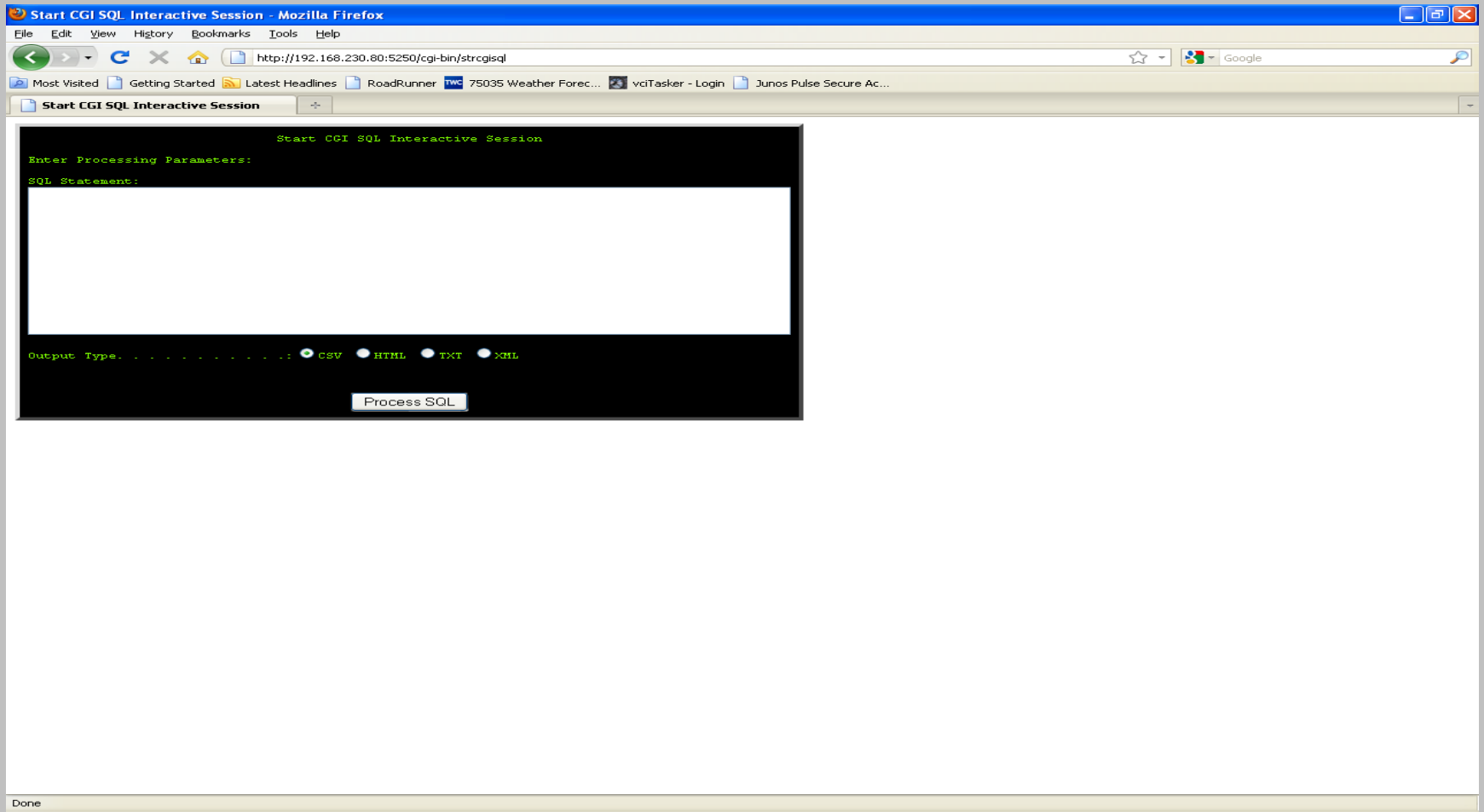
Query Manager

```
New York
File Edit View Communication Actions Window Help
Columns . . . : 1 80
SEU=>
0044.00
0045.00      CHGVAR      VAR(&SQL1) VALUE('UPDATE F4101 SET IMLLX +
0046.00              = 200')
0046.04
0047.00      CHGVAR      VAR(&SQL2) VALUE('WHERE IMGLPT LIKE ' +
0048.00              *TCAT &B1 *TCAT ', + 'FG%' *TCAT &B1 *TCAT')
0050.00              &SYSTIM *TCAT ',')
0051.00      CHGVAR      VAR(&SQL3) VALUE('SDJOBN=' *TCAT &B1 *TCAT +
0052.00              &JOBNAME *TCAT &B1 *TCAT ', ' *TCAT +
0053.00              'SDUSER=' *TCAT &B1 *TCAT &JOBUSER +
0054.00              *TCAT &B1)
0055.00      CHGVAR      VAR(&SQL4) VALUE('WHERE SDAN0=52 AND SDMCU = +
0056.00              ''          200'' AND')
0057.00      CHGVAR      VAR(&SQL5) VALUE('SDDC0=' 'ST'' AND +
0058.00              SDNXTR=' '500''')
0059.00
0060.00      STQMORY      QMORY(QGPL/STATEMENT) OUTPUT(*) +
0061.00              SETVAR((STATEMENT1 &SQL1) (STATEMENT2 +
0062.00              &SQL2) (STATEMENT3 &SQL3) (STATEMENT4 +
0063.00              &SQL4) (STATEMENT5 &SQL5) (STATEMENT6 +
0064.00              &SQL6) (STATEMENT7 &SQL7) (STATEMENT8 +
0065.00              &SQL8) (STATEMENT9 &SQL9) (STATEMENT10 +
0066.00              &SQL10))
0066.01
NECPRDSRC/JDESRC
J5541SQL01
ME a
15/001
1902 - Session successfully started
```

STRCGISQL

- Using Apache, CGI, and SQL
 - Basic Apache server
 - Allow CGI
 - Run SQL Selects
 - Data is displayed in CSV, HTML, TXT, or PDF
- Quick data dump to Excel

STRCGISQL



STRCGISQL

Start CGI SQL Interactive Session - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://192.168.230.80:5250/cgi-bin/strcgisql

Most Visited Getting Started Latest Headlines RoadRunner 75035 Weather Forec... vciTasker - Login Junos Pulse Secure Ac...

Start CGI SQL Interactive Session

```
Start CGI SQL Interactive Session
Enter Processing Parameters:
SQL Statement:
select sddocs, sdlitm, sdsogs, (sdaexp * .01) as LineTtl
from necprddta/F4211
where sdlitm = '40444'
Output Type: . . . . . :  CSV  HTML  TXT  XML
[Process SQL]
```

Done

STRCGISQL

- <http://www.mcpressonline.com/database/db2/make-db2-data-downloads-easy-with-start-cgi-sql-session-strcgisql.html>

Helpful Links

- SQL Reference
- <http://publib.boulder.ibm.com/infocenter/series/v5r3/index.jsp?topic=/db2/rbafzmst02.htm>
- SQL Error Codes
- <http://publib.boulder.ibm.com/infocenter/series/v5r3/index.jsp?topic=/rzala/rzalaml.htm>
- “An Introduction to SQL,” with Paul Tuohy, System i Developer
- Dates 01/17/2012 - 01/26/2012
- <http://www.systeminetwork.com/events>
- “Power Tips for iSeries Database and SQL,” with Paul Conte
- <http://www.amazon.com/Power-Tips-iSeries-Database-SQL/dp/1583040986>

Questions?

- cadair@natenv.com