The Fall IUA Workshop Proudly Presents

ROLLBACK TO THE FUTURE

October 22 – 24, 2007
CA Education Center
Herndon, Virginia

Keynote Speakers:

John Cullinane
Founder of Cullinane Corporation and Cullinet Software

Judith Kruntorad
CA Vice President, CA IDMS Product Line Manager

Cost: $249.00 (USD) per person

Please join us as we ROLLBACK TO THE FUTURE for our IUA Fall Workshop. We will have 3 tracks: Application Programming, Database Administration, and IDMS Modernization.

To register, go to the IUA website (www.iuassn.org) and select the EVENTS button at the top of the page.

The CA-IDMS Database and Applications User Association

http://iuassn.org

Summer 2007, Number 67

IUA CONNECTIONS

INSIDE THIS ISSUE

LETTER FROM THE EDITOR

By Gary Cherlet
Justice Technology Services
South Australian Department of Justice

My role

Although listed in the credits as the 'Editor', I have not fulfilled the traditional editorial role in terms of actually reviewing the submissions. Rather I have organised contributors and collated their submissions – and then Rebecca Shaw did all the hard work. Many thanks Becki for all your efforts on this, and past issues of IUA Connections.

Thanks to our contributors

We have two types of contributors in this issue. We have some substantial input from CA. The reader can be well assured that these have been vetted for both technical accuracy and for adherence to CA best practice for publications. Nothing goes out the doors at CA without 'process' being adhered to.

Our second type of contributor are the developers and DBAs who have gone those extra yards to pass on the benefit of their experience and hard earned knowledge. These contributions are accepted in good faith that the writer has made their best effort to accurately describe their own experience in a particular environment.

As the reader we must accept the author's good intentions, that our experience might not match the writer's, and that there may be minor technical issues. Many will have seen the lively discussions that sometimes crop up on IDMS-L when somebody has an 'issue' with a post or thread. This is good – it means people are reading what has been run up the flagpole, are taking it seriously and thinking about it, and in that great democratic tradition and spirit are taking the opportunity to reply. I think this is a good thing.

Sincerely from myself and on behalf of our readership – many thanks to all contributors who have been willing to 'stick their heads up over the trenches' and go the extra

(continued on page 3)

MESSAGE FROM INTERNATIONAL CHAIR

By Terry Schwartz

It has been six years since I last wrote an article for Connections as IUA chair. I am pleased that I can start this article with a shameless plug for our Herndon Workshop.

I strongly encourage you to spread the word in your shop to anyone who might benefit from coming to experience quality IDMS education. I am truly excited about the opportunity for us to have John Cullinane as our keynote speaker. As a pioneer in the software business and former owner of the IDMS software, John brings a wealth of history and insight to our workshop. Couple a great keynote speaker with excellent sessions from CA and IDMS industry experts, and there is a potential great learning experience.

Over the last several years Service Oriented Architecture (SOA) has become the most visible IT industry trend. According to Wikipedia SOA is defined as:

"Service Oriented Architecture (SOA) is an evolution of distributed computing and modular programming. SOAs build applications out of software services. "

If you boil this down, an example of an SOA structure would be a Java or .NET application calling multiple web services to exchange data or invoke business logic. This loosely coupled structure of multiple services, potentially on multiple platforms, is the basis of SOA.

So how does SOA relate to us and the CA-IDMS software? Actually CA-IDMS is well positioned to play in the SOA environment. There are several methods by which you could have your CA-IDMS systems participate in an SOA application. If you have the SQL option and CA-IDMS server you can write table procedures that can function as a form of service. Table procedures can be called from Java or .Net via SQL calls and can contain business logic.

With the introduction of the TCP/IP interface into CA-IDMS version 16, we gained the ability to write our own web services. I have personally written 3 CA-IDMS based web services that exchange standard XML documents with .NET applications. We have plans to implement 2 more web services within the next 4 months. I have even given thought to writing a SOAP handler for IDMS so that XML with a standard SOAP wrapper could be sent to CA-IDMS. A SOAP call sends XML and instructions in the SOAP wrapper on what function (think IDMS task or program) is to be executed.

There are also 3rd party products that you can purchase where you can define web services that talk to CA-IDMS.

I am looking forward to our IUA workshop and another good year of service to the CA-IDMS community. We are always interested in your feedback, so if you have any IUA questions or issues please contact us at info@iuassn.org.

Best regards, Terry Schwartz - IUA Chairperson

Letter from the Editor cont'd from page 2

distance to get your articles to us for publication in this issue of IUA Connections.

The issue with no theme

This issue did not start out with a theme, but it must be a sign of the times that many of the contributions focus on 'modernisation' and taking advantage of the new features of IDMS-DB/DC that make it an equal participant in today's world of two-phase commits, using TCP/IP as the common denominator for inter-environment communications, SQL as a great leveller for simplifying access to non-SQL data on an IDMS back-end machine, and Server providing JDBC and ODBC access to the above.

The emerging 'theme' is appropriate as it follows so well from the theme that CA's IDMS stream had at the last CA-World, and the 'Back to the Future' theme for the IUA Workshop next October in Herndon. Watch this space to see what the next issue's 'theme' turns out to be!

Future Direction

My plan is to have regular contributions from CA, and from regular volunteers who have offered to provide at least one article per issue. This gives a solid base for preparing the next issue - on top of which we can round out the structure with articles from users. Tell us about your successes, your trials and tribulations, and the applications that are 10, 15, or 20 years old and are still providing value to your company. Better yet – tell us about those exciting **new** projects and applications! All contributions gratefully accepted and greatly appreciated.

That's all there is – because there is no more!

Cheers - Gary

TRAVEL TIDBITS FOR THE HERNDON AREA

By Diane Montstream

This year the IUA workshop is being held in Herndon Virginia. Herndon is right outside Washington DC so there are many things to do for those who want to come in early or stay after the workshop.

For those of you who are not familiar with Herndon here is some history about it the town. This information is from a local website.

Herndon was named for Commander William Lewis Herndon, American naval explorer and author of *Exploration of the Valley of the Amazon*. Commander Herndon captained the ill-fated steamer *SS Central America*, going down with his ship while helping to save over 150 of its passengers and crew. The settlement was named Herndon in 1858. In the 1870s, many Northern soldiers and their families came to settle in the area, taking advantage of moderate climate and low land prices.

Originally part of the rural surroundings of the Washington, D.C. area, the town of Herndon developed into a hub of dairy farming and vacationing for area residents, aided by its presence along the Alexandria, Loudoun and Hampshire Railroad (later to become the Washington and Old Dominion Railroad). When the railroad was converted into a hike-and-bike trail, Herndon capitalized on history and small-town feel (in a major metropolitan region) by converting its train station into a museum and visitors center.

On January 14, 2004, the Town of Herndon commemorated its 125th anniversary.

Fall is typically a beautiful time of year as far as the weather goes. You can often have days reaching into the high 70's but the nights get cool so a light jacket is advisable.

For those who don't want to venture into Washington DC and would like to see the top ten things to do in Herndon check out http://www.herndoncuisine.com/topten.html.

CONTRIBUTED SOFTWARE LIBRARY

Save time and use the experience of others to resolve problems.

BUSINESS VALUE CORNER

By Greg Beedy

Did you know?

- ⇨ That a huge multi-national, household-name corporation runs over $10 billion of its business through their core CA IDMS applications each year?
- ⇨ That several U.S. government agencies use CA IDMS systems to keep the United States safe. How do they do this? Sorry, we can't tell you, but they tell us these are life and death systems.
- ⇨ That in one New England state you can register your car over the internet with a web front-end that talks to a CA IDMS back-end?
- ⇨ That BT manages over 20 million phone customer accounts using CA IDMS systems?
- ⇨ That a number of U.S. and international government agencies use CA IDMS systems to manage their criminal justice operations?
- ⇨ That one American state processes over three million transactions a day on its inmate tracking system?

Greg Beedy is Director, Product Management for the CA IDMS family of products in Framingham MA. After joining Cullinane Corp. in 1979, he has worked with IDMS in a variety of capacities including applications development, software delivery, technical support systems and management. Greg would like to hear how IDMS delivers business value to your organization.
greg.beedy@ca.com

DBA HUMOUR

FROM THE AUSTRALIAN IUA ARCHIVES

Q: How many DBA's does it take to screw in a light bulb?

A: None – that's hardware!

Q: What do you call a DBA who's wearing a Santa suit and has a banana in each ear?

A: Anything you want – because he can't hear you!

Editor's note: can you, or the combined efforts of your local User Group, do better? Send in your best DBA jokes and get yourself published in this prestigious publication. It's just like being a Public Servant – we're in it for the glory – not for the money!

ADS TERMINOLOGY 101 - EXTENDED RUN UNIT

By Linda Casey

To understand what an extended run unit is, it's important to first understand what a run unit is, and how it's created within the ADS environment. Let's take a few minutes and talk about how dialogs interact with the IDMS database.

Dialogs issue navigational DML verbs to access data within the IDMS database. When the first DML verb is issued in a dialog, a session is established with the database. This session is known as a run unit. That first DML verb triggers the ADS runtime system to BIND the run unit and READY the areas attached to the subschema. When the BIND and READYs occur, all kinds of wonderful things happen automatically for the dialog: addressability to variable storage where the IDMS communication block lives is established, addressability to storage areas for records is established, and areas are READYed.

The dialog is now ready to communicate with the database, retrieving or updating records. The run unit normally stays open until a control command is encountered, such as a DISPLAY, LINK, INVOKE, etc. The exception is when a run unit is automatically "extended" or left open. A dialog ***automatically*** extends its run unit to another dialog under the following conditions:

1. The LINKed to or INVOKEd dialog has no subschema associated with it - or

2. The subschemas of both dialogs are the same, and the called dialog doesn't READY the areas in a more exclusive way than the calling dialog.

So why extend the run unit?

