Using the New MCPSQL

Paul Kimpel

2017 UNITE Conference

Session MCP-4055
Wednesday, 1 November 2017, 11:00 a.m.

Paradigm Corporation

Presentation Topics

- What is MCPSQL?
  - History
  - Features and Enhancements
- Installing and Configuring MCPSQL
- MCPSQL Java-based Client Utilities
  - RDC – Relational Design Center
  - QDC – Query Design Center
- Host program interfaces
  - ModLang – Module Language
  - CLI – Command Level Interface
- New ODBC Driver
- A Bit About Performance

What is MCPSQL?

- Officially, the "Relational Database Server for ClearPath MCP Query Processor"
- Dual aspects
  - A relational data base system built on top of DMSII
  - An SQL engine for existing DMSII data bases
- Entirely MCP-based and DMSII-oriented
  - No separate Windows-based processor
  - Accessible directly from MCP applications
  - Accessible from external systems over TCP/IP
- Derived from DMSQL (2006) and SIM / SQLDB / DMS.View (mid-1990s)

About this Presentation

- MCPSQL is an extension of DMSQL
  - Same look and feel
  - Same features, plus new ones
- This presentation builds upon two earlier UNITE talks for DMSQL
  - 2006 – *Using DMSQL*
    http://www.digm.com/UNITE/2006/#MCP3023
  - 2008 – *DMSQL Query Capabilities and Performance*
    http://www.digm.com/UNITE/2008/
- Refer to these for background and details

MCPSQL Features

- ANSI-92 SQL compliance
  - Entry level X3.135-1992 feature set
  - With some nice extensions
- Language support
  - COBOL-74, COBOL-85
  - Algol
  - Java (via JDBC)
- Native APIs
  - Module Language (ModLang)
  - Call Level Interface (CLI)
  - JDBC driver for Java
  - ODBC driver

New Features Since DMSQL

- ODBC Driver (Windows and Linux)
- Query retrieval limits (only one per query)
  - Select **top** *n* [**skip** *n*]
  - Select … **limit** *n* [**offset** *n*]
- Additional string and numeric functions
  - ISNULL, NULLIF, NVL, NVL2, COALESCE
  - Plus some aliases for existing string functions
  - Support for DMSII DATE, TIME, and DATETIME items
- Case expressions
- Derived tables (!)
- Stored procedures (!)

Features – Case Expressions

- Conditional selection of a value
- Used in SELECT, WHERE, GROUP, etc.
- Two forms:
  - `case when <condition> then <expression>`
    `when <condition> then <expression>`
    `. . . else <expression> end`
  - `case <expression>`
    `when <value> then <expression>`
    `when <value> then <expression>`
    `. . . else <expression> end`

Paradigm MCP-4055 7

Features – Derived Tables

```
/* Determine if there are any Spend Group/Bill-to
   inconsistencies */
select CUSTBILLTO, BILLNAME, count(*) as nbr
from (
    select distinct CUSTSUBSYS, CUSTBILLTO, CUSTSPNDGRP
        from OECUST
        where CUSTSUBSYS=1 and CUSTCUSTSTAT='A'
        ) as bill2s
left join OEBILL on BILLSUBSYS=CUSTSUBSYS and
                    BILLBILLTO=CUSTBILLTO
group by CUSTBILLTO, BILLNAME
having count(*) > 1
order by BILLNAME, CUSTBILLTO
```

Paradigm MCP-4055 8

Installing and Configuring MCPSQL Software

System Requirements

- MCP 16.0 or later
  - Unbundled and separately licensed in MCP 16-17
  - Bundled in IOE starting with MCP 18.0
- Java JRE 1.6 or later (for RDQ & QDC)
- Strongly recommend you install with SIMPLEINSTALL or Install Center
  - ***Note:*** **`SYMBOL/MCPSQL/PROPERTIES`** is not installed by default on MCP 18
  - This is a required file
  - May need to copy from release media manually

