Queries in PSM

- The following rules apply to the use of queries:
  1. Queries returning a single value can be used in assignments.
  2. Queries returning a single tuple can be used with INTO.
  3. Queries returning several tuples can be used via a cursor.

Cursors

- A cursor serves as a tuple-variable that ranges over the tuples of the result of a query.
  DECLARE c CURSOR FOR (<query>);
- Opening a cursor evaluates <query>.
  OPEN c;
- Closed with CLOSE c;

Fetching Tuples From a Cursor

- Get next tuple:
  FETCH c INTO a1, a2, ..., ak;
  - a1, a2, ..., ak are the attributes of the result of the query of c.
  - c is moved to the next tuple.
- A cursor is used by creating a loop around FETCH.

End of Cursor

- SQL operations return status in SQLSTATE (in PSM).
- FETCH returns ‘02000’ in SQLSTATE when no more tuples are found.
- Useful declaration:
  DECLARE NotFound CONDITION FOR SQLSTATE ‘02000’

Cursor Structure

DECLARE c CURSOR FOR...
  ...
cursorLoop: LOOP
    ...
    FETCH c INTO...;
    IF NotFound THEN LEAVE cursorLoop;
    END IF;
    ...
  END LOOP;
Cursor Example

- Write a procedure that makes free all beers sold for more than $5 at Spoon.
  
  ```sql
  CREATE PROCEDURE FreeBeer()
  DECLARE aBeer VARCHAR[30];
  DECLARE aPrice REAL;
  DECLARE NotFound CONDITION FOR SQLSTATE '02000';
  DECLARE CURSOR c FOR
      SELECT beer, price FROM Sells WHERE bar = 'Spoon';
  BEGIN
    OPEN c;
    menuLoop: LOOP
      FETCH c INTO aBeer, aPrice;
      IF NotFound THEN LEAVE menuLoop END IF;
      IF aPrice > 5.00 THEN
        UPDATE Sells
        SET price = 0
        WHERE bar = 'Spoon' and beer = aBeer;
      END IF;
    END LOOP;
    CLOSE c;
  END;
  ```

Example

```sql
BEGIN
  OPEN c;
  menuLoop: LOOP
    FETCH c INTO aBeer, aPrice;
    IF NotFound THEN LEAVE menuLoop END IF;
    IF aPrice > 5.00 THEN
      UPDATE Sells
      SET price = 0
      WHERE bar = 'Spoon' and beer = aBeer;
    END IF;
  END LOOP;
  CLOSE c;
END;
```

MySQL Routines

- MySQL's version of PSM (Persistent, Stored Modules).
  - Stored procedures.
  - Functions.
- Brand new feature (in 5.0).
  - Adheres to standards (similar to IBM's DB2, different from Oracle PL/SQL).
  - Bugs possible (bugs.mysql.com)

Procedures

```sql
CREATE PROCEDURE <name>(<arglist>)
BEGIN
  <declarations>
  <statements>
END;
```

Functions

```sql
CREATE PROCEDURE <name>(<arglist>)
RETURNS <type>
BEGIN
  <declarations>
  <statements>
END;
```

Arguments

- Argument list has name-mode-type triples.
  - Mode: IN, OUT, or INOUT for read-only, write-only, read/write, respectively.
  - Types: standard SQL.
Example

• A procedure to add a beer and price to Spoon’s menu:

```
DELIMITER //
CREATE PROCEDURE addSpoonMenu(
    IN b CHAR(20),
    IN p REAL)
BEGIN
    INSERT INTO Sells
    VALUES('Spoon', b, p);
END;
DELIMITER ;
CALL addSpoonMenu('Guinness', 7.50);
```

Declarations

• Variables
• Conditions
• Cursors
• Handlers
• Must be declared in this order!

Conditions

```
DECLARE <condName>
    CONDITION FOR SQLSTATE <errorStr>
```

```
DECLARE <condName>
    CONDITION FOR <errorNumber>
```

• The following conditions are predefined:
  – NOT FOUND (no more rows)
  – SQL EXCEPTION (error)
  – SQLWARNING (warning)

Handlers

• Define what to do in case of errors (or conditions)

```
DECLARE { EXIT | CONTINUE }
    HANDLER FOR
    {<errorNum> | SQLSTATE <errorStr> | <condName> }
    SQL statement
```

• Common practice: set a flag for CONTINUE handlers and check inside stored procedure.

Body Constructs

• Assignments:
  SET <variable> = <expression>
  – Variables must be declared.

• Branches
  IF <condition> THEN
  <statement(s)>
  ELSE
  <statement(s)>
  END IF;

Queries in Routines

1. *Single-row selects* allow retrieval into a variable of the result of a query that is guaranteed to produce one tuple.
2. *Cursors* allow the retrieval of many tuples, with the cursor and a loop used to process each in turn.
Cursors in MySQL

- The cursor declaration is:
  DECLARE <curName>
  CURSOR FOR <query>;
- Fetching is done with:
  FETCH c INTO <variables>;

Example (1/3)

- The FreeBeer in MySQL:
  CREATE PROCEDURE FreeBeer()
  BEGIN
  DECLARE aBeer CHAR(20);
  DECLARE aPrice REAL;
  DECLARE flag INT DEFAULT 0;

Example (2/3)

  DECLARE menu CURSOR FOR
  SELECT beer, price
  FROM Sells
  WHERE bar = 'Spoon';
  DECLARE CONTINUE HANDLER
  FOR NOT FOUND
  SET flag = 1;

Example (3/3)

  OPEN menu;
  REPEAT
    FETCH menu INTO aBeer, aPrice;
    IF aPrice > 5.00 THEN
      UPDATE Sells
      SET price = 0
      WHERE bar = 'Spoon' AND beer = aBeer;
    END IF;
  UNTIL flag = 1
  END REPEAT;
  CLOSE menu;
  END;//