DB2 for i 7.1 – More Than Meets the Eye

Kent Milligan, kmill@us.ibm.com
IBM STG Lab Services
ISV Enablement – IBM i

DB2 for i Focus Areas

- **The Self Managing Database**
  - Reduced TCO thru automation
  - Integration: Built-in Security and Auditing

- **Trusted Reliability & Scalability**
  - Simplified, best of breed scaling
  - Integrated transaction management
  - Advanced, flexible logging facilities

- **Open for Business**
  - SQL, the strategic interface
  - Latest de facto standards

- **Innovative Applications**
  - SQL & Data-centric programming
  - Move to SOA over time

- **Business Intelligence**
  - Store, manage, and ANALYZE data!
  - End user query and reporting to large scale data warehousing
**DB2 for i 7.1 Enhancements**

**Rapid Application Development**
- SQL & RPG Integration
- Stored procedure Result Set consumption
- FIELDPROC for transparent column-level encryption
- XML Integration
  - XML data type
  - Annotated XML Decomposition
  - SQL XML Publishing functions
- Three-part Aliases
- Compatibility with DB2 Family & Oracle
  - MERGE statement
  - Array support & Global Variables
  - REPLACE option on CREATEs
  - Currently Committed supported
- JDBC & .NET enhancements

**Performance & Self-Tuning Enhancements**
- SQL Query Engine (SQE) enhancements
  - Adaptive Query Processing
  - Self-Learning Optimization
  - Inline UDF query rewrite
  - Logical File on FROM support
- Indexing Advancements
  - SQL Select/Omit Indexes
  - EVI Aggregates
- CPYFRMIMPF performance
- SSD & In-Memory Database Enablement
- OmniFind Text Search Server enhancements

**Trusted Reliability**
- Enhanced Remote Journal filtering
- Library-level Journaling filtering
- IASP spanning transactions

**Simplified Management**
- IBM i Navigator Enhancements
  - Progress Monitors – Alter Table, Index Build
  - Index Advisor improvements
  - Enhanced Generate SQL capability
  - Object Folder content saves

**Performance & Self-Tuning Enhancements**
- SQL Query Engine (SQE) enhancements
  - Adaptive Query Processing
  - Self-Learning Optimization
  - Inline UDF query rewrite
  - Logical File on FROM support
- Indexing Advancements
  - SQL Select/Omit Indexes
  - EVI Aggregates
- CPYFRMIMPF performance
- SSD & In-Memory Database Enablement
- OmniFind Text Search Server enhancements

**Simplified Management**
- IBM i Navigator Enhancements
  - Progress Monitors – Alter Table, Index Build
  - Index Advisor improvements
  - Enhanced Generate SQL capability
  - Object Folder content saves

**Data Intelligence & Interoperability**
- DB2 Web Query for System i
  - Excel client support
  - Microsoft SQL Server adapter

**New Wiki for DB2 Enhancements via PTF**

- Regularly check (or Subscribe) to the DB2 for i Updates Wiki!
  - Contains details on new PTFs that deliver new DB2 capabilities
  - Examples:
    - CANCEL_SQL system stored procedure
    - PROGRAM NAME keyword for controlling SQL Triggers Program Name
    - SQL Query Engine 6.1 support for Logical File on FROM clause
  - Wiki URL:
    https://www.ibm.com/developerworks/systems/ibmi/techupdates/db2

- The wiki is part of a new IBM i zone in IBM developerWorks
  https://www.ibm.com/developerworks/systems/ibmi/
Application Development Enhancements

Enhancements for IBM i Application Development

- ILE COBOL SQL Pre-compiler Enhancements
  - Concurrent Access Resolution parameter
  - Support for new COMP-5 type

- Improved SQL & ILE RPG Pre-compiler Integration
  - Concurrent Access Resolution parameter
  - Debug Encryption Key parameter
  - ALIAS keyword support

```sql
CREATE TABLE customers(
  customer_number INTEGER,
  customer_name CHAR(30),
  customer_address VARCHAR(80))
```

```sql
d DS1 E DS EXTNAME(customers) QUALIFIED ALIAS
```

* The subfields of data structure would be:
  * CUSTOMER_NUMBER
  * CUSTOMER_NAME
  * CUSTOMER_ADDRESS
Industry Standard Application Interface Improvements

- **ADO.NET**
  - ‘Concurrent Access Resolution’ property
  - Visual Studio 2008 support
  - Online help integration with Visual Studio
  - Support for Multi-row Delete, Merge, and Update statements

- **ODBC**
  - ConcurrentAccessResolution connection keyword
  - Support for Multi-row Delete, Merge, and Update statements

- **OLE DB**
  - ‘Concurrent Access Resolution’ connection property

- **SQL CLI**
  - TINYINT data type support
  - SQL_ATTR_CONCURRENT_ACCESS_RESOLUTION connection attribute
  - Support for Multi-row Delete, Merge, and Update statements
  - QIBM_SRV/RMODE_SBS environment variable for QSQSRVR jobs (PTFs for 6.1, 5.4)

