

Intro to the iSeries

Embedded SQL

Objectives

- Explain Embedded SQL concepts
- Show how to code Embedded SQL in RPG IV

What is Embedded SQL?

- SQL is a language used to retrieve data
- Embedded SQL is coding SQL statements in a HLL

Embedded SQL

- SQL can be used with other programming languages
- Instead of using a RPG CHAIN, use SELECT
- Must use a SQLRPG compiler
- SQL statements have “special” syntax within the program

Embedded SQL

- Each language has its own syntax and commands for identifying SQL statements

```
      :           :           :           :           :           :           :  
MOVE PAY_RATE TO EMP_PAY_RATE  
EXEC SQL  
      SELECT HRS FROM MYCLCXX/TC  
      WHERE SSN = 123456789  
END-EXEC  
PAY = EMP_PAY_RATE * HRS  
      :           :           :           :           :           :           :
```

Coding in RPG IV

- Start with EXEC SQL
- SQL code
- SQLCA is automatically included when you compile

Compiling

- Not a normal compile - two steps
 - ◆ Pre-compile – prepares SQL
 - ◆ Compile – translates into ML

```
      :           :           :           :           :           :           :  
MOVE PAY_RATE TO EMP_PAY_RATE  
CALL SQLINTERFACE (' SELECT HRS  
                    FROM MYCLCXX/TC WHERE SSN =  
                    123456789')  
PAY = EMP_PAY_RATE * HRS  
      :           :           :           :           :           :           :
```

RPG IV & SQL

- For RPG IV program source should be SQLRPGLE
- Use Option 14 when compiling

Host Variables

- Host variables are preceded by a colon (:)
- Host variables are a field defined in the RPG IV programs
- They are the parameters that are passed to the SQL statement

Special SQL commands

- Small problem with SQL and most programming languages:
 - ◆ SQL returns more than one record/row at a time
 - ◆ Most PL's work on one record at a time

uh-oh !

- Solution: Cursors and FETCH

SQL Cursors

- DECLARE CURSOR: SQL Select statements
- OPEN: Opens the cursor
- FETCH: Reads in 1 row from the cursor
- CLOSE: Closes the cursor

Special SQL commands

- Cursors can be positioned to rows within the returned table of data
- Fetch retrieves the row the cursor is pointing at

```
MOVE pay_rate TO emp_pay_rate
```

```
EXEC SQL
```

```
DECLARE mycursor CURSOR FOR
```

```
SELECT hrs FROM myclcxx/tc
```

```
OPEN mycursor
```

```
FETCH mycursor INTO :emp_hrs_worked
```

```
END-EXEC
```

```
PAY = emp_pay_rate * emp_hrs_worked
```

SQL Program Commands

- Information is supplied to the program in the SQLCA after each I/O (like a LDA)
- SQLCA is comprised of a series of fields
- The field SQLCODE indicates whether a read is successful or not
 - ◆ 100 means row not found (EOF)
 - ◆ Negative number means serious error

SQL Program Commands

- You should check the value after each I/O
- Check SQLCODE value using keywords:
 - ◆ NOTFOUND (100)
 - ◆ SQLERROR (negative number)
- **WHENEVER** will always check (like a global MONMSG)

WHENEVER NOTFOUND GO TO endofpgm

Check Status

- Check `SQLSTAT` or `SQLCODE` after each SQL statement
- Check for OK:
 - ◆ `SQLSTAT = '00000'`, `SQLCODE = 0`
- Check for EOF:
 - ◆ `SQLSTAT = '02000'`, `SQLCODE = 100`

Points to Remember

- IBM has said “SQL is the future” and “there will be no more enhancements to the native tools”