# Advanced SQL

Jim Mason

jemason@ebt-now.com

www.ebt-now.com

Web solutions for iSeries

*engineer, build, deploy, support, train*

508-728-4353

# What We'll Cover …

- SQL and Database environments
- Managing Database objects
- Managing data
- SQL Functions
- Advanced SQL Queries
- SQL Stored Procedures
- External Stored procedures

# What's NOT covered – a lot!

- ISeries Navigator, STRSQL
- Security
- Metadata and data modeling
- Data reporting and warehouses
- Database automation with ANT
- OLAP
- ETL and replication
- Triggers
- User-defined objects
- Automated SFTP
- XML
- Linked resources
- Custom functions
- Interoperability
- Application development
- Java data access
- WDSC database applications
- Web services for data
- Diagnostics
- Performance
- Testing
- New features – identity columns
- All details - LOL

# SQL standards

- Fortunately, SQL standards have been defined over the years
- Database vendors provide compliance info for database products
  - Many are on SQL-1999 or SQL-2003 now
- Applications that comply to standards work well on most databases
- Which databases don't follow standards? Microsoft Access
- SQL is not very different than CL
  - a flexible, keyword based command language
- Statements can be run individually
- Statements can be grouped in programs, procedures

# DB2/400 .. A little different

- OS/400 comes with DB2/400 embedded
- If you run a Linux partition or Windows, you may want to buy a database
- You have many choices for database vendors
  - IBM, Oracle, Microsoft, MySQL, Apache and more
- Vendors like IBM offer multiple database products
  - DB2/400, DB2 UDB, DB2 zOS, DB2 DW, DB2 Alpablox
- You can create 1 or more database instances of a given type
- DB2/400 is a single database instance
- Each data library is a schema in SQL, not a database

# DB administration options

- For DB2/400 you can use:
  - Green screen with STRSQL
  - iSeries Navigator
  - DDS
  - Open-source tools like Squirrel
- For other databases, the open-source tools work well

- Which is best?
  - For multi-platform: Squirrel
  - For visual administration: iSeries Navigator

# Interactive SQL

- For development and administration
- Choices: iSeries Navigator, STRSQL, Squirrel

```
set schema "JEM";
SELECT FIRSTNME AS FIRSTNAME, LASTNAME, SALARY,
  WORKDEPT FROM EMPLOYEE
WHERE LASTNAME >= 'M'
ORDER BY SALARY DESC;
```

# SQL calls from Java

```
String tableName = " EMPLOYEE  ";
String whereClause = "  ";
String orderClause = "  ";
String columns = " * ";
String stmt = " SELECT " + columns + " FROM " +
   tableName + whereClause + orderClause;
cdm.mgt.QueryProcessor qm = new
   cdm.mgt.QueryProcessor();
System.out.println(
   qm.getTextReportForSelectQuery(p)
   );
```

# RPG SQL calls

```
C/Exec Sql Declare C1 Cursor For
C+   Select
C+   SALARY + BONUS + COMM AS IPay,
C+   FIRSTNME AS IFirstName,
C+   LASTNAME AS ILastName
C+
C+   From EMPLOYEE
C+   Where EMPNO  >= :IEmpno
C+
C+  For Fetch Only -- Read Only Cursor
C/End-Exec
```

# What We'll Cover …

- SQL and Database environments
- Managing Database objects
- Managing data
- SQL Functions
- Advanced SQL Queries
- SQL Stored Procedures
- External Stored procedures

# Managing Database objects

- DDL – schema, table, view, procedures, index
- Schema – a logical partition for a database
  on AS/400, a library = a schema
- Table – physical file of records
- View – a logical view of 1 or more tables
  similar to a logical file on AS/400
- Index – a sort index for a table
  Similar to the keys in an AS/400 logical file
- Procedures
  custom SQL, Java or Programs run by a CALL
- Statements perform create, drop, alter on objects