- **JDBC**
  - Support for SQL routine ARRAY parameters
  - "concurrent access resolution" connection property
  - Native JDBC driver enhancements
    - "servermode subsystem" property to control subsystem used for QSQSRVR jobs
    - Metadata compatibility with Toolbox JDBC and other industry drivers

FIELDPROC - Seamless Column-Level Encoding and Decoding

**New Order**

Encrypt

1111 2222 3333 4444

Decrypt

T3vS#45zlIJ9*m$p6

Authorized Access

1111 2222 3333 4444

Encrypt

DB2
FIELDPROC Implementation Details

- Developers have complete freedom to create virtually any column encoding/decoding
  - Encryption (3rd party solutions: Linoma Software, Patrick Townsend, nuBridges)
  - Data compression
  - Text normalization …

- DB2 automatically calls registered FIELDPROC program for ALL interfaces (applications, SQL, native record level-access, CL: DSPPFM, CPYF…)
  - Program must be an ILE program object and contain no SQL
  - Fieldproc program called for 3 different events:
    - Column creation/registration to define attributes of the stored encoded value
    - Write operations to encode data
    - Read operations to decode data

- FIELDPROC registration requires usage of SQL
  - Be extremely careful of using SQL ALTER statement on Physical Files
    
    ALTER TABLE orders ALTER COLUMN creditcardnum
    SET FIELDPROC mylib/ccpgm

Global Variables

- Enables simpler sharing of values between SQL statements and SQL objects (Triggers, Views, etc) across the life of a job/database connection
  - Variables created within a job on first reference
  - Variable values for job can be displayed with RETURN_VARIABLE_VALUES procedure in QSYS2

- Example #1 – Cache User Information

  CREATE VARIABLE gvdept INTEGER DEFAULT
  (SELECT deptno FROM employee WHERE empuserID = USER);

  CREATE VIEW filtered_employee AS
    SELECT firstname, lastname, phoneno FROM employee WHERE deptno = gvdept);

  ...

  SELECT firstname, phoneno FROM filtered_employee;
Global Variables

- Example #2 – Conditional Trigger Behavior

```sql
CREATE VARIABLE batch_run CHAR(1);

CREATE TRIGGER track_expenses AFTER INSERT ON expenses
REFERENCING NEW AS n FOR EACH ROW
WHEN (batch_run='N')
BEGIN
    DECLARE emplname CHAR(30);
    SET emplname = (SELECT lastname FROM employee WHERE empid=n.empno);
    IF n.totalamount < 10000 THEN
        INSERT INTO travel_audit
        VALUES(n.empno, emplname, n.deptno, n.totalamount, n.enddate);
    ELSE
        SIGNAL SQLSTATE '38001' SET MESSAGE_TEXT='Exceeded Maximum';
    END IF;
END;
```

VALUES 'Y' INTO batch_run;

---

Result Set Integration – Embedded SQL & SQL Routines

- Programmers can now directly integrate stored procedure result sets with embedded SQL & SQL Routines
  - Key Enabler Statements: ASSOCIATE LOCATOR & ALLOCATE CURSOR
  - Optionally, DESCRIBE PROCEDURE & DESCRIBE CURSOR statements can be used to dynamically determine the number and contents of a result set

```sql
DECLARE sprs1 RESULT_SET_LOCATOR VARYING;
CALL GetProj(projdept);
ASSOCIATE LOCATORS(sprs1) WITH PROCEDURE GetProj;
ALLOCATE mycur CURSOR FOR RESULT SET sprs1;
SET totstaff=0;
myloop: LOOP
    FETCH mycur INTO prname, prstaff;
    IF row_not_found=1 THEN
        LEAVE fetch_loop;
    END IF;
    SET totstaff= totstaff + prstaff;
    IF prstaff > moststaff THEN
        SET bigproj = prname;
        SET moststaff= prstaff;
    END IF;
END LOOP;
CLOSE mycur;
```
Stored Procedure Enhancements

- Expressions on CALL statement
  
  \[
  \text{CALL myprocedure ( 1, UPPER(company\_name), company\_discountrate*100 )}
  \]

- ARRAY support for SQL Routines
  - Enables exchange of data collections
  - ARRAY element limited to simple data types
  - ARRAY type can be used as parameter for SQL Routine or a local variable
  - Interfaces supporting SQL Routine ARRAY parameters:
    - JDBC
    - SQL Routines
  - Examples:
    
    \[
    \text{CREATE TYPE partids AS CHAR(3) ARRAY[10];}
    \]
    
    \[
    \text{CREATE TYPE intarray AS INTEGER ARRAY[5];}
    \]