Recovery is one major reason. Since both dialogs are operating on the same run unit, if an ABEND occurs in the linked to dialog, then all database updates from the beginning of the run unit will rollback together. If the updates are not rolled out as a package, logical database inconsistency can result.

The second reason to extend a run unit is BIND/FINISH overhead. The creation of a run unit is very costly in terms of memory consumed and commands executed by the IDMS system in setting up the database access environment for the dialog.

Once extended, the run unit will stay open until another control command is issued. So what does that mean? If Dialog A links to Dialog B and the run unit gets extended, if Dialog B issues a DISPLAY command, then the extended run unit will close. Other control commands will also cause the extended run unit to close, but results can vary depending on conditions. A run unit can be passed down multiple LINK levels in this manner.

Since Release 10.2 of IDMS, a run unit could be ***unconditionally*** extended even if the same subschema is not being used, or the same subschema is being used but the READY modes are inconsistent. This is achieved by explicitly passing SUBSCHEMA-CONTROL to the called dialog. To do this, the called dialog can only access records defined to the calling dialog's subschema and the can only utilize update operations if the calling dialog has readied those areas in update mode.

```
LINK TO DIALOG .....
     USING (SUBSCHEMA-CONTROL).
```

Of course by using "link to program .... using (SUBSCHEMA-CONTROL)" it is possible to extend a run unit to DC-Cobol or DC-Assembler programs as well. In all cases please take note that if a run unit does not already exist – a run unit will be established for the calling dialog before control is passed to the called dialog or program.

Linda J. Casey, Managing Member, Run Right, LLC. Is a small business owner with 32 years in the software industry specializing in systems design, development and implementation, management of both application and infrastructure projects, training development and delivery and extensive vendor experience with both Cullinet and CA.

(continued on page 5)

CHECK OUT THE IUA ARCHIVE LIBRARY OF IDMS PRESENTATIONS

HOW EXTEND SPACE SAVED MY LIFE
(WELL, NOT REALLY, BUT YOU GET THE IDEA)

By Chris Hoelscher

Our shop keeps close watch on IDMS database utilization – we keep area utilization history for 378 days, and determine trends based on many factors, the result of which is that we can (usually) tell to the day when a growing database area would reach 70% full or 100% full if left untended.

The problem with all, this, however, is that it depends upon **"history"** and **"trends"**. When a batch process (planned by an end user, but not commumicated to the DBA staff) inserts a disproportionate (with regards to history) amount of data, our "**trends**" go out the window, and often, the database area fills, and we get calls asking us how we could let this happen (to which we explain to them, in no uncertain tests, EXACTLY how this happened). Nonetheless, we are (correctly) tasked with getting the database area available for updates (yesterday, it seems).

Since the arrival of release 12.0 in 1991<?>, I have found a lifesaver in the form of the BCF command "EXTEND SPACE". It has saved our "bacon" countless times, and made our life quite a bit easier even in situations where the database area has NOT filled.

Following are the steps we use to implement EXTEND SPACE

1) Assure that sufficient unassigned pages exist adjacent to the end of the "problem" database area (**LOOK DMCL SORTED PAGES** is great for determining this)

2) Create a physical dataset with the same attributes as a file of the "problem" database area (**ISPF 3.2** is great for doing this) – wherever possible, We have been sizing the new file as large as the previous (only?) file assigned to the "problem" database area, or less if we are constrained by the number of (previously) unassigned pages determined in step (1).

3) Define a file definition to the IDMS system catalog, ensuring to create a unique file name, DDname, and DSname.

```
CREATE FILE segment-name.file-name-2
     ASSIGN TO ddname-2
     DSNAME dataset-name-2
     DISP SHR
     NONVSAM
;
```

4) Alter the "maximum pages" attribute of the "problem" database area to a value large enough to contain the additional pages.

```
ALTER
PHYSICAL AREA segment-name.area-name
MAXIMUM SPACE total-pages  PAGES ;
```

5) Alter the "problem: database area as follows

```
Alter area segment-name.area-
name
Extend space new-pages pages
Within file file-name-2
     FROM 1 FOR new-pages BLOCKS
;
```

6) For each DMCL in which an existing file for the "problem" database area was overridden, determine the overrides (buffer name, shared cache name, etc) and apply them to the new file in the "problem" database area.

```
ALTER DMCL dmcl-name
INCLUDE segment-name
INCLUDE segment-name.file-name-2
BUFFER buffer-name
SHARED CACHE cache-name    ;
;
```

7) Generate & Link all DMCLs in which the "problem" database area is defined

8) Format the new **file(s)** using the BCF "FORMAT" command

9) IF this is an emergency, execute:

LOCKMON REL ALL LOCKS FOR AREA ***segment-name.area-name***

to release all (presumably retrieval) locks for the "problem" database area that would otherwise prevent the DMCL NewCopy from successfully completing.

10) Vary DMCLs with read-only access to the "problem" database area NewCopy

(It is important to newcopy the read-only DMCLs first because you do not want to create a situation where records can be written to the extended area, but other processes in other CVs are looking for the records (perhaps by DBKEY) but can access them due to the extended area not available to the read-only DMCL

11) Vary the DMCL with update access to the "problem" database area NewCopy

12) Enjoy your newly-extended database!

Should an unload/reload be performed soon after the EXTEND SPACE? Well ... in our shop a performance problem is a performance problem when someone says it is a performance problem – so for us we would like to unload/reload such areas, but do not place a high priority on such tasks. There is always one more fire to put out .....

Chris Hoelscher
Senior IDMS & DB2 Database Administrator
Humana Inc
502-476-2538
choelscher@humana.com

IDD CODE TABLES AND SQL - A LESSON ABOUT 'DATA INDEPENDENCE'

By Gary Cherlet

Most CA-ADS developers have used IDD Code Tables for editing, validation, and decoding on their IDMS-DC/UCF Maps. The encode/decode Tables in particular are interesting because many sites have chosen to use them either in place of, or as well as database "tables" for storing this type of data.

With SQL you can actually treat the 'source' for the IDD Tables as an SQL database Table. Of course you need to know that the 'source' is stored as an IDD Module, and that the 'load module' used at run-time in ADS applications and IDMS-DC/UCF Mapping may differ because somebody has not done the 'Generate'. However – given that we can still use an SQL View on top of a non-SQL IDD database structure to return the IDD Table data as an SQL 'Table' (pun intended).

```
create view rsqlshr.codeValues
( codeTable, codeTableName, encodeValue, decodeValue ) as
select          MOD_NAME_067                    as  codeTable     ,
        substr(MOD_NAME_067,1,8)                as  codeTableName ,
        cast(rtrim(substr(CMT_INFO_084_1,5,34))    as VARCHAR(34))
                                                as  encodeValue ,
          concat(substr(CMT_INFO_084_1,39,12) ,
                 CMT_INFO_084_2       )     as  decodeValue
FROM      jisdict."MODULE-067" ,
          jisdict."MODCMT-084"
WHERE     "MODULE-MODCMT"
          and   LANG_067 = 'TABLE'
          and   CMT_ID_084 = -9 and IDD_SEQ_084 = 0
          and   substr(CMT_INFO_084_1,5,1) <> '?'
          and   substr(CMT_INFO_084_1,5,34) <> ' ';
```

The rsqlshr.codeValues View will return some 'special case' rows that are peculiar to our use of Tables at JIS (see the 'substring' conditions in the View definition) – but it can be used in SQL 'joins' for decoding purposes, or to simply extract all of the encode:decode value pairs. The View has the Code Table Name in it twice – once as CodeTable CHAR(32) for use as a 'calc' key entry, and once as codeTableName CHAR(8) to reflect the fact that all Code Tables only have 8 characters

Extracting IDD Code Table Values

The following example shows a likely use of the VIEW – that is to obtain a list of code and decode values from a nominated IDMS Table, to download to a mid-range application for future reference.

```
SELECT SUBSTR(ENCODEVALUE,1,8) AS ENCODE,
       SUBSTR(DECODEVALUE,1,24) AS DECODE
  FROM RSQLSHR.CODEVALUES WHERE CODETABLE     = 'TBORENDC';
*+
*+ ENCODE    DECODE
*+ ------    ------
*+ DIS       DISCHARGED
*+ ESC       ESCAPED
*+ HVO       HOME VOLUNTARY HOME DETE
*+ HBC       BREACH OF HOME DETENTION
*+ HCO       OFFENCE COMMITTED ON HOM
*+ HUL       UNLAWFULLY AT LARGE ON H
<<<<<<<<<<<<<<<<<< snip >>>>>>>>>>>>>>>>>>>
*+ DEATH     DEATH OF CLIENT
*+ EXPBR     EXPIRED IN BREACH
*+
*+ 20 rows processed
                        * * *   END OF DATA   * * *
```

continued on page 8)

IDD Code Tables and SQL continued from page 8

Disguising the IDD Table as a Base Table – Data Independence

In the next example we use a 'View of a View' to demonstrate 'data independence' through effective use of SQL. If you look at the previous example, and think about it from the developer's point of view, it has the following characteristics:

- You need to know the IDD Table Name
- Without specifying 'alias' names in the query the Column names are encodeValue and decodeValue
- What becomes of these names when you have processes that reference multiple IDD Tables?

Wouldn't it be nice if you didn't need to know the Table name, and if the encode and decode values had Column Names that reflected the 'domains' that they represented in a database sense? For example, the Table referenced above is the 'Reason Order Ended' Table. Wouldn't it be nice if the Column name for the ***decodeValue*** was ***reasonOrderEnded***?