# SQL data types = ILE data types

SQL Data types close to ILE RPG types

- BINARY
- NUMERIC
  - SMALLINT, INTEGER, BIGINT, FLOAT, DOUBLE, DECIMAL
- CHARACTER
  - CHAR, VARCHAR
- DATETIME
  - DATE, TIME, TIMESTAMP
- LOB
  - CLOB, BLOB
- UDT

You can cast between data types

# What We'll Cover …

- SQL and Database environments
- Managing Database objects
- Managing data
- SQL Functions
- Advanced SQL Queries
- SQL Stored Procedures
- External Stored procedures

# Managing data

- SELECT
  selects a result set or a scalar value from one or more tables and views
- Applications can use a cursor to navigate the result set of rows

```
SELECT * FROM EMPLOYEE
```

- 
- INSERT
- Inserts a row of data to a table, view or result set

```
INSERT INTO DEPARTMENT (DEPTNO,DEPTNAME,MGRNO,
   ADMRDEPT) VALUES ('A25','www.ebt-now.com','000010',
   'A25');
COMMIT;
```

- 
- UPDATE
  Updates a row of data in a table, view or result set

```
UPDATE DEPARTMENT SET (DEPTNAME, MGRNO) = ('SPIFFY
   COMPUTER SURE', '000010') WHERE DEPTNO = 'A00';
COMMIT;
```

# What We'll Cover …

- SQL and Database environments
- Managing Database objects
- Managing data
- SQL Functions
- Advanced SQL Queries
- SQL Stored Procedures
- External Stored procedures

# SQL Functions

- Functions operate on different types of data
- You can use functions to cast from 1 data type to another
- Functions can also create new values
- Functions can be combined in complex expressions
- ***Scalar versus Column functions***
- Column functions have multiple column values for an argument.
- Scalar functions have a single value argument.
- Below, SUM returns a single value for a set of bonus amounts.
- Interactive example:

```
SELECT SUM(BONUS) FROM EMPLOYEE
  WHERE JOB = 'CLERK'
```

- Generates

```
2800
```

# Character functions

CHAR
- converts another argument to character value

CONCAT
- concatenate 2 strings

SUBSTR
- return a subset of a string

- CONCAT example

```
SELECT CONCAT('S_', CONCAT(TRIM(CHAR(7)),
TRIM(CHAR(3))))
  FROM EMPLOYEE WHERE EMPNO = '000010'
```

- Result

```
S_73
```

# Substring function

- SUBSTR example

```
SELECT SUBSTR(LASTNAME, 1, 5)
  FROM EMPLOYEE
  WHERE EMPNO = '000070'
```

- Returns

```
'PULAS'
```

# Numeric functions

- Many numeric functions exist.
- Simple example

```
SELECT SUM(BONUS) AS BSUM, ROUND(AVG(BONUS),0) AS
   BAVERAGE
          FROM EMPLOYEE
          WHERE JOB = 'CLERK'
```

- Generates sum of bonus and avergae bonus for Clerks

```
2800, 467
```

# Date functions

- Many Date and time functions exist
- DAYOFWEEK example
- Returns day number from 1 (Sunday ) to 7 (Saturday)

```
     SELECT LASTNAME, EMPNO,
DAYOFWEEK(HIREDATE)
       FROM EMPLOYEE
       WHERE EMPNO = '000070'
```

- Result

  3

# What We'll Cover …

- SQL and Database environments
- Managing Database objects
- Managing data
- SQL Functions
- Advanced SQL Queries
- SQL Stored Procedures
- External Stored procedures

# Sub queries

- Advanced queries may contain many predicates, clauses, functions and use sub-queries (or nested queries) along with common table expressions (generated table results)

```
SELECT D.DEPTNO, D.DEPTNAME, EMPINFO.AVGSAL,
   EMPINFO.EMPCOUNT FROM DEPARTMENT D,
   (SELECT ROUND(AVG(E.SALARY),0) AS AVGSAL,COUNT (*) AS
   EMPCOUNT   FROM EMPLOYEE E WHERE E.WORKDEPT =
       (SELECT X.DEPTNO FROM DEPARTMENT X WHERE
   X.DEPTNO = E.WORKDEPT )  )
AS EMPINFO
```