---

Stored Procedure Enhancements – ARRAY Example

- Return part type and quantity for the specified collection of parts
  
  \[
  \text{CREATE OR REPLACE PROCEDURE list\_parts}
  \]
  
  \[
  \text{(IN inparts partids, OUT part\_qty intarray)}
  \]
  
  \[
  \text{DYNAMIC RESULT SETS 1}
  \]
  
  \[
  \text{LANGUAGE SQL}
  \]
  
  \[
  \text{BEGIN}
  \]
  
  \[
  \text{DECLARE cur1 CURSOR FOR SELECT t.id, part\_qty, part\_type}
  \]
  
  \[
  \text{FROM parts, UNNEST(inparts) AS t(id) WHERE t.id = part\_id;}
  \]
  
  \[
  \text{IF CARDINALITY( inparts )}>5 THEN}
  \]
  
  \[
  \text{SIGNAL SQLSTATE \textquote{38003}}
  \]
  
  \[
  \text{SET MESSAGE\_TEXT='Too many parts';}
  \]
  
  \[
  \text{END IF;}
  \]
  
  \[
  \text{SET part\_qty = (SELECT ARRAY\_AGG(part\_qty)}
  \]
  
  \[
  \text{FROM parts,UNNEST(inparts) AS t2(id) WHERE t2.id = part\_id;}
  \]
  
  \[
  \text{OPEN cur1;}
  \]
  
  \[
  \text{END;}
  \]
  
  \[
  \text{...}
  \]
  
  \[
  \text{SET myparts = ARRAY[\textquote{W12},\textquote{S55},\textquote{M22}];}
  \]
  
  \[
  \text{CALL list\_parts(myparts, out\_qty);}\]
  
  \[
  \text{...}
  \]

---

<table>
<thead>
<tr>
<th>ID</th>
<th>PART_QTY</th>
<th>PART_TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>W12</td>
<td>25</td>
<td>KSR</td>
</tr>
<tr>
<td>S55</td>
<td>124</td>
<td>KSR</td>
</tr>
<tr>
<td>M22</td>
<td>125</td>
<td>MNG</td>
</tr>
</tbody>
</table>

- OUTPUT

out\_qty Array:

\[
[1] = 25
\]

\[
[2] = 124
\]

\[
[3] = 125
\]
Simplified Remote Data Access

- **Three-part Aliases**
  - Simplifies access to DB2 objects on different partitions or servers (implicit DRDA connection)
  - **Examples**:
    - `CREATE ALIAS mylib.tab1 FOR rdb1a.mylib.tab1`
    - `SELECT c1, c2 FROM mylib.tab1`
    - `SELECT c1, c2 FROM rdb1a.mylib.tab1`
  - **Considerations**
    - A single SQL statement can only reference objects from a single database server
    - Alias name must be the same as the object on the remote server. Local Alias can reference an Alias on remote server

- **Automatic SQL DRDA Package Creation**
  - Removes burden from developer having to create the required SQL packages on a remote server

---

Enhanced Parameter Marker Support

**BEFORE:**

```sql
SET stmt1 = 'SELECT c2,c2 FROM tab1 WHERE c1 > ' || 
CAST(? AS DECIMAL(8,2)) + CAST(? AS DECIMAL(8,2)) || ';
PREPARE pstmt1 FROM stmt1;
```

**AFTER:**

```sql
SET stmt1 = 'SELECT c2,c2 FROM tab1 WHERE c1 > ? + ?';
PREPARE pstmt1 FROM stmt1;
```
XML Integration

XML Integration with DB2

- Rich XML Support within DB2 for i – integrated solution that replaces DB2 XML Extender product
  - New XML data type to simplify storage and retrieval of XML documents
    - XML data access protected with rock-solid DB2 security
    - XML covered by Database Backup and Recovery processes
  - Annotated decomposition of XML documents into DB2 columns
  - Generate XML document with SQL-XML publishing functions

- IBM OmniFind Text Search Server provides advanced, high-speed search capabilities for stored XML documents
  - Scope searches to specific elements of an XML document: `/book/title[contains("winning") ]`
  - XQuery interface not yet supported
XML Data Type

- New XML data type
  - Support XML values up to 2 GB
  - Type can be used for column, parameter, and host variable values

- XML Schema-based validation supported

```
INSERT INTO reservations (resdoc)
VALUES (XMLVALIDATE(
  XMLPARSE (DOCUMENT
    GET XML_FILE ('/dir1/r6.xml'))
  ACCORDING TO XMLSCHEMA
  ID mylib.resschema)
)
```

- XML File Reference variables enable simple export XML documents to IFS

```
CREATE TABLE reservations
  (  resID  INTEGER
      GENERATED ALWAYS
      AS IDENTITY,
  resDoc  XML )
```

DB2

```
<booking unitCharge="50" units="2" currency="USD" status="confirmed">
  <item>
    <room hotelName="White Palace" type="suite" bookedFrom="2011-05-25" bookedTo="2011-05-29" />
  </item>
<br />
</booking>
```

Integrated XML Utilities

**Built-in Functions:**

- `GET_XML_FILE` Returns the contents of an IFS file or source physical file member as a LOB Locator value (requires Commitment Control)
- `XMLVALIDATE` Validates XML value against an XML schema
- `XMLPARSE` Parses Character/LOB data to produce XML value
- `XMLSERIALIZE` Converts XML value into Character/LOB data
- `XMLPARSE` Parses Character/LOB data to produce XML value
- `XMLSERIALIZE` Converts XML value into Character/LOB data
- `XSLTRANSFORM` Converts XML data into other XML, HTML, and plain text formats using the XSLT processor (requires 5733-XT2)