MCPSQL DSS Commands

- **`NA MCPSQL HELP`** [<command>]
- Controlling the MCPSQL server
  - **`NA MCPSQL +`**
  - **`NA MCPSQL -`**
  - **`NA MCPSQL QUIT NOW`**
- Diagnostic commands
  - **`NA MCPSQL STATUS`** [<worker id>]
  - **`NA MCPSQL LOG`** [**`RELEASE`**]
  - **`NA MCPSQL TRACE`** [**`+`** / **`ON`** / **`-`** / **`OFF`** / **`RELEASE`**]

Paradigm

MCP-4055 11

Client Software Installation

- Two utilities for Windows environments
  - Relational Design Center (**RDC**)
  - Query Design Center (**QDC**)
- Requires Java 1.6 or higher on workstation
- Install from the MCP INSTALLS share or download from support.unisys.com
  - Standard Windows MSI file
  - Linux versions (SUSE, Redhat) are also available
  - By default on Windows
    - Only the QDC is installed
    - Must explicitly select RDC to install it

Paradigm

MCP-4055 12

Preparing a Data Base for MCPSQL

- Two alternatives
  - Define a new relational schema using SQL DDL
  - Create a schema from an existing DMSII data base
    - Must be redone after every DASDL update
- Overview of steps:
  - Update Resources and Users in ***MCPSQL/CONFIG**
  - Use RDC to establish relational schema and catalog
  - Optionally, use RDC to
    - Assign alias names to tables, indexes, columns
    - Suppress columns, map columns to dates
    - Add relational integrity constraints for tables
    - Establish **GRANT** security
    - Define SQL views

Paradigm MCP-4055 13

*MCPSQL/CONFIG File

- Global configuration for MCPSQL
  - Must be under same usercode and family as **SYSTEM/MCPSQL/CONFIG** codefile
  - CANDE **SEQDATA** format
- Configuration commands
  - Version control:
    - **RELEASE**
  - Access Control:
    - **USER**
    - **PROGRAM**
    - **DEFAULT**
  - Data base definition:
    - **RESOURCE**

Paradigm MCP-4055 14

Sample Release Specs

```
RELEASE MCPSTD (PACK=DISK)

RELEASE MCP11FIXED = MCPSTD (PACK=TEMP)
  SL MCPSQLPARSER =       *SYSTEM/MCPSQL/011-014/PARSER

RELEASE DIAGNOSTICS (PACK=MCPTEST)
  SL MCPSQLDRIVER =       *SYSTEM/MCPSQL/DRIVER
  SL MCPSQLSUPPORT =      *SYSTEM/MCPSQL/SUPPORT
  SL MCPSQLDMSIIMAPPER = *SYSTEM/MCPSQL/DMSIIMAPPER
  SL MCPSQLADMIN =        *SYSTEM/MCPSQL/ADMIN
  SL MCPSQLSCODESUPPORT =*SYSTEM/MCPSQL/SCODESUPPORT
  SL MCPSQLPARSER =       *SYSTEM/MCPSQL/PARSER
  SL MCPSQLCATALOG =      *DESCRIPTION/SQLDIR/
                                  MCPSQL-CATALOG
  SL MCPSQLSCODE =        MCPSQLSCODE
  SL MCPSQLPROCEDURES =   *SYSTEM/MCPSQL/PROCEDURES/
                                  LANGUAGE/DIAGNOSTIC
  SL MCPSQLTEMPLATE =     *SYMBOL/MCPSQL/PROCEDURES/
                                  TEMPLATE
```

Paradigm

MCP-4055 15

Configuring User & Program Access

- **USER**
  - Specifies MCP user capabilities within MCPSQL
  - Users optionally are associated with a Release
  - Users optionally have a limit on data base operations
- **PROGRAM**
  - Specifies MCP codefile data base operation limits
  - Overrides any applicable **USER** limits when this program is used
- **DEFAULT** specifies a Release and limit for all users not otherwise mentioned

Paradigm

MCP-4055 16

Sample USER & PROGRAM Specs

```
USER PAUL = MCP11TEST
USER FRED LIMIT 1500 = MCPSTD
USER PROD, SYSTEMS, MSMITH
    LIMIT UNLIMITED = MCPTEST
USER DEMO LIMIT 500 = MCPSTD


DEFAULT LIMIT 5000 = MCPSTD


PROGRAM (PROD)OBJECT/GL/EXTRACT ON FINANCE
    LIMIT=UNLIMITED
PROGRAM (TEST)OBJECT/SQL/DEMO ON WORKPACK
    LIMIT=2000
```