In the following View we do all of the above. Just consider an IDD Table of 'State Codes', which we have called TBSTATEC.

```
connect to jisdict;
drop   View RSQLSHR.AustralianState ;    (idd table TBSTATEC)
create View RSQLSHR.AustralianState (
stateAbbreviation    ,
stateName                          )
as
select    substr(encodeValue,1,3)     as stateAbbreviation  ,
          substr(decodeValue,1,30)    as stateName
  from    rsqlshr.codeValues
 where    codeTable = 'TBSTATEC';
```

The following example shows one use of this VIEW – that is to obtain a list of code and decode values from IDD State Code Table.

```
SELECT * FROM rsqlshr.AustrlianState;
*+
*+ STATEABBREVIATION  STATENAME
*+ -----------------  ---------
*+ ACT                AUSTRALIAN CAPITAL TERRITORY
*+ NSW                NEW SOUTH WALES
*+ NT                 NORTHERN TERRITORY
*+ QLD                QUEENSLAND
*+ SA                 SOUTH AUSTRALIA
*+ TAS                TASMANIA
*+ VIC                VICTORIA
*+ WA                 WESTERN AUSTRALIA
*+
*+ 8 rows processed
                        * * *   END OF DATA   * * *
```

Notice how the SQL coder (Java, .Net, mainframe) does not need to know the physical implementation of the AustralianState Table. If we decided in the future to implement this as an IDMS/SQL Table, or as an IDMS non-SQL database record type – we could simply change the View and **any** code the references this View would not be affected. **That is what 'data independence' is all about!**

Oh, and in case you didn't recognize the 'State Names' – notice that the Table name is **AustrlianState!** Cheers mate.

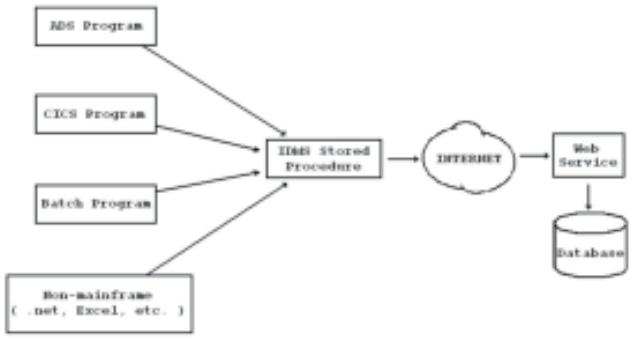
Gary Cherlet, Justice Technology Services

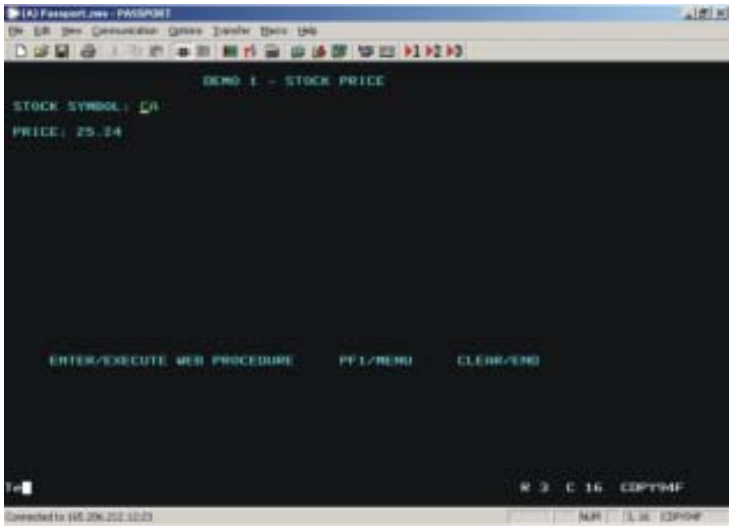
Gary Cherlet is an expat Canadian pretending to be an Australian. He has been working with IDMS (or similar) databases for somewhere around 25 years. Gary still dabbles in the dark arts of Assembler, Cobol, TSO clists, and REXX, and has the odd foray into the wonders of 'mark up' languages such as GML, HTML and XML.

ACCESSING WEB SERVICES FROM MAINFRAME PROGRAMS

By Kay Rozeboom

This article describes one way to access external web services from mainframe programs, using IDMS SQL called procedures and the IDMS TCP/IP sockets interface. As the diagram below shows, a single SQL procedure can be called from ADS, CICS, and batch programs. You can also call the procedure from a non-mainframe program using IDMS Server. This method requires: 1) IDMS maintenance level R16 SP2 or higher; 2) the IDMS SQL option; and 3) a SOAP-based web service to access.





Example – Stock Price

In this example, we will call a public "stock price" web service from a CICS program. The CICS screen to the left takes a stock symbol as input ("CA" in this case), and returns the current price of the stock.

The CICS program calls an IDMS stored procedure using either of the following SQL statements. This same SQL syntax can be embedded in a batch program, ADS dialog, or non-mainframe program. For example, if you have IDMS server, you can select your stock price into an Excel spreadsheet.

```
SELECT * FROM SCHEMA1.STOCK_PRICE WHERE STOCK_SYMBOL = 'CA'
CALL SCHEMA1.STOCK_PRICE (STOCK_SYMBOL = 'CA')
```

The SQL procedure is defined as follows. As you can see, it looks like any SQL table. "External name" points to the underlying DC-COBOL program, which is PROGRAM1 in this case.

```
CREATE PROCEDURE SCHEMA1.STOCK_PRICE
 ( STOCK_SYMBOL                    CHARACTER(5) WITH DEFAULT,
   QUOTE_RESULT                    CHARACTER(10) WITH DEFAULT,
   ERROR_MESSAGE                   CHARACTER(256) WITH DEFAULT
 )
   EXTERNAL NAME PROGRAM1 PROTOCOL IDMS
   DEFAULT DATABASE NULL
   USER MODE
   LOCAL WORK AREA 1024
   ;
```

PROGRAM1, called by the SQL procedure, does the following:

- Format a SOAP request message in XML format, using the stock symbol passed through the SQL procedure. See example below.
- Link to PROGRAM2 (a generic SOAP request program), passing the SOAP request along with some additional information such as host name, web service location, and soap action. PROGRAM2 does all of the TCP/IP processing. It passes the SOAP request to the external web service, waits for the response, then returns the response to PROGRAM1.

(continued on page 10)

Accessing Web Services from Mainframe Programs cont'd from page 9

- Link to PROGRAM3 (a generic SOAP editing program), passing the response received from PROGRAM2. PROGRAM3 performs various edits such as removing the HTTP headers, and converting special characters to standard XML format.
- Parse the resulting XML, using the COBOL XML parser, to find the stock price. See example below.
- Return either the stock price (QUOTE_RESULT) or an error message (ERROR_MESSAGE).

Here is a sample SOAP request for the stock quote web service. This syntax is hard-coded in the working storage of PROGRAM1. Only the stock symbol (highlighted) is inserted at run time.

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
    <soapenv:Body>
       <tns:GetQuickQuote
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://ws.cdyne.com/">
          <tns:StockSymbol>CA</tns:StockSymbol>
          <tns:LicenseKey>0</tns:LicenseKey>
       </tns:GetQuickQuote>
    </soapenv:Body>
</soapenv:Envelope>
```

Here is a sample SOAP response received from the web service. PROGRAM1 uses the COBOL XML parser to extract the stock price (highlighted).

```
<soap:Envelope
     xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <soap:Body>
       <GetQuickQuoteResponse xmlns="http://ws.cdyne.com/">
          <GetQuickQuoteResult>25.25</GetQuickQuoteResult>
       </GetQuickQuoteResponse>
    </soap:Body>
</soap:Envelope>
```

Summary

By embedding an external web service call inside an IDMS SQL procedure, you can read or update non-mainframe data from batch, CICS, and ADS programs.

A common SOAP request program (PROGRAM2 in the example above) hides the TCP/IP details from your application programmers. They do not need be concerned with TCP/IP commands, HTTP headers, or ASCII/EBCDIC conversions. The person who writes the IDMS SQL procedure only needs to learn how to format a SOAP request, and interpret a SOAP response. The people who write the front-end programs (batch, CICS, ADS) don't need to know anything about TCP/IP or SOAP - they only need to know how to make an SQL call.

Kay Rozeboom is a DBA/Systems Programmer with the State of Iowa. She has 18 years of IDMS experience. Her special interest is in integrating mainframe data and applications with other platforms.

SAMPLE SQL USER FUNCTION IN CA IDMS R16

By Ian Hill

One of the new features of CA IDMS r16 is the implementation of user-defined SQL scalar functions. A scalar function is something which can be used in an SQL statement to perform an operation on a column or expression.

For example, consider the simple SQL statement:

```
SELECT * FROM DEPARTMENT WHERE DEPT_ID=6000;
*+
*+ DEPT_ID   DEPT_HEAD_ID   DIV_CODE   DEPT_NAME
*+ -------   ------------   --------   ---------
*+     6000           1003   D09        LEGAL
*+
*+ 1 row processed
```

If we wanted to return only the first three characters of the department name, you could code it as follows:

```
SELECT DEPT_ID, SUBSTR(DEPT_NAME, 1, 3) FROM DEPARTMENT WHERE DEPT_ID=6000;
*+
*+ DEPT_ID   SUBSTR(FUNCTION)
*+ -------   ----------------
*+     6000   LEG
*+
*+ 1 row processed
```

Historically, most SQL implementations have supported only a predefined set of scalar functions. With CA IDMS r16, you can write your own – and as usual they can be written in Cobol, PL/1 or Assembler, but now they can also be written in CA ADS. In this example, we'll be looking at a useful function which involves some interesting concepts including optional parameters, calling IDMSIN01 to retrieve user session values, and the function itself issuing SQL statements.

Consider the supplied SQL function DAYNAME, which returns the name of the day of the week, given an input date. An example of this is as follows:

```
SELECT *, DAYNAME(COL2) AS "DAY NAME" FROM T_DATE;
*+
*+    COL1   COL2         DAY NAME
*+    ----   ----         --------
*+       1   2007-07-01   Sunday
*+       2   2007-07-02   Monday
*+       3   2007-07-03   Tuesday
*+       4   2007-07-04   Wednesday
*+       5   2007-07-05   Thursday
*+       6   2007-07-06   Friday
*+       7   2007-07-07   Saturday
*+       8   2007-07-08   Sunday
*+       9   2007-07-09   Monday
*+      10   2007-07-10   Tuesday
*+      11   2007-07-11   Wednesday
*+
*+ 11 rows processed
```

The DAYNAME function returns the names of the weekdays in English only. Let's assume that we had a requirement for a function to return the name in a foreign language, and that the language could be either supplied as a second parameter to the function call, or if omitted, is to be retrieved according to a parameter in the profile of the signed on user.

continued on page 11)

A Sample SQL User Functionin CA IDMS R16 continued from page 10

The function would be defined in the dictionary with this syntax:

```
CREATE FUNCTION DAYNAMEL
  ( P1                                DATE,
    P2                                CHARACTER(2)
  )
    RETURNS CHARACTER(12)
    EXTERNAL NAME DAYNAMED PROTOCOL ADS
    DEFAULT DATABASE CURRENT
    ESTIMATED ROWS 1
    SYSTEM MODE
    TRANSACTION SHARING OFF
    LOCAL WORK AREA 0
    ;
```

Note the table like syntax, where the "columns" are used to define each parameter in the function call (SQL Functions are represented in the catalog by a row in the SYSTEM.TABLE table with a TYPE of 'F'). Note also the RETURNS clause which defines the datatype of the value to be returned. In the EXTERNAL NAME clause we identify the name of the ADS dialog to be called – it must be mapless. DEFAULT DATABASE CURRENT is specified to inherit the database name of the parent session.

When defining the dialog, you must assign a work record with the name <SCHEMA>.<FUNCTION>, in this case IJHSQL.DAYNAMEL. This work record does not exist in the dictionary but is built internally by ADSC at compile time, in order to reflect the parameters passed to the function.

In the ADS code, field P2 is used to reference the second (optional) parameter passed in the function call. Indicator variables follow the usual convention in that they are named <column>-I, so in this case, we examine the value of P2-I. If it is zero, we know the function was called with a particular language code. If P2-I is not zero (-1), we know that no value was passed, and so we do the call to IDMSIN01 to determine the language code of the signed on user from his/her profile.

The system-supplied field USER-FUNC (with its indicator variable USER-FUNC-I) are used to assigned the result value and be passed back to the calling program.

continued on page 13)

UPCOMING EVENTS IN EUROPE AND THE UK

By Jan Rabaut

The European countries will have their regional IDMS events on the following dates.

- August 28-29 2007 Finland IDMS user group.
- November 23 2007 United Kingdom IDMS user group.
- November 25-26 EIUA autumns board meeting in France (Paris)
- November 27-28 CA info exchange Germany
- November 29 AUI IDMS user group France
- November 30 BIUA Belgium and the Netherlands IDMS user group Event

BOARD OF DIRECTORS ELECTIONS

By Diane Montstream

Elections were held this year to fill two Board of Directors positions. The new directors were announced at the Business meeting at CA-World in New Orleans. The new directors are Bob Wiklund and Craig McGregor.

Bob Wiklund works for Tiburon Technologies. Bob has served as a Director on the IUA Board in the past. He was also the IUA Chairperson and has served on many IUA committees. He is currently over-seeing the planning for the IUA Workshop.

Craig McGregor currently works for Acxiom and has been interacting at different support levels with various IDMS customers for the past ten years. Craig has been and continues to serve as the president of the IDMS Chicago Midwest User Group.

A Sample SQL User Functionin CA IDMS R16 continued from page 11

The ADS code to implement the function looks like this:

```
ADD  PROCESS NAME IS DAYNAMED-PREMAP VERSION IS 1
         PROCESS SOURCE FOLLOWS
MOVE 'EN' TO WS-DEFAULT-LANGUAGE.              ! USE ENGLISH IF NOTHING ELSE
MOVE WS-DEFAULT-LANGUAGE TO WS-LANGUAGE-CODE.  !  CAN BE FOUND
IF P2-I = 0 THEN DO.                           ! IS THERE A 2ND PARM?
    MOVE P2 TO WS-LANGUAGE-CODE.!USE IT
END.
ELSE DO.                                       ! IF NO 2ND PARM SUPPLIED
    MOVE 2 TO IN01-REQUEST-CODE.               ! SET UP AN IDMSIN01 CALL TO
    MOVE 'LNG' TO IN01-KEYWD.                  !  GET THE USER'S LNG
ATTRIBUTE
    MOVE 0 TO IN01-REQUEST-RETURN.!  AND USE THAT
    LINK PGM 'IDMSIN01' USING (IN01-RPB-REC,
                               IN01-REQ-REC,
                               IN01-KEYWD-REC,
                               IN01-VALUE-REC).
    IF IN01-REQUEST-RETURN = 0 THEN DO.        ! WAS THE IDMSIN01 CALL OK?
        MOVE IN01-VALUE TO WS-LANGUAGE-CODE.   ! USE THE LANGUAGE CODE WE GOT
    END.
    ELSE DO.                                   ! OTHERWISE:
        MOVE '38101'    TO SQLPROC-SQLSTATE.   ! ISSUE BAD SQLSTATE
        MOVE 'NON-ZERO RETURN CODE FROM IDMSIN01.'
          TO SQLPROC-MSG-TEXT.                 !
        LEAVE ADS.                             ! AND GET OUT
    END.
END.
MOVE P1 TO WS-DATE-RDF.                        ! CRUNCH SOME NUMBERS
DIVIDE 32 INTO WS-FIRST4 GIVING WS-DATE-NBR.   !  TO GET A SEQUENTIAL
DIVIDE 7 INTO WS-DATE-NBR GIVING WS-WEEK       !  DAY OF WEEK NUMBER ...
                        REMAINDER WS-DAY-NBR.
ADD 1 TO WS-DAY-NBR.                           !  ... BETWEEN 1 AND 7
CALL GETNAME.                                  ! GET DAYNAME FOR NUMBER
IF WS-DAY-NAME = '************' THEN DO.       ! IF IT WASN'T THERE
    IF WS-LANGUAGE-CODE                        !  AND WE WEREN'T AFTER THE
       NE WS-DEFAULT-LANGUAGE THEN DO.         !  DEFAULT (ENGLISH) ONE
        MOVE WS-DEFAULT-LANGUAGE
          TO WS-LANGUAGE-CODE.                 ! GET THE DEFAULT ONE
        CALL GETNAME.
    END.
END.
MOVE WS-DAY-NAME TO USER-FUNC.                 ! RETURN THE RESULT
MOVE 0           TO USER-FUNC-I.               ! WITH NULL INDICATOR OFF
LEAVE ADS.
!
!   COMMON ROUTINE TO GET THE DAY NAME FROM A PERMANENT TABLE.
!
DEFINE SUBROUTINE GETNAME.
EXEC SQL
    SELECT DAYNAME_TEXT
    INTO :WS-DAY-NAME
    FROM IJHSQL.DAYNAME_TABLE
    WHERE DAY_OF_WEEK_NBR=:WS-DAY-NBR
      AND LANGUAGE_CODE=:WS-LANGUAGE-CODE
END-EXEC.
IF SQLCODE NE 0 THEN DO.                       ! IF NON-ZERO SQLCODE
    IF SQLCODE = 100 THEN DO.                  ! 100 IS OK
        MOVE '************' TO WS-DAY-NAME.
    END.
    ELSE DO.
        MOVE '38102'    TO SQLPROC-SQLSTATE.   ! OTHERWISE ISSUE BAD SQLSTATE
        MOVE 'UNEXPECTED SQLCODE=' TO X19.     !  AND FORMAT SQLCODE FOR
        IF SQLCODE GE 0 THEN DO.               !  READABILITY IN MESSAGE
            MOVE SQLCODE TO SQLCODE-DISP.
            MOVE '+' TO SQLCODE-SIGN.
        END.
        ELSE DO.
            MOVE -SQLCODE TO SQLCODE-DISP.
            MOVE '-' TO SQLCODE-SIGN.
```

(continued on page 14)

A Sample SQL User Functionin CA IDMS R16 continued from page 12

The dialog also queries a static SQL table which is defined to house the actual day names in whatever languages are necessary. This makes the function more flexible in that all you need to do to support another language is add the relevant rows to table IJHSQL.DAYNAME_TABLE. If the day name row for the requested language is not found, the function will then look for a default value (in this case English), and if that is not found either it returns asterisks.

Don't forget to create an access module for dialog DAYNAMED!

Here are some examples of calling the function in OCF, given that the signed on user has a language code of 'DK' (Danish), that 'FR' (French) is supported and that 'XX' is not:-

```
SELECT *, DAYNAMEL(COL2) AS "DAY NAME" FROM T_DATE WHERE COL1=11;
*+
*+   COL1  COL2        DAY NAME
*+   ----  ----        --------
*+     11  2007-07-11  ONSDAG
*+
*+ 1 row processed
SELECT *, DAYNAMEL(COL2, 'FR') AS "DAY NAME" FROM T_DATE WHERE COL1=11;
*+
*+   COL1  COL2        DAY NAME
*+   ----  ----        --------
*+     11  2007-07-11  MERCREDI
*+
*+ 1 row processed
SELECT *, DAYNAMEL(COL2, 'XX') AS "DAY NAME" FROM T_DATE WHERE COL1=11;
*+
*+   COL1  COL2        DAY NAME
*+   ----  ----        --------
*+     11  2007-07-11  WEDNESDAY
*+
*+ 1 row processed
```

Note the flexibility this grants to the application developer. They don't have to worry about the language in which the signed on user needs to see the day names. They can just code the function call with the one parameter and all the hard work is done under the covers. Alternatively, if a particular language is required regardless of the user, then that option is also available.

Ian Hill is a Principal Support Engineer for CA IDMS in the CA Denmark office. He has over twenty years experience working with CA IDMS. Prior to joining CA in 1991, he worked as a CA IDMS Application Programmer and DBA in Melbourne, Australia.

WHAT AM I DOING WRONG?

FROM THE JTS DEVELOPER HELP DESK ARCHIVES

To: Cherlet, Gary (JTS)
Subject: What am I doing wrong?
G'day Gary, I'm getting the following error