**System Stored Procedures (SYSPROC library):**

- `XSR_REGISTER` Add an XML Schema document into the DB2 XML Schema Repository (XSR) for Validation / Decomposition
- `XSR_ADDSCHEMADOC` Merge an XML Schema within an existing XML Schema
- `XSR_COMPLETE` Complete the registration of XML Schema(s) within DB2 XSR
- `XSR_REMOVE` Remove a registered XML Schema document
- `XDBDECOMPXML` Decompose an XML document into specified DB2 objects using annotated decomposition
Annotated XML Document Decomposition

1) Map the DB2 and XML document relationships

XML Doc

2) Define mapping in XSD file

XSD

3) Register and stored XSD mapping within DB2 XML Schema Repository (XSR)

4) Decompose/Shred the XML document

Decomposition Example

XSD
Decomposition Example

- **XML Decomposition Steps:**
  1. Create XSD file with DB2 to XML mapping
  2. Store and register XSD file within DB2 Schema Repository (XSR)

```
CALL SYSPROC.XSR_REGISTER ('MYLIB', 'BOOKSCHEM', null,
GET_XML_FILE('/dir/authbooks.xsd'), null)
```

```
CALL SYSPROC.XSR_COMPLETE('MYLIB','BOOKSCHEM',null,1)
```

3. Decompose XML Document

```
CALL SYSPROC.XDBDECOMPXML
  ('MYLIB','BOOKSCHEM', GET_XML_FILE('/mydir/ship1.xml'),null)
```

**Decomposition Generated Statements:**

```sql
INSERT INTO authors
VALUES(22, 'Tony Dungy')
```

```sql
INSERT INTO books
VALUES(22, 'Quiet Strength', 1414318014),
(22, 'Uncommon', 1414326815)
```

---

**SQL XML Publishing Functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLATTRIBUTES</td>
<td>Returns XML sequence that contains an attribute node for each non-null argument</td>
</tr>
<tr>
<td>XMLCOMMENT</td>
<td>Returns XML value with a single comment node from a string</td>
</tr>
<tr>
<td>XMLCONCAT</td>
<td>Returns XML value that represents a forest of XML elements generated by concatenating a variable number of arguments</td>
</tr>
<tr>
<td>XMLDOCUMENT</td>
<td>Returns XML value with a single document node and zero or more nodes as its children</td>
</tr>
<tr>
<td>XMLELEMENT</td>
<td>Returns XML value that represents an XML element</td>
</tr>
<tr>
<td>XMLFOREST</td>
<td>Returns XML value that represents a forest (sequence) of XML elements that all share a specific pattern</td>
</tr>
<tr>
<td>XMLPI</td>
<td>Returns XML value with a single processing instruction node</td>
</tr>
<tr>
<td>XMLNAMESPACES</td>
<td>Returns the declaration of one or more XML namespaces</td>
</tr>
<tr>
<td>XMLROW</td>
<td>Returns XML value with a single document node containing one top-level element node</td>
</tr>
<tr>
<td>XMLTEXT</td>
<td>Returns XML value with single text node that contains value of argument</td>
</tr>
<tr>
<td>XMLEAGG</td>
<td>Returns an XML sequence that contains an item for each non-value in set of XML values</td>
</tr>
<tr>
<td>XMLGROUP</td>
<td>Returns XML value with a single document node containing one top-level element node from a group of rows</td>
</tr>
</tbody>
</table>
SQL XML Publishing Example – XMLELEMENT & XMLATTRIBUTE

- Generate XML values for employees celebrating 25th anniversary:

```sql
SELECT XMLSERIALIZE(
    XMLELEMENT(NAME "employee", XMLATTRIBUTES(e.empno as "id"),
    XMLELEMENT(NAME "Name", e.firstname || ' ' || e.lastname),
    XMLELEMENT(NAME "Extension", e.phoneno),
    XMLELEMENT (NAME "DeptNo", d.deptno))  AS CLOB(100) ) as "XMLResult"
FROM employee e, department d
WHERE e.workdept = d.deptno AND
YEAR(CURRENT DATE) –
    YEAR(hiredate) = 25
```

Output for XMLResult:
- `<employee id="000010">`
  - `<Name>JENNA HAAS</Name>`
  - `<Extension>0420</Extension>`
  - `<DeptNo>A00</DeptNo>`
- `<employee id="000050">`
  - `<Name>JOSH GEYER</Name>`
  - `<Extension>1103</Extension>`
  - `<DeptNo>E01</DeptNo>`

---

SQL XML Publishing Example - XMLFOREST

- Generate XML values for employees celebrating 25th anniversary using XMLFOREST to simplify query:

```sql
SELECT XMLSERIALIZE(
    XMLELEMENT(NAME "employee", XMLATTRIBUTES(e.empno as "id"),
    XMLFOREST(e.firstname || ' ' || e.lastname as "Name",
    e.phoneno as "Extension",
    d.deptno as "DeptNo")
)  AS CLOB(100) ) as "XMLResult"
FROM employee e, department d
WHERE e.workdept = d.deptno AND
YEAR(CURRENT DATE) –
    YEAR(hiredate) = 25
```

