CS2102 Database Systems

Database Application Development

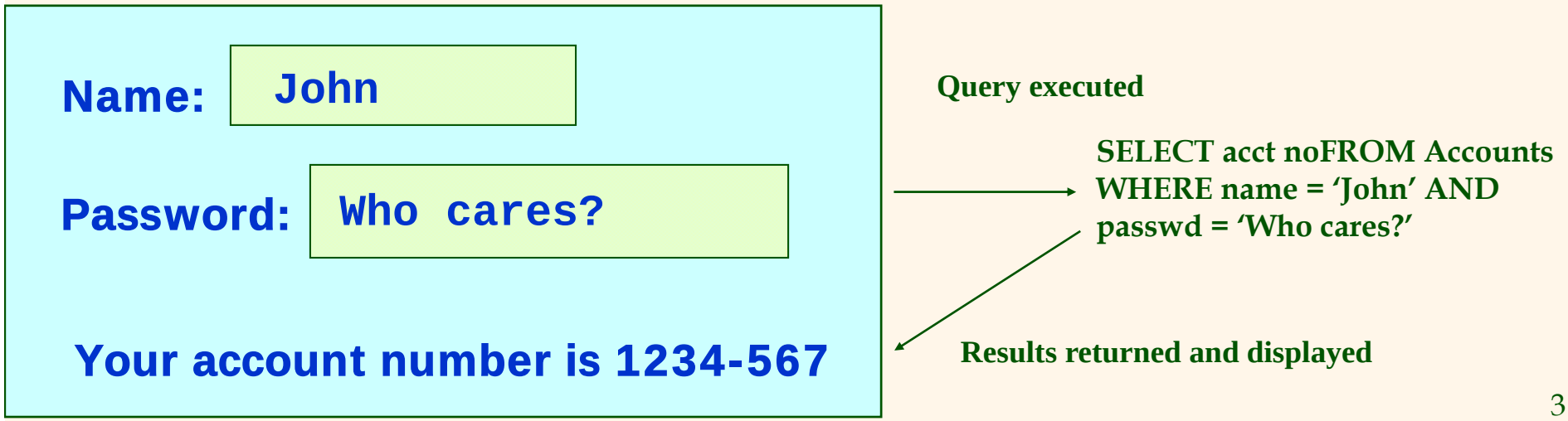
SQL in Practice

- We have seen how SQL is used at the generic query interface, where we sit at a terminal and ask queries of a database
- In practice, SQL queries are often constructed by programs, and these queries may take constants from users

Example

- Relation: Accounts(acctno, name, passwd)
- Web interface: Get name and password from user, store in strings *n* and *p*, issue query, display account number.

```
SELECT acctno FROM Accounts
WHERE name = :n AND passwd = :p
```



SQL in Application Code

- Application that use DBMS to manage data run as separate process and connect to DBMS to interact with it
  - Include a statement to connect to database
  - SQL commands to insert/delete/modify data can be called within a host language (C++/Java) program.
  - SQL statements can refer to host variables.

SQL in Application Code

- SQL queries can be used to retrieve data BUT need to bridge how DBMS sees data and how application program sees data
- Impedance mismatch
  - SQL relations are (multi-)sets of records
  - No such data structure in programming language such as C++/Java
  - SQL supports a mechanism called a cursor to handle this

SQL in Application Code

- Approaches:
  - Embed SQL in a host language (Embedded SQL)
  - Create special API to call SQL commands (JDBC, PHP)

Embedded SQL

- Embed SQL in a host language
- A preprocessor converts SQL statements into special calls to database
- Then a regular compiler compiles the code

- Embedded SQL statements follow the basic form:
  - EXEC SQL <*SQL Statement*> ;

Embedded SQL

- Connect to database
  - EXEC SQL CONNECT TO *connection_target*
    [AS *connection_name*]
    [USER *username*]
    [USING *password*];

- Disconnect from database
  - EXEC SQL DISCONNECT *connection_target*

Embedded SQL

- Declare shared variables between SQL and host
  - EXEC SQL BEGIN DECLARE SECTION

    char c_title[20];
    char c_director[20];
    int c_myear;
    float c_rating;

    Naming convention c_ indicate host language var

    EXEC SQL END DECLARE SECTION
- Two special error variables
  - SQLCODE (long, is negative if error occur)
  - SQLSTATE (char[6], predefined codes for common errors)

Embedded SQL

- SQL statements
  - EXEC SQL

    Insert into Movies values

    (:c_title, :c_director, :c_myear, :c_rating);

  - Insert a row into Movies relation.
  - Column values are based on the values of the host language variables
  - Shared variables are preceded by a colon in SQL

Cursors

- We can declare a cursor on a relation or query statement which generates a relation
- Open a cursor, and repeatedly fetch a tuple then move the cursor, until all tuples have been retrieved
- Use ORDER BY clause to control the order in which tuples are returned
- Can also modify or delete the tuple pointed to by cursor

Cursors

- Example: Cursor that gets titles of movies, in alphabetical order

  - EXEC SQL DECLARE minfo CURSOR FOR

    SELECT M.title

    FROM Movies M

    WHERE M.rating > 5

    ORDER BY M.title;

Example: Embed SQL in C

```
char SQLSTATE[6];
EXEC SQL BEGIN DECLARE SECTION
char c_title[20]; char c_director[20];
int c_myear; float c_rating;
EXEC SQL END DECLARE SECTION
EXEC SQL DECLARE minfo CURSOR FOR
    SELECT M.title, M.myear
    FROM Movies M
    WHERE M.rating > 5
    ORDER BY M.title;
EXEC SQL OPEN minfo
do {
    EXEC SQL FETCH minfo INTO :c_title, :c_myear;
    printf ("%s is made in %d\n", c_title, c_myear);
while (SQLSTATE != '02000');
EXEC SQL CLOSE minfo;
```

Position cursor just before first row

Cursor point to next row and column values are copied into corresponding host variables

Denotes NO DATA

Dynamic SQL

- SQL query strings are not always known in advance, e.g., graphical DBMS frontend
- Construct SQL statements on-the-fly
- Example:

```
char c_sqlstring[] =
    {"Delete from Movies where rating<5"};
EXEC SQL PREPARE dynamicquery FROM :c_sqlstring;
EXEC SQL EXECUTE dynamicquery;
```

Declare C variable and initialize value to string representation of an SQL command

Parse string and compile as SQL command. Resulting executable bound to SQL variable called dynamicquery

Database APIs

- Embedded SQL enables integration of SQL with general purpose programming language.
  - Requires compilation
  - Final executable works with one specific DBMS.
- Another approach to connect database to conventional language by using library calls
  - C + CLI (Call-Level Interface)
  - Java + JDBC
  - PHP + PEAR/DB

Next Week is e-Learning Week

- Breeze lecture in IVLE on

  SQL and Programming Languages

  - Procedural SQL or **PL/SQL** which is the procedural extension of Oracle 9i - no connection to DBMS is needed as program is executed directly in database.
  - Database connectivity – ODBC, **JDBC**
  - Embed SQL in Java or **SQLj**

- Design report for project is due 13 Sept
  - Submit to your tutor