```
Press <ENTER> to continue - PREVIOUS TASK
ABENDED WITH ABEND CODE D002 AND MESSAGE
CODE 171069
IDMS DC171071 V1 NO RBE FOUND. LAST
DIALOG: WGAA022D : CONTROL COMMAND: LINK
IDMS DC171069 V1 INTERNAL ERROR - NO RBE
WAS FOUND AT 18EAC2BC
V1 Release 16.0 >>> JIS Development CV
<<<  Enter next task:
```

The error message talks about subscript out-of-range, invalid data base procedure, or map edit module But none of those make sense in this context.

The error occurs when the dialog WGAA022D attempts to a hard-coded link to WGAA005D.

Can you help? Thanks M

Subject: RE: What am I doing wrong? Answer

This is usually caused by a record size mismatch - most often for a database record. When the record is obtained it overlays the control area (RBE) and contents of the record that follows it in the Record Buffer Block (RBB).

This problem is usually fixed by generating the dialog to make sure when the record is next allocated at run time that enough space is allocated.

(continued on page 15)

CALLED AND TABLE PROCEDURES TIPS AND TECHNIQUES – BRAIN DUMP #1

By Linda Casey

1. How do you decide when to use a table procedure or the new 16.0 called procedure? When you want many rows returned, known as a result set, then you must use a table procedure. When you only want one row of data, then a called procedure can be used.

2. Are called procedures in 16.0 really easier to use? Yes. If you think about how table procedures interact with the SQL Engine, there is a back and forth "dance" that occurs during table procedure execution; open scan, first row, next row, close scan. This is necessary when result sets are being created. Since the nature of called procedures are "one row only", then things are less complicated. The SQL Engine knows that it will only call the called procedure once; in and out and you're done. The procedure doesn't have to take care of the open scan, close scan, first row, etc processing.

3. So just how different are ADS called procedures from mapless dialogs? An ADS called procedure is setup almost identically in the ADS Compiler as a mapless dialog. If you already know ADS, learning to create ADS called procedures is easy; a few additional records and minor coding differences and presto, you too can be coding ADS called procedures.

4. Can I use all the native SQL verbs to invoke called procedures? No, you can't. Called procedures only allow the use of the SELECT or CALL verbs. All other SQL verbs, such as INSERT, are invalid.

5. So does this limit my ability to update the database through Called Procedures? No, you can still store, modify or erase records in the database. It's the called procedure code that handles this, not the invoking SQL commands.

6. Do called or table procedures have access to all information supplied in the SELECT statement's WHERE criteria? No. For the values of the parameters in the WHERE criteria of the invoking SELECT to make their way into the procedure, three things must be true:

   a. The value must be specified in an equal condition
   b. The condition mustn't contain a NOT operator
   c. The condition cannot be combined with any other equal condition that uses the OR operator.

7. What happens to the parameter values that don't pass into the called or table procedures? They are used by the SQL Engine to evaluate the returning rows. This final evaluation uses ALL conditions in the WHERE criteria.

8. Should the table or called procedure code ever change the value of one of the parameters sent into the procedure on an equal WHERE criteria? When a SELECT statement is used to execute a procedure, never change the value of any field listed in an equal condition within the WHERE criteria of the SELECT in the procedure. Why? The SQL Engine sends the values of these parameters into the procedure. The procedure is free to use these values, however, once the procedure completes it's work and control is returned to the SQL engine, SQL will evaluate the data returned from the procedure against the values in the original WHERE criteria and if you've changed these values in the procedure, the resultant rows returned can be altered. Let's say you have a WHERE criteria that says IN_DATE = 0 and in the code you code MOVE 20070827 TO IN-DATE, the SQL engine will eliminate that row from the result set since 20070827 is not equal to 0. GOTCHA!

Linda J. Casey, Managing Member, Run Right, LLC. Is a small business owner with 32 years in the software industry specializing in systems design, development and implementation, management of both application and infrastructure projects, training development and delivery and extensive vendor experience with both Cullinet and CA.

What am I doing wrong? continued from page 14

The second most common cause is a Cobol program going out of bounds in an ODO data item (occurs depending on) - or something similar.

HTH - cheers - Gary

Editor's note: in this particular case it was the first possibility – which in my experience is about 90% of the time when you get a DC171071 error.

Note to DBA's: "storage protect" will **not** trap this error. As this is happening inside an RBB it is in storage that is allocated to the task (and so won't be caught by storage protect). Also being (usually) somewhere in the middle of a storage allocation it will fall between the "storage framing" – and so you won't get the "STORAGE VIOLATION" on a #FREESTG – which is detected by overwritten "storage framing". Of course this can happen – but only when the record is near the end of the allocation and large enough to overwrite both a following RBE and the storage frame that follows it. Because of the number of variables involved this can be a very frustrating problem to diagnose and solve – partly because it might not be detected until the user exits the ADS application and the RBB's are freed as ADS tidies up.

MODERNIZE CA IDMS FOR SOA

For a fraction of the cost of conversion, you can modernize your existing IDMS databases and applications, and reuse them as enterprise resources in SOA and web based applications.

By Dave Ross

As a long time user, you have a significant investment in CA IDMS and get significant business value from your IDMS systems. These systems are core systems that your organization depends upon to run its operations. But, like many users, you may be wondering how, or even if, IDMS fits into a modern application environment based on a Service Oriented Architecture.

SOA and Legacy Applications

In a Service Oriented Architecture (SOA), business processes are implemented as services that are invoked using well-defined interfaces. These interfaces are independent of the platforms and communications protocols used by the service provider and consumer. Services may be delivered over a common communications path called an Enterprise Services Bus (ESB). Web Services are a standards-based implementation of SOA based on technologies such as XML, Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI).

Service-based applications are typically multi-tier applications that execute business logic on J2EE application servers such as WebSphere, WebLogic, JBoss, or the Windows .NET platform. The business logic is written in modern languages such as Java, C#, or Visual Basic. The user interface is graphical and often consists of web pages displayed on a PC, PDA, or even a cell phone. It may be implemented using technologies such as Servlets, Java Server Pages, or Active Server pages. Developers use distributed tools designed for this environment, such as Eclipse and Visual Studio.

Traditional on-line IDMS applications are single tier applications written in COBOL, CA ADS, or Assembler that execute business logic in a transaction server such as CA IDMS/DC or CICS. The user interface is set of menus and forms displayed on text mode 3270 terminals.

Legacy applications and databases, such as IDMS, can be reused as "enterprise resources" in the modern environment. These enterprise resources can be used directly by applications but can often be more effectively reused in new multi-tier applications if they are encapsulated as services. However, in either case the legacy applications must be made compatible with the new technologies.

There are two ways to make legacy enterprise resources compatible with service and web-based applications. Conversion vendors, competitors, and even people within your own organization might be recommending conversion from IDMS to another "more modern" DBMS. But converting IDMS applications to another database is expensive, and there is no guarantee that the converted applications will work as well as they did before. It is difficult to make a business case for spending a lot of time and money to replace a working application with a functionally equivalent one, especially when there is an alternative.

The alternative is to modernize your IDMS databases and applications. For a fraction of the cost of conversion, you can modernize your existing IDMS databases and applications, and reuse them as enterprise resources in SOA and web based applications.

Modernizing your IDMS databases and applications allows the next generation of developers to use the modern languages and tools that they already know to build new applications that access IDMS like any enterprise resource, using standard open application program interfaces. These applications can be deployed on industry standard J2EE or .NET application server platforms.

Modernizing IDMS

So, how do you modernize IDMS? What are the techniques that allow web and service based applications to access and update data in IDMS databases and to invoke business logic in IDMS applications?

A powerful modernization approach is to use CA IDMS SQL and CA IDMS Server. CA IDMS SQL can query and update existing network databases using SQL, and can also be used to invoke business logic running in CA IDMS/DC. CA IDMS Server provides the standard open interfaces that modern applications and tools use to issue SQL requests. It includes a JDBC driver that can be used to access IDMS from J2EE application servers and an ODBC driver that can be used with ADO.NET. Modern tools can be used to develop web services and applications that use SQL just as they do for other relational databases.

continued on page 17)

Modernize CA IDMS for SOA continued from page 16
Figure 1 illustrates an example of how a web application based on SOA might access an IDMS database.

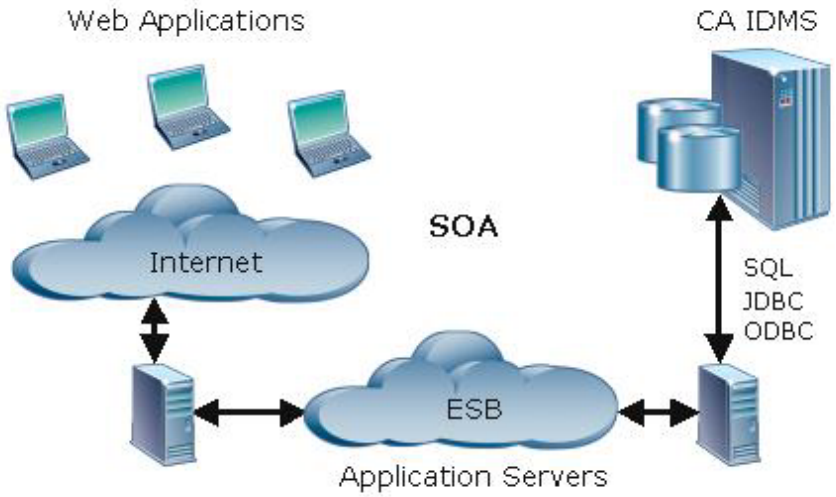


Figure 1. CA IDMS in a Service Oriented Architecture

In this example the user views the system as a web site accessed over the internet. This web site is hosted by a web server that uses Java Servlets to build dynamic HTML pages containing the customer's account information. It gets this customer information by invoking web services that are implemented on an application server running inside the organization's firewall. The web services use JDBC and SQL to retrieve the customer's data from the customer database. Some of this data is stored in an existing network database, while some of the data is calculated by a subroutine shared with an existing ADS application.