Output for XMLResult:
- `<employee id="000010">`
  - `<Name>JENNA HAAS</Name>`
  - `<Extension>0420</Extension>`
  - `<DeptNo>A00</DeptNo>`
- `<employee id="000050">`
  - `<Name>JOSH GEYER</Name>`
  - `<Extension>1103</Extension>`
  - `<DeptNo>E01</DeptNo>`
```
SQL XML Publishing Example - XMLAGG

- Generate count and XML value for parts with specified type:

```sql
SELECT COUNT(*) AS PartCnt,
XMLSERIALIZE(
  XMLELEMENT(NAME "Parts", XMLATTRIBUTES(parttype AS "type"),
    XMLAGG(
      XMLELEMENT(NAME "pid", partid)  ORDER BY partid
    ) AS CLOB(130)) AS PartList
FROM parts WHERE parttype IN ('C01', 'E21')
GROUP BY parttype
```

<table>
<thead>
<tr>
<th>PartCnt</th>
<th>PartList</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td><code>&lt;Parts type=&quot;C01&quot;&gt;&lt;pid&gt;000130&lt;/pid&gt;&lt;pid&gt;200140&lt;/pid&gt;&lt;/Parts&gt;</code></td>
</tr>
<tr>
<td>3</td>
<td><code>&lt;Parts type=&quot;E21&quot;&gt;&lt;pid&gt;000320&lt;/pid&gt;&lt;pid&gt;100330&lt;/pid&gt;&lt;pid&gt;200340&lt;/pid&gt;&lt;/Parts&gt;</code></td>
</tr>
</tbody>
</table>

SQL XML Publishing Example - XMLGROUP

- Generate count and XML value for parts with specified type:

```sql
SELECT COUNT(*) AS PartCnt,
XMLGROUP( parttype AS "type", partid AS "pid"
  ORDER BY parttype, partid
  OPTION ROW "Parts" ROOT "PartList") AS partlist
FROM parts
WHERE parttype IN ('C01', 'E21')
GROUP BY parttype
```

<table>
<thead>
<tr>
<th>PartCnt</th>
<th>PartList</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td><code>&lt;PartList&gt;&lt;Parts&gt;&lt;type&gt;C01&lt;/type&gt;&lt;pid&gt;000130&lt;/pid&gt;&lt;pid&gt;200140&lt;/pid&gt;&lt;/Parts&gt;&lt;/PartList&gt;</code></td>
</tr>
<tr>
<td>3</td>
<td><code>&lt;PartList&gt;&lt;Parts&gt;&lt;type&gt;E21&lt;/type&gt;&lt;pid&gt;000320&lt;/pid&gt;&lt;pid&gt;100330&lt;/pid&gt;&lt;pid&gt;200340&lt;/pid&gt;&lt;/Parts&gt;&lt;/PartList&gt;</code></td>
</tr>
</tbody>
</table>
SQL Enhancements

MERGE Statement

- Allows application to use a single SQL statement to Update, Delete, or Insert into a table based on values from a source table/query

- Simplifies applications trying to merge detailed transaction data into a summary file
  - Typical processing…
    - Perform INSERT when transaction type does NOT yet exist in summary file
    - Perform UPDATE when transaction type does exist in summary file to add to the total for that type
MERGE Statement – Detailed Examples

- **Example #1**: Merge rows into the Account table, Updating the balance from the set of transactions against an account ID and Inserting new accounts from the consolidated transactions that do not already exist.
  
  ```sql
  MERGE INTO account_summary AS a
  USING (SELECT id, SUM(trans_amount) sum_amount FROM trans GROUP BY id) AS t
  ON a.id = t.id
  WHEN MATCHED THEN UPDATE SET balance = a.balance + t.sum_amount
  WHEN NOT MATCHED THEN
  INSERT (id, balance) VALUES (t.id, t.sum_amount)
  ```

- **Example #2**: Update the activities from Atlanta group in the archive table. Delete all outdated activities and update the archived activities information for any changed activities. Insert new upcoming activities into the archive.
  
  ```sql
  MERGE INTO archive ar
  USING (SELECT actID, actDesc, actDate, actLastChg FROM actAtlGrp) ac
  ON (ar.activityID = ac.actID) AND ar.activityGroup = 'A'
  WHEN MATCHED and ac.actDate < CURRENT DATE THEN DELETE
  WHEN MATCHED and ar.LastChg < ac.actLastChg THEN
  UPDATE SET(activityDesc,activityDate,LastChg)=(ac.actDesc,ac.actDate,DEFAULT)
  WHEN NOT MATCHED AND ac.actDate >= CURRENT DATE THEN
  INSERT (activityGroup, activityID, activityDesc, activityDate)
  VALUES ('A', ac.actID, ac.actDesc, ac.actDate)
  ELSE IGNORE
  ```
**DB2 Concurrent Access Resolution**

- Concurrent Access Resolution behavior controllable at different levels
  - System-wide: SQL_CONCURRENT_ACCESS_RESOLUTION QAOQINI option
  - Program-level: CONACC pre-compiler option
  - Connection-level property/attribute
    - IBM i Access middleware: ADO.NET, JDBC, ODBC, OLE DB
    - SQL CLI & Native JDBC Driver
  - Statement-level
    - USE CURRENTLY COMMITTED
    - WAIT FOR OUTCOME
    - SKIP LOCKED DATA (added in 6.1)

**JOB#1:**

