Tecnologie di Sviluppo per il Web

Programmazione su Basi di Dati: Tecnologie

versione 3.0

G. Mecca – mecca@unibas.it – Università della Basilicata

Sommario

- Panoramica
- SQL Immerso (“Embedded SQL”)
  - ⇨Cursori
- Call Level Interface (CLI)
- Stored Procedures
- In Questo Corso

Panoramica

- Riassumiamo i problemi principali
  - ⇨1. Stabilire una connessione con il DBMS
  - ⇨2. Inviare istruzioni SQL al DBMS
  - ⇨3. Gestire il risultato (collezione di ennuple)
- Storicamente
  - ⇨architetture e soluzioni tecnologiche diverse per questi problemi

Panoramica

- In questa lezione
  - ⇨una panoramica sulle architettura tecnologiche
- In particolare
  - ⇨SQL Immerso (“Embedded SQL”)
  - ⇨Call Level Interface
  - ⇨“Stored Procedures”

SQL Immerso

- SQL Immerso (“Embedded SQL”)
  - ⇨è la soluzione tradizionale, molto usata fino a qualche anno fa
- Due varianti
  - ⇨SQL statico
  - ⇨SQL dinamico
  - ⇨ci concentreremo principalmente sull’SQL statico

SQL Immerso

- Il presupposto architetturale
  - ⇨il DBMS deve avere un modulo (libreria) scritto nel linguaggio di programmazione utilizzato (es: C)
  - ⇨questa libreria deve esporre delle funzioni per effettuare connessioni di rete (utilizzando i socket), eseguire istruzioni SQL e prelevare i risultati

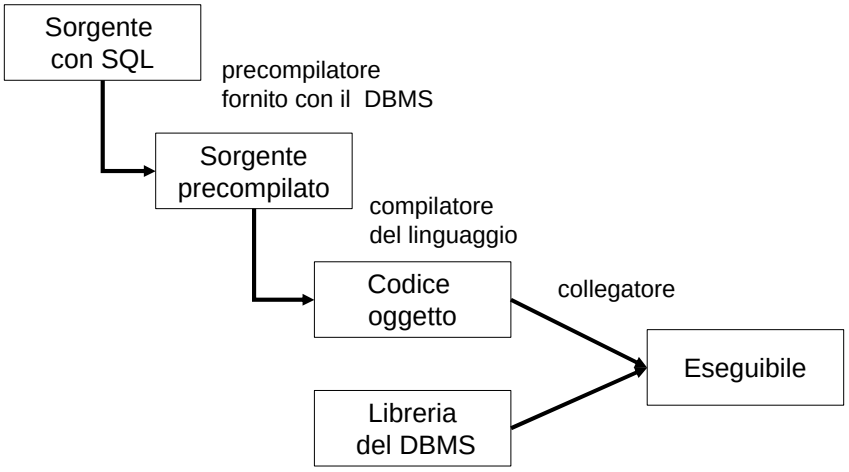
SQL Immerso

- Idea fondamentale
  - ⇨il codice SQL diventa parte del codice del linguaggio di programmazione (“linguaggio ospite”)
  - ⇨le istruzioni SQL sono introdotte dalle parole chiave EXEC SQL
  - ⇨il codice viene precompilato utilizzando un precompilatore per l’SQL fornito a corredo del DBMS

SQL Immerso

- Idea fondamentale (continua)
  - ⇨il precompilatore traduce le istruzioni SQL in chiamate alle funzioni della libreria del DBMS
  - ⇨il codice oggetto generato dal compilatore deve essere collegato con il codice della libreria fornita con il DBMS
  - ⇨normalmente precompilatori e librerie e per C/C++, Pascal, Cobol, FORTRAN

SQL Immerso



SQL Immerso: Esempio

```
int main() {
  printf("Avvio dell'applicazione");
  exec sql connect to universita
           user 'pguser' identified by 'pguser';
  printf("Connessione effettuata");
  exec sql create table studente
         (matricola integer primary key,
          nome varchar(20),
          annodicorso integer);
  printf("Tabella creata correttamente");
  exec sql disconnect;
  return 0;
}
```

SQL Immerso: Esempio

```
/* These three include files are added by the preprocessor */
#include <ecpgtype.h>
#include <ecpglib.h>
#include <ecpgerrno.h>
#include <sqlca.h>

int main() {
  ECPGconnect(__LINE__, "universita" , "pguser" , "pguser" ,
   NULL, 0);

  ECPGdo(__LINE__, NULL, "create  table studente ( matricola
   integer   primary key  , nome varchar ( 20 )     , annodicorso
   integer    )", ECPGt_EOIT, ECPGt_EORT);

  ECPGdisconnect(__LINE__, "CURRENT");

  return 0;
}
```


SQL Immerso

- Istruzioni SQL
  - ⇨sono fissate a tempo di compilazione
- Varianti sintattiche
  - ⇨è possibile utilizzare le variabili del linguaggio nell'istruzione SQL
  - ⇨sintassi: :<nomevariabile> es: :matricola
  - ⇨le variabili devono essere "annunciate" al preprocessore in una "declare section"

SQL Immerso

- Variabili di controllo
  - ⇨il preprocessore mette a disposizione del programmatore una struttura chiamata sqlca, che contiene diverse variabili
- Variabile SQLCODE (sqlca.sqlcode)
  - ⇨serve a verificare l'esito dell'ultima istruzione eseguita
  - ⇨valori >=0 segnalano esito corretto