Configuring Resources

- Resources define the entities MCPSQL can open and access
  - Each represents 1–5 DMSII or SQL data bases
  - Optionally specifies a CCS for the resource
  - Separate access modes per data base
    - **INQUIRY**
    - **UPDATE**
- Queries can operate across data bases that are *within* their common resource

Sample RESOURCE Specs

```
RESOURCE AVRDB =
    (ONLINE)AVRDB ON OPS


RESOURCE TESTDB = CCS SPANISH
    (BETA)FINDB ON WORKPACK (MODE=UPDATE)


RESOURCE BUNCH =
    (PROD)GLDB ON GLPACK (MODE=INQUIRY),
    (PROD)ARDB ON AR01 (MODE=UPDATE),
    (PROD)APDB ON FINANCE (MODE=UPDATE)
```


Establishing the Configuration

- After updating ***MCPSQL/CONFIG**, you must inform MCPSQL the file has changed
- From CANDE or MARC
  - Run ***SYSTEM/MCPSQL/CONFIG**
  - Program verifies **CONFIG** file syntax
  - Writes **CONFIG** file name and status to a remote file
- Changes to the configuration:
  - Take effect the next time a connection is established
  - Not visible to any currently-connected users

Using the Relational Design Center (RDC)

- Imports SQL DDL or DMSII schemas
- Manages entire SQL schema update process
- Establishes aliases and date items
- Interfaces to MCP-resident MCPSQL schema administration software

Creating a Schema from DMSII (1 of 3)

- Run the Relational Design Center
  - Add a server, if necessary
  - Right-click server, select "**Import Relational Schema**"
  - Fill in
    - Data base name (i.e., title – control file location)
    - Usercode and password
    - Description file title [optional]
    - Access control – check box if is to be read-only
    - Click "**Import**"
  - Errors are reported in the "Output" window
- The default schema is now established

Creating from DMSII (2 of 3)

- If desired, modify the default schema
  - Assign name aliases to tables, indexes, columns
  - Hide tables, indexes, columns
  - Create date columns from other columns
  - Add referential integrity constraints for tables
  - Add **GRANT** security
  - Create views
- When finished with schema updates
  - Right-click data base and select "**Apply Schema Modifications**"
  - Relational mapping will be stored on MCP host as **DESCRIPTION/**<dbname>**/MCPSQL-CATALOG**

Creating from DMSII (3 of 3)

- Mapping of dates creates virtual columns
  - Date columns can be updated
  - Original columns are marked read-only in MCPSQL
  - Can hide original columns in MCPSQL if desired
- Notes:
  - DASDL must specify **DMSUPPORT**, **ACR**, and **DMRECOVERY** titles with explicit usercode and family
  - DASDL must specify **INDEPENDENTTRANS** option
  - For data update, DASDL must specify the **REAPPLYCOMPLETED** option
  - If using guardfiles, give **SYSTEM/MCPSQL/WORKER** appropriate access to data base files
  - If no **GRANT**s are specified, all users get full access

An Alternate to the RDC

- RDC is just a GUI front end to schema utilities that run in the MCP environment
- SQL data bases and DMSII schema imports can also be maintained using **`*SYSTEM/MCPSQL/ADMIN`**
  - Documented in the MCPSQL *Installation and Operations Guide* and *Programming Guide*
- Basic mapping for a DMSII schema:

```
RUN *SYSTEM/MCPSQL/ADMIN(
  "SQLVIEW (MYUSER)MYDBNAME ON MYPACK:" &
  "ACCESSCONTROL=UPDATEOK, " &
          "STATISTICS, LOCKEDFILE");
```


Using the Query Design Center (QDC)

- Workstation tool to test and analyze DML syntax
- Useful for ad-hoc query and update
- Supports a "query-by-example" (QBE) mode
- Can save both query text and result data as files

Query Design Center (QDC)