```sql
UPDATE parts
SET part_qty = 25
WHERE part_id='W12'
```

**JOB#2:**

```sql
SELECT part_id
FROM parts
WHERE part_type
  IN ('KSR', 'MNG')
USE CURRENTLY COMMITTED
```

**PARTS**

<table>
<thead>
<tr>
<th>PART_QTY</th>
<th>PART_ID</th>
<th>PART_TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>W12</td>
<td>KSR</td>
</tr>
<tr>
<td>220</td>
<td>T53</td>
<td>MNG</td>
</tr>
<tr>
<td>124</td>
<td>S55</td>
<td>KSR</td>
</tr>
<tr>
<td>240</td>
<td>M11</td>
<td>MNG</td>
</tr>
<tr>
<td>125</td>
<td>M22</td>
<td>MNG</td>
</tr>
<tr>
<td>649</td>
<td>K01</td>
<td>TJ</td>
</tr>
</tbody>
</table>

**Built-In SQL Functions Toolbox Additions**

- MQ Integration Functions
  - Scalar functions:
    MQSEND, MQREAD, MQREADCLOB, MQRECEIVE, MQRECEIVECLOB
  - Table functions
    MQREADALL, MQREADALLCLOB, MQRECEIVEALL, MQRECEIVEALLCLOB

- BLOB & CLOB Integration
  - GET_BLOB_FROM_FILE
  - GET_CLOB_FROM_FILE, GET_DBCLOB_FROM_FILE

- Bit Manipulation functions
  - BITAND, BITOR, BITXOR
  - BITNOT, BITANDNOT
Miscellaneous SQL Enhancements

- **REPLACE Option for CREATE statements**
  - Eliminates need for the Drop statement
  - Preserves existing object dependencies & privileges!
  - Supported objects: Alias, Function, Procedure, Sequence, Trigger, Variable, View
    ```sql
    CREATE OR REPLACE ALIAS myAlias FOR schema.tab1
    ```

- **ALTER TABLE Enhancements**
  - ADD BEFORE column
  - Identity Column support for existing columns
  - Preservation of statistics
  - Improved performance for partitioned tables

- **Partitioned Table Enhancements**
  - RI Constraint support
  - Identity Column support

- **SQL Object Deflation – Table, View, Index**

- **128-byte Schema Names**

---

Miscellaneous SQL Management Utilities

- **All SQL** interface for cancelling long-running SQL statements
  - QSYS2.SQL_CANCEL procedure (6.1 PTF)
  - Example: CALL QSYS2.SQL_CANCEL(’197968/QUSER/QZDASOINIT’)

- Retrieve associated QSQSRVR jobs & metrics for a specified job/application
  - QSYS2.FIND_QSQSRVR_JOBS procedure (6.1 & 5.4 PTFs)
  - Example: CALL QSYS2.FIND_QSRVR_JOBS(’566463/USERNAME/QP0ZSPWP’)
IBM OmniFind Text Search Server Enhancements

- Product (5733-OMF) originally released after GA of IBM i 6.1
  - Common DB2 Family text search support
  - Supports text columns and text documents (PDF, DOC, PPT, ...)
  - High-speed, advanced linguistic searches
    
    \[
    \text{CONTAINS( feedDoc, 'California insurance settlement')} = 1 \\
    \text{CONTAINS( textFld, 'Man wins lottery')} = 1
    \]

- OmniFind 7.1 Enhancements
  - Enhanced XML search support
    - Date and Date/Time comparisons:
      - \[ /\text{Book} \left[ \text{pubDate} > \text{xsd:dateTime}('2005-04-15') \right] \]
    - Numeric comparisons:
      - \[ /\text{Book} \left[ \text{Cost} \leq 59.95 \right] \]
    - Namespace specific searches
  - Enhanced Save / Restore capabilities
  - Graphical text index management

Performance Enhancements
DB2 Performance Enhancements

- SQL Query Engine (SQE) Enhancements
  - Support for Logical File on FROM clause
  - Performance advancements
    - Background Self-Learning Query Optimization
    - Adaptive Query Processing
    - Global Statistics Cache
    - Inline User-Defined Function rewrite
- SQE Indexing Advancements
  - Optimizer awareness of SQL Select/Omit Indexes & Select/Omit Logical Files
  - Encoded Vector Index Aggregate support
- Improved locking performance for LARGE transactions
- Improved CPYFRMIMPF performance (6.1 & 5.4 PTFs)
- DB2 Object-level performance
  - SSD Media Preference and Random/Sequential Usage Statistics
  - OVRDBF … REUSEDLT(*NO) for faster Blocked Inserts/Writes
  - In-Memory Database Enablement
    - CHGPF … KEEPINMEM(*YES)
    - CHGLF … KEEPINMEM(*YES)

SQE Adaptive Query Processing

- Real-time self-learning query optimization
  - Enables query plan to be changed while query is running
  - Plan adjustments & query restart completely transparent to the application
- Intelligent monitor agents automatically assigned to each query by SQE
  - Monitoring starts after 2 seconds
  - Periodically polling measures progress against estimates and other plan assumptions
- Real-time plan adjustments can include
  - Change in join order
  - Utilization of a new index
  - …
  - No user interaction required!
SQE Indexing Advancements

- Query Optimizer awareness of SQL Select/Omit Indexes for query plans

  CREATE INDEX cust_ix1 ON customers(cust_id) WHERE activeCust='Y'

- Encoded Vector Index (EVI) Aggregate Support

  CREATE ENCODED VECTOR INDEX idx1 ON sales(region)
  INCLUDE (SUM(saleamt), COUNT(*))

  CREATE ENCODED VECTOR INDEX idx2
  ON sales(territory)
  INCLUDE (SUM(saleamt + promoamt))

  SELECT territory, SUM(saleamt+promoamt) FROM sales
  GROUP by territory

  SELECT region, SUM(saleamt) FROM sales GROUP BY region

  EVIs are maintained as the underlying table is modified

DB2 SSD (Solid State Disks) Enablement

- SSD can improve performance for some DB2 objects
  - Large amount of random data access and...
  - Data that is read many times, but written less frequently
- DB2 interfaces enhanced to allow a user to indicate an SSD media preference on table, index, physical file, and logical file
  - SQL: UNIT SSD clause for object and partition
    - CREATE/ALTER TABLE
    - CREATE INDEX
  - CL: UNIT("SSD") parameter
    - CRTPF, CRTLF, and CRTSRCPF
    - CHGPF, CHGLF, and CHGSRCPF
- ALTER and CHGPF/LF interfaces support asynchronous movement of data and indexes
- Key DB2 7.1 Addition - New random and sequential statistics for tables and indexes

  5.4 - Database Group SF99504 #23
  6.1 - Database Group SF99601 #10

  Moving DB2 tables to SSD reduced month end batch run time by 40%!

  Associated Bank