- Results in:

| DEPTNO | DEPTNAME | AVGSAL | EMPCOUNT |
|---|---|---|---|
| A00 | SPIFFY COMPUTER SURE | 27,304 | 32 |
| B01 | PLANNING | 27,304 | 32 |
| C01 | INFORMATION CENTER | 27,304 | 32 |

# Common table expression

- Where it fits, a WITH clause is more efficient than sub-query
  - Sub-query (nested query) executes for every iteration
- Can also do calculations in common table expression

```
WITH DEPTS ( DEPTNO, DEPTNAME )
AS (SELECT DEPTNO, DEPTNAME FROM DEPARTMENT )
SELECT FIRSTNME AS FIRSTNAME, LASTNAME,
   DEPTS.DEPTNAME FROM EMPLOYEE, DEPTS
ORDER BY DEPTNAME DESC
```

- Result

| FIRSTNAME | LASTNAME | DEPTNAME |
|---|---|---|
| CHRISTINE | HAAS | SUPPORT SERVICES |
| MICHAEL | THOMPSON | SUPPORT SERVICES |
| SALLY | KWAN | SUPPORT SERVICES |

# Sub-query with derived table

- Using a sub-select, create a derived table.
- Find the highest salaried employees in each department

```
SELECT EMPNO, LASTNAME, SALARY, WORKDEPT FROM
   EMPLOYEE
WHERE SALARY IN
(SELECT MAX_SAL FROM
(SELECT DEPTNAME, MAX(SALARY) AS MAX_SAL,
   COUNT(*) AS EMP_COUNT FROM EMPLOYEE, DEPARTMENT
WHERE EMPLOYEE.WORKDEPT=DEPARTMENT.DEPTNO
GROUP BY DEPTNAME)
AS DEPT_TOT) ORDER BY LASTNAME
```

| | | | |
|---|---|---|---|
| 50 | GEYER | 80,175 | E01 |
| 10 | HAAS | 152,750 | A00 |
| 90 | HENDERS | 89,750 | E11 |

# Find potential duplicate records

- Use an inner join on same table to find similar records
- Search on similar last names for duplicates.

```
SELECT A.LASTNAME, A.FIRSTNME, A.EMPNO, A.JOB
FROM EMPLOYE2 A INNER JOIN EMPLOYE2 B
ON A.LASTNAME=B.LASTNAME
AND A.SALARY=B.SALARY
AND A.EMPNO<>B.EMPNO
ORDER BY A.LASTNAME
```

| LASTNAME | FIRSTNME | EMPNO | JOB |
|---|---|---|---|
| HAAS | CHRISTINE | 10 | PRES |
| HAAS | CHRISY | 11 | PRES |
| THOMPSON | MICHAEL | 20 | MANAGER |
| THOMPSON | BIG MIKE | 22 | MANAGER |

# What We'll Cover …

- SQL and Database environments
- Managing Database objects
- Managing data
- SQL Functions
- Advanced SQL Queries
- SQL Stored Procedures
- External Stored procedures

# Stored procedures

- Stored Procedures can be defined as either SQL or an external program
- SQL procedures are described by SQL statements
- iSeries program can be used as an externally defined procedure
- Databases add special support to generate Java code as procedures
- SQL Procedures are defined by statements
- An SQL procedure consists of:
  - A procedure name
  - A sequence of parameter declarations
  - The procedure properties (defining number of result sets, and the kind of SQL access that is included into the stored procedure)
  - A set of parameters that control the way in which the stored procedure is created
  - An SQL routine statement body

# Creat and call a procedure

- Procedures created with CREATE PROCEDURE
- CALL will call a procedure in the database
- The procedure can have input and output parameters
- A procedure can also return a cursor to a result set
- To call a procedure it must first be created.
- You can also declare a procedure prior to a call
  - Procedures can be created from SQL statements
  - Procedures can be created from existing programs
  - RPG, COBOL, C++, Java
  - Databases often can generate a Java procedure from source