- Java GUI to develop and test SQL DML
  - Somewhat like Microsoft's Query Analyzer for SQL Server or MSQRY32.exe for ODBC
  - Can connect to multiple resources simultaneously
  - Supports multiple simultaneous query windows
  - Supports **`SELECT`**, **`UPDATE`**, **`INSERT`**, **`DELETE`**
  - Supports **`COMMIT`**/**`ROLLBACK`** and isolation levels
- Three main modes
  - **Analyze Query** – enter SQL directly and execute
  - **Design Query** – build SQL using QBE wizard
  - **View Catalog** – browse relational schema

Paradigm

MCP-4055 27

QDC Analyze Query Mode

- Select File>Connect>Analyze Query…
  - Fill in server, user, and resource info
  - Opens a two-pane sub-window
  - Enter SQL text (or highlight part of it) in top window
  - Press F5 to execute, F4 to parse only ("prepare")
  - View tabular result set in bottom window
- Options
  - Save and reload SQL query text
  - Save result set as CSV, TSV, or columnar text
  - Select debugging output (Qgraphs/Qdumps)
  - Select query isolation level
- Demo

Paradigm

MCP-4055 28

QDC Design Query Mode

- Select File>Connect>Design Query…
  - Fill in server, user, and resource info
  - Opens a four-pane sub-window
    - Table and join diagram
    - **Q**uery-**B**y-**E**xample column/sort/condition grid
    - Generated SQL text
    - Tabular result set
  - Press F5 to execute or Alt+P to parse
- Save/reload options same as Analyze Query – no debug or isolation-level options
- Demo

Paradigm

MCP-4055 29

QDC View Catalog

- Select View>Catalog…
  - Enter server and usercode/password info
  - Opens a two-pane browse sub-window
- Basics
  - Click on items in the left-hand tree pane
  - View item details in right-hand pane
- Advanced
  - ***Right***-click on tables in the left-hand pane
  - View more detailed information on columns, primary keys, foreign keys, statistics in a separate window
- Demo

Paradigm

MCP-4055 30

Transactions in the QDC

- By default, each **`UPDATE`**, **`INSERT`**, or **`DELETE`** is a separate transaction
- Select Options>Manual Commit Mode…
  - Supports multiple SQL statements as one transaction
  - Enables **`COMMIT`** and **`ABORT`**/**`ROLLBACK`** statements
  - Enables creation of and rollback to "savepoints" within a larger transaction
- Transaction control commands are also available by right-clicking in the QDC SQL pane

Paradigm

MCP-4055 31

QDC Hints

- Query isolation level
  - Default is **`MEDIUM`** (locks rows on read)
  - Usually more efficient if set to **`LOW`** (dirty reads)
- **`qdc.properties`** file
  - Controls GUI configuration, including display font
  - Usually stored under
    `C:\Documents and Settings\<user name>\.mcpsql\qdc.properties`
- No way to stop a query once started, except by File>Disconnect

Paradigm

MCP-4055 32

Alternative to the QDC

- Run ***SYSTEM/MCPSQL/DMQUERY** from CANDE or MARC
  - Must be run from the MCP host having the data base
  - Uses a remote file interface
  - Documented in *Installation and Operations Guide*
- Open the data base:

```
OPEN DMDEMO (USER=DEMO, FAMILY=TEST,
   MODE=UPDATE)
```

- Commands:
  - **SELECT**, **UPDATE**, **INSERT**, **DELETE**
  - Transaction control: **COMMIT**, **ROLLBACK**, …
  - **DEFINE**, **DO**, **SET** options, a few others

Paradigm

MCP-4055 33

Using the Module Language

Invoking SQL From an MCP Program

- The DMSII API is easy and convenient
  - Record-at-a-time retrieval
  - Record and field layouts are known to the compiler
- SQL results sets are more difficult to use
  - Rows and columns are determined by the query
  - Fields are not known to the compiler in advance
  - SQL values must be converted to host language types
  - A more dynamic programming interface is needed
- Three SQL APIs for MCP host programs
  - Module language [ModLang] (COBOL, Algol)
  - Call Level Interface [CLI] (COBOL, Algol)
  - JDBC (Java only)