  *http://www.ibmsystemsmbpower.com/nxtbooks/ibmsystemsmag/ibmsystems_power_200909/index.php#16

  4:22
  2:43
  2:48
OVRDBF REUSEDLT(*NO)

- Temporarily override Reuse Deleted Rows feature of a table to speed up Insert/Write performance
  - Enables DB2 to utilize row-level blocking
  - Enables DB2 SMP feature to perform parallel index maintenance
  - Preserves table’s ability to benefit from Enable Concurrent Write (ie, Holey Inserts)
  - Support for 5.4 and 6.1 releases can be purchased from IBM Lab Services
    (http://ibm.com/systems/services/labservices/contact.html)

Maintain indexes in parallel...then Insert Blocked data into table at the end

Availability & Recovery Enhancements
Database Availability and Recovery

- **DB2 Engine Improvements**
  - Independent ASPs Enhancements
    - Support for transactions spanning System & Independent ASP
    - Support for CICS transactions
  - Constraint Enforcement fast-path for HA Switchover
    - CHGPFCST CHECK(*NO) (5.4 & 6.1 PTFs)

- **Journal Enhancements**
  - Localized-journaling for indexes with large logical page sizes
  - Additional controls for remote journal filtering
  - Enhanced generic-name filtering for STRJRNLIB & CHGJRNOBJ commands
  - New user-defined table function, QSYS2/DISPLAY_JOURNAL, for simpler retrieval of journal entries with SQL (5.4 & 6.1 PTFs)

Ease of Use & Management Enhancements
IBM Tooling for DB2 for i

- IBM DB2 Web Query for i – New Version: 1.1.2
  - New InfoAssist Interface
  - Oracle JDEdwards OneWorld and EnterpriseOne Adapter
  - Performance, Usability and Security Enhancements

- IBM i Navigator – DB2 Management Interface

- IBM Information Management Products
  - IBM InfoSphere Data Architect
  - IBM InfoSphere CDC (Change-Data-Capture)
  - IBM Optim Data Growth Solution
  - IBM Optim Test Data Management & Data Privacy Solution
  - IBM Data Studio
    - SQL and Java Procedure development & debug
    - Wizard-based web service development
    - pureQuery runtime for Java developer productivity

IBM DB2 Web Query Enhancements

- New Product Version – 1.1.2
  - InfoAssist Interface
    - Simpler and faster report creation with ribbon-like interface
    - Report output flexibility
    - Full Compatibility with existing 1.1.1 reports
  - Oracle JDEdwards OneWorld and EnterpriseOne Adapter
  - Performance Enhancements
    - Optimized SQL translation
    - Enhanced join support and performance
    - Simplified Work Management with new Dedicated Subsystem
  - Usability
    - Dynamic report output formatting
    - Cancel long-running query interface
  - Security - DBA for segment, column, and value level security

- DB2 Web Query supports querying XML documents with IBM i 7.1

Bring BI and Query Solution Back to IBM i

- Stop the pain and expense of replicating data
- Make decisions with current data

DB2 Web Query provides an integrated Business Intelligence solution that offers rich functionality and breakthrough performance
IBM i 7.1 Enhancements:

OnDemand Performance Center
- Authority Simplification
- Index Advisor Improvements
- Database monitor
  - Client register filter
  - Errors only filter
- Show Statements - Variable replacement
- Enhanced SQL Details for a Job
- SQL Monitor integration
- Connecting QSQSRVR job info

Database Management
- OmniFind Text Index support
- Generate SQL – Privilege & CCSID
- Progress Status Monitors
  - Index Build
  - Table Alters
  - Enhanced Reorganize
- Object List enhancements
  - Performance of large lists
  - Object list filtering
  - Save list contents

Health Center
- SQL0901 Error Tracker
SQL Monitor – Client Register Filters

CL interface: STRDBMON OUTFILE(OUTMON1) JOB(*ALL) FTRCLTPGM(STRSQL)

Enhanced “SQL Details for a Job”

- SQL Performance Monitor integration
- Connecting QSQSRVR job information (5.4 & 6.1 PTFs)
**DB2 OnDemand Performance Center – User Authority Simplification**

*JOBCTL (Job Control Authority)*
- Whatever worked with *JOBCTL in IBM i 6.1 will continue to work

**QIBM_DB_SQLADM – Database Administrator**
- This is a database specific alternative to *JOBCTL. It is a superset of the function authorized by QIBM_DB_SYSMON.
- Examples:
  - Change parallel degree for DB2 SMP feature
  - Work with Plan Cache
  - Work with OmniFind Text Search Server

**QIBM_DB_SYSMON – Database Information**
- This allows a user to view some system level details, but not specifics about operations or anything related to changing or controlling the database.
- Examples:
  - QUSRJOBI for SQL information
  - Show SQL Information for Jobs

No Special Authority required when using OnDemand Performance Center with own job
- Starting and ending SQL Performance Monitors on your own job
- Analysis of SQL Monitor data and Plan Cache snapshots
- Visual Explain in Run SQL Scripts

---

**DB2 OnDemand Performance Center & Sensitive Data – SECURE columns**

- Prevents sensitive data values from being displayed in DB2 performance tools – Database Monitor & Plan Cache (5.4 & 6.1 PTFs)
  - Only security officer will be able to see sensitive values, **“SECURE” value presented to normal users (... WHERE cardnumber=:hostvar1 )**
  - User must register sensitive columns with DB2 tooling

- Registration interface is system stored procedure: SET_COLUMN_ATTRIBUTE
  - Procedure parameter descriptions
    - Table_Schema - System name of a table’s schema
    - Table_Name - System name of a table
    - Column_Name - System column name being secured.
    - Attribute - Secure attribute setting for column
      » SECURE NO
      » SECURE YES
  - Example:
    ```sql
    CALL SYSPROC.SET_COLUMN_ATTRIBUTE
    ('MYLIB1', 'ORDERS', 'CARDNUMBER', 'SECURE YES');
    ```

---

**User Authorization Commands:**