# Create SQL read-only procedure

- This simple procedure adds 2 input numbers returning the sum as the 3rd output parameter
- The create statement can be done interactively

```
CREATE PROCEDURE JEM.ZADD
(IN P1 INTEGER, IN P2 INTEGER, OUT P3 INTEGER)
LANGUAGE SQL SPECIFIC JEM.ZADD
S1: BEGIN
SET P3 = P1 + P2;
END S1
```

# Call SQL procedure

- Calling this procedure from Java with a call statement

```
String callString = "CALL JEM.ZADD (?, ?, ?)";
cstmt = (java.sql.CallableStatement)
   conn.prepareCall(callString);
cstmt.setInt(1, i1);
cstmt.setInt(2, i2);
cstmt.registerOutParameter(3,
   java.sql.Types.INTEGER);
cstmt.execute();
i3 = cstmt.getInt(3);
```

# Create SQL update procedure

- A procedure to update an employee's salary by a percent

```
CREATE PROCEDURE ZUPD_SAL1
(IN EMPLOYEE_NUMBER CHAR(6),
IN RATE DECIMAL(6,2))
LANGUAGE SQL MODIFIES SQL DATA

UPDATE JIM.EMPLOYE2
SET SALARY = SALARY * RATE
WHERE EMPNO = EMPLOYEE_NUMBER
```

# Call SQL update procedure

- calling the SQL update procedure
- The employee's salary is increased by 10%

```
String callString = "CALL JEM.ZUPD_SAL1 (?, ?)";
java.math.BigDecimal rate = new
  java.math.BigDecimal("1.1");
String empno = "000070";
cstmt = (java.sql.CallableStatement)
  conn.prepareCall(callString);
cstmt.setString (1, empno);
cstmt.setBigDecimal(2, rate );
cstmt.execute();
```

# What We'll Cover …

- SQL and Database environments
- Managing Database objects
- Managing data
- SQL Functions
- Advanced SQL Queries
- SQL Stored Procedures
- External Stored procedures

# Using existing programs as procedures

- Instead of using SQL statements to define a procedure
- Create a procedure as a call to an external program
- 2 types of program call formats: Java and General
- All programs are called using the General format except Java
- Parameters can be used as input, output or both
- Program can return 1 or more result sets
- Cursors need to be declared in the client to access result sets

- Why use RPG as a stored procedure?
  - You can take MANY existing programs and reuse the results in Web applications easily
  - Simpler than rewriting RPG apps using CGI-DEV

# Run AS/400 program from PC

- Create a CL program that takes a generic command string
- Create an external stored procedure that references the CL program
- CALL the CL program in SQL passing a command and checking the result error code

```
/* ZTST002C:  RUN A CMD FROM A REMOTE CLIENT IN
THIS JOB                  */
/* PARMS
*/
/*   1   CMD TO RUN
*/
/*   2   CMD SIZE - DEC(15.5)
*/
/*   3   RESULT
*/
```

# Create an RPG procedure

```
create procedure empgc02
 ( IN EMPNO CHAR(6), OUT FLAG CHAR(5))
DYNAMIC RESULT SET 1
RESULT SET 1
LANGUAGE RPGLE
READS SQL DATA
FENCED
EXTERNAL NAME 'JMASON3/EMPGC02'
PARAMETER STYLE GENERAL
```

# Summary on Advanced SQL

- Get started the easy way:
  - For administration, use a simple, open-source client or iSeries Navigator
  - For developers, use an IDE with DB2/400 support: WDSC, Eclipse WTP, MyEclipse
  - Separate business transaction from reporting workloads
  - Leverage tools to analyze SQL performance
  - Leverage SQL views to simplify data reporting for end users
  - Leverage RPG programs as external procedures to access complex data
  - Leverage open-source reporting tools ( BIRT, QWBERS ) for user reporting
  - Leverage advanced DB2 connectivity options for enterprise reporting