Now we will look at some of the modernization techniques that you can use today with CA IDMS SQL and CA IDMS Server, the native CA IDMS TCP/IP support, and tools from other vendors to access existing network databases and reuse application logic from web and service-based applications.

Network to Relational Mapping

Network to relational mapping allows SQL to be used to access existing network databases as if they were defined using SQL. This enables applications running on distributed platforms to use standard JDBC and ODBC interfaces to access these databases. This network to relational mapping is essentially the process of mapping:

- Network schemas to SQL schemas
- Records to tables
- Elements to columns
- Sets to referential constraints

You enable network to relational mapping by defining an SQL SCHEMA that identifies the network schema and database instance. All records in the schema are automatically accessible as tables and can be referenced using the SQL schema and network record names. The record's elements are mapped to columns. The COBOL data types used in the element definitions in the dictionary are mapped to the equivalent SQL types. Record names that contain hyphens must be enclosed in quotes when referenced in an SQL statement. Element names are translated to more "SQL friendly" names, with hyphens replaced by underscores, but you can override this by defining element synonyms for SQL.

This automatic mapping of records to tables enables the use of SQL to query most network records as independent tables. Tools can use standard JDBC and ODBC metadata functions to discover the network records as if they were SQL-defined tables. Some record structures, such as occurs depending and group items, cannot be represented as SQL elements.

Network sets are analogous to referential constraints. A referential constraint represents relationships between two tables in an SQL defined database by constraining the value of a foreign key in one table to reference (that is, to equal) the value of a unique primary key in another table. The referencing table is analogous to the member of a set and the referenced table is analogous to the owner. Joining tables based on the primary and foreign keys is analogous to navigating network set structures.

continued on page 18)

Modernize CA IDMS for SOA continued from page 17

Sets are not automatically recognized as constraints by SQL. Referential constraints require the primary key value to be duplicated in the referencing table, but in network databases the relationships between records are implemented implicitly with pointers, and the member record does not usually contain the key value that identifies the owner.

Techniques that can be used to solve the problems with mapping network sets and unsupported record structures to the relational model include SQL syntax extensions, the use of table procedures, and adding foreign keys to sets.

SQL Extensions

IDMS provides two extensions to SQL that can be used to navigate and update records in sets. Using these SQL extensions requires no additional IDMS programming and no changes to your network database, but, since they are non-standard SQL, most tools will not generate them automatically.

· The "WHERE <set name>" predicate can be used to join the owner and member records. This allows access to the member occurrences for a specific owner, and is analogous to navigating a set with GET NEXT requests. This technique can be used to query most network structures but is of limited use for updating records.

· The ROWID pseudo column is essentially a DBKEY that can be returned in a query. It is stable within a transaction and can be used to update or delete specific members of a set, although it is not useful when attempting to insert a record.

Table Procedures

A table procedure can make a network database structure appear to be an SQL-defined table that can be accessed using SELECT, INSERT, UPDATE, and DELETE statements. The table procedure encapsulates the native DML navigation of the network records and sets. Table procedures can also be used to access occurs depending records from SQL. Tools can use standard JDBC and ODBC metadata functions to discover table procedures as if they were SQL-defined tables.

Using table procedures requires no changes to existing applications or databases, but does require some IDMS programming. You can use CA IDMS QuickBridge to generate table procedures that navigate network structures. QuickBridge is a Windows application included with IDMS SQL that generates native DML navigation logic for elements and records that you select from the schema definition in the dictionary.

Foreign Keys

You can define primary and foreign keys for network sets, turning them into "referential sets" that are equivalent to referential constraints. Adding foreign keys is a very powerful technique that allows the use of standard SQL statements to INSERT, UPDATE, and DELETE rows in the owner and member records. There is no need to use IDMS extensions in SQL DML, since the IDMS SQL optimizer generates an access plan that navigates the network database and enforces referential integrity, just as it does for SQL-defined databases. Tools can use standard JDBC and ODBC metadata functions to discover the relationships between network records when they are defined as referential sets. Adding foreign keys allows an IDMS network database to look like any other relational database in the modern application environment.

You define a referential set by specifying the primary and foreign keys in the schema set definition. The primary key in the owner must be a unique system owned index or CALC key. In an IDMS database, this key will usually not exist as an element in the member record, so the database may need to be restructured to add the element, and the element must be populated with the value of the owner's key. Programs that store the member record must be modified to maintain the value of the foreign key elements.

Many records already have elements that can be defined as the primary key in the owner record of a set. If the member record already contains elements that can be used as the foreign key, all that is needed is a change to the schema. Code that issues native STORE and CONNECT requests may be isolated in a few modules, limiting application program changes. It may not be necessary to restructure an entire database, but only those records that will be accessed through SQL. In any case, it is easier to restructure a database than to convert it to another platform.

continued on page 19)

Modernize CA IDMS for SOA continued from page 18

Mapping Technique Tradeoffs

Table 1 summarizes the tradeoffs involved with using these network to relational mapping techniques.

	Standard SQL	New Programs	Application Changes	Restructure Required	Supports Sets
Syntax Extensions	No	No	No	No	Limited
Table Procedures	Yes	Yes	No	No	Encapsulated
Foreign Keys	Yes	No	Limited	Targeted	Referential Constraints

Table 1. Mapping Technique Tradeoffs

SQL extensions require no new programming or changes to applications, and work well for retrieval applications, but most tools won't generate them. Table procedures can be accessed with standard SQL and can query or update any set of records, but programming is required to navigate the database. Adding foreign keys may require restructuring the database and changing programs that store or connect records, but they are the most effective way to allow applications to use standard SQL to query and update network databases. These techniques are all compatible with each other, so you can mix and match them as appropriate for your environment.

Reusing Application Logic

So far we have focused on techniques for reusing IDMS databases, but you can also leverage your investment in IDMS by reusing existing business logic and invoking it from web and service based applications. This can include logic from on-line applications running in CA IDMS/DC or CICS as well as batch applications.

Two problems must be solved in order to reuse existing applications. The first, and most difficult, is separating the business logic from the presentation logic. The other is how to invoke the business logic from the modern application environment. Depending on the application, business logic may be invoked using a call interface or a presentation interface.

Call Interface

Using a call interface means extracting the business logic and calling it directly. It is usually not a trivial task to separate the business logic from the presentation logic in an application, but it can be done. Applications often have common subroutines that have no user interface, and these routines can be easily reused. Business logic extracted from batch programs can also be invoked using this approach.

Once the business logic is made into a callable subroutine, it can be invoked as an SQL procedure or by a socket program.

SQL Procedures

Any IDMS application code that has no user interface can be called as an SQL PROCEDURE. IDMS SQL can call map-less ADS dialogs directly. A small wrapper program may be needed to call routines written in COBOL, PL/1 or assembler.

Applications running in distributed application servers can use IDMS Server to call the procedure that invokes the business logic. Tools can use standard JDBC and ODBC metadata functions to discover the procedures and their parameters in the same way that they discover stored procedures on other databases. Once the business logic is exposed this way, modern development tools can be used to wrap the call to the procedure in a web service.

TCP/IP

A program running in IDMS can use the TCP/IP support in IDMS r16 to call a program running on another platform. This technique can even be used to invoke a web service from an application running within IDMS.

You can also use TCP/IP to invoke existing business logic that has been separated from the presentation logic. Since this requires developing a server interface to run in IDMS and a client interface to run on the distributed platform, customers often prefer to take advantage of the client-server infrastructure that is provided by IDMS SQL and Server.

Presentation Interface

The other approach to reusing on-line applications is to invoke the presentation interface, that is, to simulate a person entering data on 3270 screens. Third party products are available that use scripts to navigate through dialogs, reading and

(continued on page 20)

Modernize CA IDMS for SOA continued from page 19

writing the 3270 screen buffer, and exposing a path through an on-line application as a web service.

This approach completely avoids the logic separation problem and requires no changes to the existing applications, but the navigation scripts are sensitive to changes in the underlying application flow or maps. These products use CICS or a proprietary mainframe address space to invoke the dialogs and act as the web service provider.