```
CHGFCNUSG FCNID(QIBM_DB_SQLADM)
USER(userid) USAGE(*ALLOWED)

CHGFCNUSG FCNID(QIBM_DB_SYSMON)
USER(userid) USAGE(*ALLOWED)
```
Additional Information

- **DB2 for i Websites**
  - Home Page: [ibm.com/systems/i/db2](http://ibm.com/systems/i/db2)
  - DeveloperWorks Zone: [ibm.com/developerworks/db2/products/db2i5OS](http://ibm.com/developerworks/db2/products/db2i5OS)
  - Porting Zone: [ibm.com/partnerworld/i/db2porting](http://ibm.com/partnerworld/i/db2porting)

- **Newsgroups & Forums**
  - USENET: comp.sys.ibm.as400.misc, comp.databases.ibm-db2

- **Education Resources - Classroom & Online**
  - ibm.com/systemi/db2/gettingstarted.html
  - ibm.com/partnerworld/wps/training/i5os/courses

- **DB2 for i Publications**
  - Online Manuals: [ibm.com/systems/i/db2/books.html](http://ibm.com/systems/i/db2/books.html)
  - DB2 for i Redbooks ([http://ibm.com/redbooks](http://ibm.com/redbooks))
    - Getting Started with DB2 Web Query for System i (SG24-7214)
    - OnDemand SQL Performance Analysis ... in V5R4 (SG24-7326)
    - Preparing for and Tuning the SQL Query Engine on DB2 for i5/OS (SG24-6598)
    - Modernizing iSeries Application Data Access (SG24-6393)

---

**Need help using the newest DB2 for i technologies?**

**Are you getting the most out DB2 for i?**

---

**IBM DB2 for i Consulting and Services**

- Database modernization
- DB2 WebQuery
- Database design, features and functions
- DB2 SQL performance analysis and tuning
- Data warehousing and Business Intelligence
- DB2 for i education and training

Contact: Mike Cain  
mcaion@us.ibm.com  
IBM Systems and Technology Group  
Rochester, MN USA