The Module Language (ModLang)

- Based on ANSI SQL-92 Module Language
  - Standard COBOL module type
  - Unisys-specific Algol module type
- Modules define two types of entities
  - Cursors
  - Procedures
- Each module compiles to an MCP server library program
  - Generates a wrapper for lower-level calls to MCPSQL
  - Host programs call procedures as normal library subroutines, passing parameters

Sample Cursor Declarations

- Invariant query

```
DECLARE CURSOR C1 FOR
  SELECT NAME, DOB, ZIPCODE FROM PERSONS
  WHERE STATUS='P' AND GENDER IS NULL
  ORDER BY ZIPCODE, NAME
```

- Parameterized query

```
DECLARE CURSOR C2 FOR
  SELECT NAME, DOB, ZIPCODE FROM PERSONS
  WHERE STATUS=:QSTAT AND GENDER IS NULL
  ORDER BY ZIPCODE,NAME
```

Parameter

Paradigm

MCP-4055 37

Sample ModLang Procedures

```
PROCEDURE C2_OPEN (:QSTAT CHAR(1), SQLCODE);
   OPEN C2;

PROCEDURE C2_FETCH (
      :NAME     CHAR(30),
      :DOB      NUMERIC(8),
      :ZIP      CHAR(10),
      :NOZIP    NUMERIC,
      SQLCODE);
   FETCH C2 INTO :NAME, :DOB, :ZIP:NOZIP;

PROCEDURE C2_CLOSE (SQLCODE);
   CLOSE C2;
```

An "indicator" – indicates if NULL

Paradigm

MCP-4055 38

Compiling Modules

- Run ***SYSTEM/MCPSQL/MODLANG("...")**
  - Parses module source code
  - Initiates DMALGOL compile to generate the library
- String parameter
  - List of data base titles used by the module
  - Multiple titles must be separated by "**;**"
  - Must pass an empty string if **DATABASE** is used in the module header instead
- File assignments
  - **INFILE** module source (**SEQDATA**)
  - **ERRORFILE** defaults to a remote file
  - **CODE** title of resulting library codefile

Paradigm

MCP-4055 39

Sample ModLang Calls

```
01  W-SQLCODE     PIC S9(9) COMP.
01  W-STATUS      PIC X(1).
01  W-NAME        PIC X(30).
01  W-DOB         PIC S9(8) SIGN LEADING SEPARATE.
01  W-ZIP         PIC X(10).
01  W-ZIP-NULL    PIC S9(11) SIGN LEADING SEPARATE.

CHANGE ATTRIBUTE TITLE OF "MODLIB" TO
    "(DEMO)OBJECT/MODULE/TEST ON TEMP.".
MOVE "A" TO W-STATUS.
CALL "C2_OPEN IN MODLIB" USING
    W-STATUS, W-SQLCODE.
CALL "C2_FETCH IN MODLIB" USING
    W-NAME, W-DOB, W-ZIP, W-ZIP-NULL,
    W-SQLCODE.
CALL "C2_CLOSE IN MODLIB" USING W-SQLCODE.
```

Paradigm

MCP-4055 40

Using the Call Level Interface (CLI)

The MCPSQL Call Level Interface

- Very flexible, dynamic, and general purpose interface to MCPSQL
  - Everything can be configured at run time
    - No pre-compiling or static definitions
    - Works with any MCPSQL-capable data base
  - Catalog (schema discovery) capabilities
  - Suitable for building tools and ad hoc interfaces
  - Should work with any library-capable language
  - Low-level API – a bit of a challenge to use
- Based on ANSI SQL-99 CLI specification
  - Significant differences in parameter types
  - Uses offsets within buffers rather than C-like pointers

Paradigm

MCP-4055 42

Example CLI Programs

- CLI coding is far too complex and voluminous to attempt to show here
- Check out some examples
  - **`*EXAMPLE/MCPSQL/CLI/=`** on the release media
  - Printed examples in the MCPSQL *Programming Guide*
  - Examples at http://www.digm.com/UNITE/2017