In Conclusion

We have seen that there are a number of approaches to modernizing IDMS databases and applications that can be used separately or in combination. No matter which approach is used, modernization can often be done incrementally, since only a subset of a database or on-line application may be needed to build a particular web service.

Modernization can extend the life of your IDMS applications, enhance the value of your investment in CA IDMS, and free resources for new development that delivers business value for your organization.

Dave Ross is an Architect at CA and has worked on the development of CA IDMS for over twenty years, focusing on SQL, communications, and open access through industry standard interfaces.

VOTING RESULT IUA DARS

Craig McGregor, the newly appointed "DARS Man" on the IUA Board, has helped the Board to organise the DARS for voting on by the IUA membership later this year. Here is the list with the current rankings, after preliminary voting by the IUA Board: Highest (needs to be worked on), Medium, Lowest (does not need to be worked on – have not been included here).

Editors note: I have kept the "functionality available in.." and "Targeted for r17" DARS in the list as this is useful information in its own right. Watch the IUA Web Site for more news on the DARS voting. To conserve space I have dropped the "lowest" priority DARS.

IDMS DAR LIST
From February 14, 2005 to July 17, 2006

****

07/24/06 **Medium** -3-
PDT: ADS370 S9475
Description: Enhance ADSORPTS to produce a cross reference between ICMD# and
statement number if FDBLIST and PROCESSES are requested for a dialog with symbol tables off.

****

07/24/06 **Medium** -6-
PDT: IDMS 14748565/1
Description: NEED ENCRYPTION SUPPORT
need support for encryption of IDMS data.

****

07/24/06 **High** -8-
PDT: IDMS 14684045/1
Description: DISPLAY RELEASE & SERVICE PACK # at STARTUP When CA-Datacom initially starts up we generate the following msg: DB00215I - Advantage CA-Datacom/DB Database Ver: 11.0 at service pack: SP01 I would like to see CA-IDMS either modify the DC013005 msg or issue a new message that also reflects the rel & SP level that the CV is running.

****

07/24/06 -9-
PDT: IDMS 14667289/1 *** ***TARGETED FOR r17*** ***
Description: NEED TO BE ABLE TO DO ONLINE TRACING OF SQL STATEMENTS Need to be able to do tracing online of sql statement (in a cv environment)

****

07/24/06 **Medium** -10-
PDT: IDMS 14666000/1
Description: DAR REQUESTS SEGMENT NAME DEALLOCATE Currently DBA's can issue the commands:
VARY SEGMENT NAME ONLINE/UPDATE (PERMANENT)
VARY SEGMENT NAME RETRIEVAL/OFFLINE (PERMANENT)
It would help if we could also issue:
VARY SEGMENT NAME DEALLOC (so all files would de-alloc)

****

07/24/06 **High** -11-
PDT: IDMS 14633877/2
Description: SIMPLIFY SECURITY ON VDBA Currently, security for Visual DBA users can only be implemented through activating DB security (internal/external). This has impact for all other users of the CV(s) accessed by Visual DBA. Request is to add a kind of READ ONLY feature for Visual DBA. This feature would avoid the need to activate DB security. See same contact, issue #1 for initial request

****

07/24/06 -12-
PDT: IDMS 14621153/1
Description: UNLOAD; VIA INDEX; PERFORMANCE In general UNLOAD will just retrieve symbolic keys and record dbkeys during the start of the UNLOAD process and not extract any data records. However any records that are stored an VIA the index are extracted during index processing. This was added back in the early '90s to better facilitate the efficient rebuilding of these VIA clusters. The utility still sweeps the area looking for any other record types or relationships that might need to be included in the UNLOAD process.

(continued on page 21)

Voting Result IUA DARS continued from page 19

If the only thing in an area is a cluster of records stored VIA an index there is an unneeded pass through the data area. I would suggest a DAR asking for this enhancement. You imagine a simple DB area with 1 record type and 1 system owned index then Unload will first walk the entire index and the records and then it will sweep the area. The resulting output file could be produced at a fraction of the I/O if the index sweep was replaced by a return key sweep.

****

07/24/06 -13-
PDT: IDMS 14584049/1 *** ***TARGETED FOR r17*** ***
Description: NEED STARTUP PARM TO SPECIFY
Need a startup parm that allows you to specify an operating system subpool to use for the getmains that IDMS does at startup to build the IDMS system.

****

07/24/06 -14-
PDT: IDMS 14582819/1 *** ***TARGETED FOR r17*** ***
Description: NEED A NEW STARTUP PARM FOR MT
For r17, a new startup parm is needed to be able to set the multi-tasking queue depth rather than requiring the user to vary the queue depth once the system has started.

****

07/24/06 -15-
PDT: IDMS 14581119/2
Description: COMP-5 SUPPORT
Would like IDMS to support the Computational-5 usage mode for elements defined in the dictionary with IDD or IDMSDDDL.

****

07/24/06 -16-
PDT: IDMS 14552904/1 *** ***TARGETED FOR r17*** ***
Description: IDMSLOOK BIND SQL
The is for IDMSLOOK to support displaying an SQL FIB as a utility would generate it like Unload would. There is no AM, so LOOK would have to build the FIB and then display it. If the utility ran into an error based upon the FIB that is built, it would be nice to know what the utility is looking at.

****

07/24/06 **Medium** -17-
PDT: IDMS 14530102/1
Description: SQL access to member of set with foreign key omitted.
Can you include a set member record option of "FOReign KEY is OMItted missing-element-name", where missing-element-name is the name (or names if composite key) that would be given to the foreign key element if it was included. Behavior should be equivalent to that provided if the foreign key was present in the member record. Consider A —> B where A is stored calc based on A-KEY and B is related to A by the set A-B.

If B contains B-A-KEY as a foreign key, we can already specify for the A-B set: "SET NAME IS A-B OWNER IS A ... PRIMARY KEY IS CALC ..... MEMBER IS B FOREIGN KEY IS (B-A-KEY)". If B does not contain a foreign key, we would like to be able to specify for the A-B set that: "SET NAME IS A-B OWNER IS A ... PRIMARY KEY IS CALC ..... MEMBERIS B FOREIGN KEY IS OMITTED (B-A-KEY)".

B-A-KEY would be a virtual column that the system would materialise so that instead of a "WHERE set-name" clause in the SQL DML statement the user/developer could specify "WHERE ownerrec.primary-key = memberrec.missing-element-name".

Selecting the missing-element should return the values of the primary key of the owner of the set. Inserting into the missing-element should connect the member to the owner record having a primary key equal to the inserted value. Updating the missing-element should disconnect and reconnect the member to the appropriate owner (if Manual). Inserting or setting a null value into the missing-element could leave the member disconnected from the set (if Optional Manual). Including a where missing-element equals clause should result in access to the member record via the set owner primary key.

The existing functionality, where the set member also contains the foreign key, has been tested against both MA and OM sets - and the system behaves as one would expect for selects, inserts, updates and deletes. The result of implementing this request would be that network databases could be accessed with "pure SQL", for selects, inserts, updates and deletes, and not have to rely so heavily on IDMS/SQL extensions.

****

07/24/06 **Medium** -18-
PDT: IDMS 14530103/1
Description: Primary key of unique member of set
Can you include an owner record option of "PRImary Key is set-name", where the owner record is itself a member of another named set. The named set should have a primary key defined for its owner. The named set should have a sort key with duplicates not allowed. The combination of the named set owner primary key and the named set sort elements would be the relational primary key for this owner record. This would support relational access to networks that are more than 1 set deep.

Consider A —> B —> C where A is stored calc based on A-KEY and the A-B set is sorted (DN) on B-SORT-KEY. We can already specify for the A-B set: "SET NAME IS A_B OWNER IS A ... PRIMARY KEY IS CALC ..... MEMBERIS B FOREIGN KEY IS ..(existing or new option as above).".

We would like to be able to specify for the B-C set that: "SET NAME IS B-C OWNER IS B ... PRIMARY KEY IS SET A-B .... MEMBER IS C FOREIGN KEY IS ..(existing or new option as above).".

****

(continued on page 22)

Voting Result IUA DARS continued from page 21

07/24/06 **High** -19-
PDT: IDMS 14504597/2
Description: CANNED SECURITY REPORT USERS/ RESOURCES/PRIVILEGES
Currenty we run against the Security DATABASE with OCF commands to get all users place in a flat file, then get all security modes place in flat file, and get record of user connected to security again place in a flat file.
Would like "canned security reports" that give this type of info: USERs and their RESOURCES; USERS and their PRIVILEGES Also by RESOURCES and all USERS; PRIVILEGES and all USERS
****

07/24/06 -23-
PDT: IDMS 14492564/2
Description: MASKING FOR RESOURCE NAMES
Would like to be able to use masking in the definition of resources.
For example: ALTER RESOURCE CATEGORY CAT_TOOLS_DMLO
ADD RUNUNIT DICT05.*.USDMAIN0;
This allows permission for rununits for any subschema accessing dictionary DICT05 using USDMAIN0.
****

07/24/06 **Medium** -26-
PDT: IDMS 14439607/2
Description: SUPPORT TEST COBOL COMPILER OPTION
Would like IDMS to support the TEST Cobol compiler option because it is required by OSMVS debugging feature provides source level debugging.
****

07/24/06 -27-
PDT: IDMS 14395853/2
Description: NEED WARNING FOR EXTEND SPACE
When you have done an EXTEND SPACE operation for an area, the only indication that the area has been extended is a comment. If PRIMARY SPACE is subsequently altered while doing additional maintenance to the area this attribute will be lost and the client may not be aware that they have created a situation that can cause logical database corruption.

If an area is currently in the EXTENDed state and an ALTER would cause the loss of the attribute, the command should be disallowed and an appropriate warning should be issued.

Some clients make AREA changes by DROP/CREATE after displaying the area AS SYNTAX and making their changes. This would also cause the loss of the EXTEND attribute so please create a special form of DROP AREA that must be issued when area has been EXTENDed, to insure client knows the consequences of the DROP AREA under these circumstances.
****

07/24/06 **Medium** -28-
PDT: IDMS 14360167/2
Description: AREA PAGE RANGE ERUS