New ODBC Driver

MCPSQL ODBC Driver

- MCP-only ODBC access to DMSII
- Multiple versions
  - Windows x86 and x64
  - Linux x86 and x64 (RPM package)
- On Windows
  - Can install both x86 and x64 on same system
  - Make sure the client tool matches the ODBC type
- Install from INSTALLS share
- Set up DSNs using 32- or 64-bit ODBC Administrator (Windows)

Current Driver Issues

- Schema queries do not seem to work
  - Cannot retrieve list of tables, columns, etc.
  - Makes it difficult to work with SSMS, etc.
  - Works with MSQRY32, though
- Fragile
  - Missing/extra items in connection string => crashes
  - 32-bit Administrator config does not restore values
  - UID & PWD seem to be required on connection string
- Once you get it to connect, seems to work quite well

Using ODBC Driver with SQL Server

- Create a 32- or 64-bit DSN for MCPSQL
- Define a new Linked Server
  - Give it a name
  - Select provider "Microsoft OLE DB Provider for ODBC Drivers"
  - Set Location to MCP host name or IP address
  - Leave Catalog blank
  - Under Security, set up a remote login for the MCP usercode and password
  - Leave Serve Options at defaults
- Use OPENQUERY in SQL Server:
  - `select * from OPENQUERY(XXDB, '<MCPSQL query>')`

Paradigm

MCP-4055 47

A Bit About Performance

MCPSQL Compared to DMSQL

- Unscientific test environment
  - MCP 18 Developer Studio, Personal Edition
  - Dell Precision 7520, 16GB, 4 x i5 7300HQ 2.50GHz
  - Data base and programs from 2008 DMSQL study
- And the results are…
  - ***Totally inconclusive***
  - Highly inconsistent run times, even for DMSII Host Language programs
  - Can't say that MCPSQL CLI is faster than DMSQL
    - Sometimes see up to 2x improvement
    - Sometimes 15-20% slower
    - Not conclusive, given the Host Language run-time inconsistencies

One Disappointing Benchmark

- Common problem:
  - Find the record having the biggest/smallest/-earliest/latest value
  - Note – want the *record having the value*, not the value

```
select PRCHPRODUCT, PRCHPRCLST, PRCHUNITPRC, PRCHEFFDT
from OEPRCH p1
where PRCHSUBSYS=1 PRCHPRODUCT='10116' and
      0=(select count(*) from OEPRCH p2
         where p2.PRCHSUBSYS=p1.PRCHSUBSYS and
               p2.PRCHPRODUCT=p1.PRCHPRODUCT and
               p2.PRCHPRCLST=p1.PRCHPRCLST and
               p2.PRCHEFFDT > p1.PRCHEFFDT)
order by PRCHPRODUCT, PRCHPRCLST
```

- It's still really, really slow

Why Use MCPSQL?

- It sure beats using ERGO & DMINQ
- SQL is the standard for data retrieval
  - Higher-level abstractions always have a cost
  - Higher-level abstractions always win in the end
- Many queries are easier to write in SQL
  - Range retrievals, **`LIKE`** patterns, set memberships
  - Summations and multi-level control breaks
- Dynamic retrieval specification is nearly impossible with Host Language interface
  - CLI offers *run-time query construction*
  - Dynamic specification of selection, grouping, sorting

Why Use MCPSQL? (continued)

- Better isolation from DMSII schema changes (reorgs, etc.)
- Compared to OLE DB & ODBC Access
  - Performance is sometimes better, sometimes not
  - No need for SQL Server and a Windows front end
  - Can be used from within MCP applications
- It's there, and it's free
  - It won't get any better unless we need it to
  - Try it – report any problems you find

References

- *Relational Database Server for ClearPath MCP Query Processor Installation and Operations Guide* (8222 3819-002)
- *Relational Database Server for ClearPath MCP Query Processor Programming Guide* (8222 3827-002)
- Prior DMSQL presentations
  `http://www.digm.com/UNITE/2006#MCP3023`
  `http://www.digm.com/UNITE/2008`
- This presentation and examples
  `http://www.digm.com/UNITE/2017`

Paradigm

MCP-4055 53

END

Using MCPSQL

2017 UNITE Conference

Session MCP-4055