For a database transaction in same address space as application program, an assembler program can obtain page range, radix & pagegroup info by going from SSCTRL to VB50 and VARS. For ERUS programs this cannot be done because the VB50 is in CV's address space. Please add code to return area page-range, radix & pagegroup in SSCTRL at READY AREA time so external application program can access this info.
****

07/24/06 **Medium** -30-
PDT: IDMS 14279234/1
Description: SYMBOLICS IN DSNAME
Would like it to be possible to use symbolics in the DSNAME within the FILE definition in a Segment? If not, would it be possible to override the DSNAME at the DMCL level? An example would be to be able to use a symbolic name like &QUAL as part of the DSNAME definition in the FILE statement. Then you could specify &QUAL = 'IDMS.TEST' on one DMCL statement and &QUAL='IDMS.PROD' on another.
****

07/24/06 -32-
PDT: IDMS 14238685/2 ***
FUNCTIONALITY AVAILABLE IN r16 SP4 *
Description: EXIT 14 FUNCTIONALITY IN SQL
Client is putting a User record to the journals that identify each USER who is performing updates. He uses EXIT14 to create a User journal record at bind

What they need is a replacement for EXIT14 when dealing with SQL. Would also like to see user id added to journal records.
****

07/24/06 **Medium** -33-
PDT: IDMS 14234018/2
Description: High Water Mark OF SYSLOCKS VALUE
Client would like IDMS to keep track of the HWM of SYSLOCKs used. At the moment there is no way to tell when they are getting close to the SYSGEN specified value.
****

07/24/06 **High** -34-
PDT: IDMS 14234018/3
Description: SYSLOCKS OVERFLOW MEESAGE
A message should be put out when SYSLOCKS go into overflow storage as a warning. At the moment there is no warning and potentially a program could cause all storage to be used up and hang the CV.
****

(continued on page 23)

Voting Result IUA DARS continued from page 22

07/24/06 -35-
PDT: IDMS 14226975/2 ***
***FUNCTIONALITY AVAILABLE IN r16 SP4 ***
Description: INCLUDE USERID IN JOURNAL
Due to new audit standards that require tracking of updates by user, please enhance the BGIN journal record to include USERID.
****

07/24/06 -37-
PDT: IDMS 14186392/2 *** *TARGETED FOR r17* ***
Description: Support "D" IN COLUMN 7 AS COMMENT
The Cobol compiler treats a 'D' in column 7 as a comment if debugging mode is turned off. Client would like IDMSDMLC to also treat a 'D' in column 7 as a comment if debugging is off instead of expanding the DML statement to Cobol code.
****

07/24/06 -38-
PDT: IDMS 14113390/1
*** *FUNCTIONALITY AVAILABLE IN r16 SP4* ***
Description: AUDIT DATABASE CHANGES
Need audit info about some of the sensitive information that they have. Objective is to have one userid, check what the user did on any specific moment in time. To know what records the user viewed, modified or erased along with the information before and after.In other words for audit purposes check anything that is changed in the database.
****

07/24/06 -42-
PDT: IDMS S8407 *** *TARGETED FOR r17* ***
Description: Extend TUNE INDEX with intermediate SR8 adoption.
****

07/24/06 **Medium** -43-
PDT: IDMS S8888
Description: Within CICS or PLI applications against a network database, you can use ACCEPT IDMS-STATISTICS INTO DB-STATISTICS, would like the ability to issue a similar command against an SQL defined database.
****

07/24/06 -44-
PDT: IDMS S8905
*** *FUNCTIONALITY AVAILABLE IN r16 SP4* ***
Description: Need to be able to identify who changed a record in the database by having the user ID on the journal AFTR records.
****

07/24/06 -45-
PDT: IDMS S8912
*** *FUNCTIONALITY AVAILABLE IN r16 SP4* ***
Description: Need to be able to relate the user ID with database activity by having the user ID inserted into the appropriate journal records for Sarbanes-Oxley.
****

07/24/06 -46-
PDT: IDMS S8959 *** *TARGETED FOR r17* ***
Description: Need a fast journal format option that only overwrites the journal header records - with large journals (198,000 pages - 10K blksize) it currently takes 20 - 30 minutes, where just overwriting the headers takes < 10 seconds.
****

07/24/06 -47-
PDT: IDMSDC 14950367/1
Description: RESOURCE TIMEOUT RETURNS NEXT TASK ABEND
If a non-interactive terminal gets into resource timeout (ie. LD000nnn) and it has no PTE. If there is a user resource timeout program, and it does a DC RETURN NEXT TASK CODE 'BYE' to goto RHDCBYE, a program check (3964) will occur in RHDCMSTR.
****

07/24/06 **Medium** -48-
PDT: IDMSDC 14605173/1
Description: AVOID CV-ABEND With S913
Request an enhancement of dealing with a RACF-produced S913-error when new files that should have been registered in RACF but weren't. Requests that the Central Version does not abend when RACF submits a S913 error to it. The affected files should be marked in error but for the others processing should go on.
****

07/24/06 -51-
PDT: IDMSDC 14515046/1
*** *TARGETED FOR r17* ***
Description: SNAP OPTIONS IN SYSGEN
The option to enable/disable snap system, snap system photo, snap task and snap task photo should be incoporated in the IDMS system generation parameters to avoid having to submit a UCFBTCH job at start up to:
VARY SNAP SYSTEM ON/OFF
VARY SNAP SYSTEM PHOTO/NOPHOTO
VARY SNAP TASK ON/OFF
VARY SNAP TASK PHOTO/NOPHOTO
****

07/24/06 -52-
PDT: IDMSDC 14509611/1
*** *TARGETED FOR r17* ***
Description: Make IDMSINTC THREADSAFE
****

07/24/06 **High** -54-
PDT: IDSERV 14138907/3
Description: HIBERNATE DIALECT FOR IDMS
Request that a Hibernate dialect be developed which works with IDMS. Hibernate is one of the JBoss suite of products. It is an example of Object-Relational Mapping (ORM) which lets Java programmers refer to data as Java objects as opposed to rows and columns. Hibernate also is a persistence mechanism similar to Container Managed Persistence and EJBs.

(continued on page 24)

Voting Result IUA DARS continued from page 23

A "dialect" is a Java class which defines DBMS-specific SQL syntax. The use of Hibernate and a dialect hides the complexities of a DBMS' SQL implementation from the Java programmer. For more information on Hibernate, see www.hibernate.org.

Also note that the Ingres folks are developing a new dialect for their product. There already was a very basic dialect which was most likely developed by a non-CA person, but the new dialect will be given back to the Hibernate group and made universally available once completed.

****

07/24/06 **Medium** -55-
PDT: IDSERV S8889
Description: Would like HIBERNATE support for IDMS

****

07/24/06 -56-
PDT: JNLA 14674550/1 ***TARGETED FOR r17***
Description: ENHANCED RANKING REPORT
DAR request for an enhanced Journal Analyzer Ranking Report

****

07/24/06 -58-
PDT: JNLA 14616763/1 ***TARGETED FOR r17***
Description: Journal Analyzer needs to HANDLE CUSTOMIZED Presspack DCTS

****

07/24/06 -59-
PDT: SQLOPT 14765604/2
Description: SQL INDEXES AND NULL VALUES
SQL index defined as UNIQUE INDEX using a particular column, for which the column is defined without the NOT NULL specification, as such allowing NULLS. INSERT in the related table with a value of 'NULL' for the index column is rejected with DB001058 Uniqueness violation on this INDEX. Request that UNIQUE INDEXES allow duplicates' on the involved column(s) if NULL values are used.

In DB2, there is an option on the CREATE INDEX statement which says 'UNIQUE WHERE NOT NULL' telling the index to consider multiple NULL columns as unequal.

****

07/24/06 **High** -60-
PDT: SQLOPT 14683815/5
Description: GET STATISTICS IN SQL PROGRAM
An equivalent 'verb' as the ACCEPT STATISTICS for nonSQL currently does not exist for SQL programs. As such, it is more difficult to monitor their programs from a performance point of view.

****

07/24/06 **Medium** -63-
PDT: SQLOPT 14549906/2
Description: DBA MAINTENANCE ON SQL TABLES
Several modification actions on existing SQL tables require to drop and recreate the table. This can have a major impact and cause a lot of work, depending on the involved database structure. If utilities like real IMPORT/EXPORT of table data, RESTRUCTURE on SQL databases (as exist for nonSQL databases) would be available, this would be a great help to accomplish these modifications.

Another point regarding SQL databases within IDMS which causes trouble is the fact that for UNLINKED CONSTRAINTS, require the FOREIGN KEY to be a CALC or INDEX key.

****

07/24/06 -64-
PDT: SQLOPT 14505329/2
Description: EXPLAIN STATEMENT AND QUOTES
EXPLAIN of an SQL statement containing quoted literals requires that all of these literals are placed between double quotes (") or syntax errors are returned : DB006001 T350 C-4M330: Unrecognizable token
This makes it difficult to use, especially when cut&pasting SELECT syntax from Visual Express, Visual DBA ....

****

07/24/06 -66-
PDT: SQLOPT 14292316/1
Description: DB006001 UNION IN UPDATE
An UPDATE statement of the form UPDATE .. SET COL = query requires that the query must be a subquery (a single query) as opposed to a query expression or you get a syntax error DB006001 on the UNION.

****

07/24/06 -67-
PDT: SQLOPT S7215
Description: Error when creating a constraint on an existing table.

****

07/24/06 -68-
PDT: SQLOPT S8124
Description: Question on the requirement of an index on a foreign key in a constraint.

****

07/24/06 **Medium** -69-
PDT: SQLOPT S8262
Description: Would like integration of the EXPLAIN command into Visual DBA (like what Oracle provides or TOAD from Quest Software).

IUA BOARD

International Chair
Terry Schwartz
Company: Perot Systems
Address: PO Box 269005
Phone: 972 577-3722
Email: terry.Schwartz@ps.net

Secretary/Treasurer
Email Coordinator
Bob Wiklund
Company: Tiburon Technologies
Address: 17101 W. Gable End Lane,
Surprise, AZ 85387
Phone: 623 594-6022
Email: bob_wiklund@tiburontech.com

International Vice Chair
Contributed Software Librarian
Laura Rochon
Company: Ajilon Professional Services
Address: 22 Jolliet, St-Bruno,
Quebec J3V 4Z1 Canada
Phone: 514-943-8290
Fax: 450 441-6880
Email: l.rochon@videotron.ca

European IUA Representative
Steve Rundle
Company: British Telecom BT Group plc.
Address: PP2B33 Angel Centre,
403 St. John Street, London
EC1V 4PL UK
Phone: +44 (0)20 7777 6920
Fax: +44 (0)20 7777 6921
Email: steve.rundle@bt.com

Board Member
Craig McGregor
Axciom
craig.mcgregor@acxiom.com

Board Member
Diane Montstream
Allen Systems Group
diane.montstream@asg.com

Board Member
Jan Rabaut
jan.rabaut@dexia.be

Editor
Gary Cherlet
Justice Technology Services
South Australian Department of Justice
cherlet.gary@saugov.sa.gov.au

Desktop Publishing
Rebecca Shaw 404 377-6982
shawrh@bellsouth.net

IUA Connections is a bi-annual publication of the CA-IDMS Database and Applications User Association (IUA). It is designed to promote its members' objectives. *IUA Connections* is not responsible for the opinions expressed by its writers and editors.

Information User Association
401 N. Michigan Ave.
Chicago, IL 60611-4267
Phone: 312/321-6827
Fax: 312/245-1081

Internet: iua@iuassn.org
www: http://iuassn.org