SQL Immerso

```
void inserisciStudente() {
   exec sql begin declare section;
     int matricola, annocorso;
     char nome[20];
   exec sql end declare section;

   exec sql connect to universita user pguser
                                identified by pguser;

   if (sqlca.sqlcode>=0) {
     printf("Matricola: ");     scanf("%d",&matricola);
     printf("Nome: ");          scanf("%s",&nome);
     printf("Anno di corso: "); scanf("%d",&annocorso);

     exec sql insert into studente
              values (:matricola,:nome,:annocorso);
     exec sql disconnect;
   }
   return;
}
```

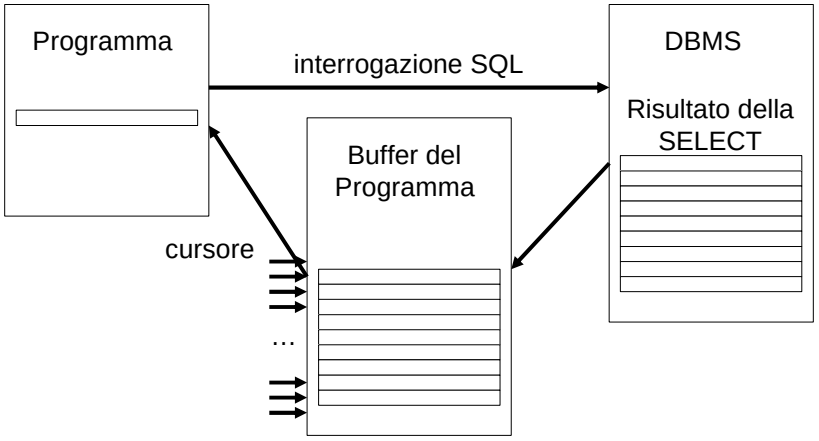
SQL Immerso

- Come gestire il risultato di una SELECT ?
  - ⇨la SELECT restituisce una collezione di ennuple
  - ⇨è necessario “scandire” il risultato ed acquisire le ennuple ad una ad una
  - ⇨per farlo si utilizza una struttura di dati detta “cursore”

Cursore

- Cursore
  - ⇨primitiva di programmazione per effettuare la scansione del risultato di una select
  - ⇨consiste di un buffer di memoria in cui viene memorizzato il risultato della select (lista di record)
  - ⇨e di un puntatore per scandire la lista
- Operazioni sul cursore
  - ⇨open, fetch (oppure next), close

Cursore



Cursore: Esempio

```
void stampaStudentiPrimoAnno() {
   exec sql begin declare section;
    int matricola;
    char nome[20];
   exec sql end declare section;

   exec sql declare CX cursor for
            select matricola,nome from studenti
            where annocorso=1;
   exec sql open CX;
   exec sql fetch CX into :matricola, :nome;

   while (sqlca.sqlcode >= 0) {
      printf("Studente %d %s", matricola, nome);
      exec sql fetch CX into :matricola, :nome;
   }
   exec sql close CX;
   return;
}
```

Call Level Interface

- Vantaggi dell'SQL Immerso
  - ⇨prestazioni (precompilazione ottimizzata)
- Problemi dell'SQL Immerso
  - ⇨l'applicazione dipende dal precompilatore
  - ⇨difficile sviluppare su DBMS diversi
- Call Level Interface
  - ⇨soluzione per rendere la programmazione indipendente dal DBMS

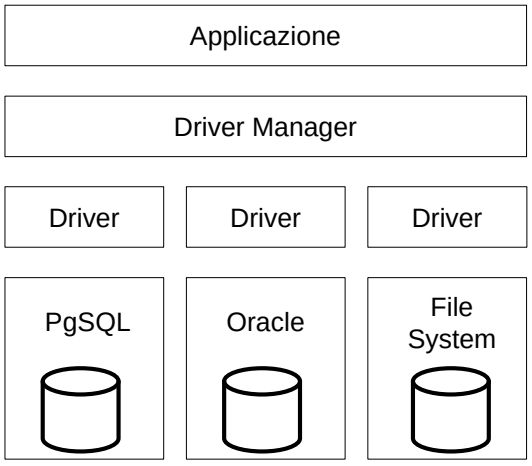
Call Level Interface

- Caratteristiche di CLI
  - ⇨la comunicazione client/server è basata su una API standardizzata (e non proprietaria)
  - ⇨il codice SQL non è immerso nel codice sorgente, ma viene inviato al DBMS sotto forma di stringhe
  - ⇨il client deve disporre di un opportuno modulo, detto "driver" che implementa l'API e gestisce la comunicazione con il DBMS

Call Level Interface

- Driver
  - ⇨modulo sul lato del client che implementa lo standard (API di comunicazione)
  - ⇨dal momento che il codice non viene precompilato, il client può utilizzare diversi driver contemp. per accedere a diversi DBMS
- Driver manager
  - ⇨modulo sul lato del client che gestisce i diversi driver disponibili

Call Level Interface



Call Level Interface

- Esempi di CLI
  - ODBC
  - JDBC
- ODBC (“Open DataBase Connectivity”)
  - tecnologia Microsoft
  - implementazione di SQL/CLI (1995)
  - insieme di segnature di funzioni C
  - es: funzione per eseguire la connessione

Call Level Interface

- JDBC (>>)
  - tecnologia Java
  - insieme di interfacce e classi Java contenute nel package java.sql
  - es: interfaccia Connection con metodi per la connessione

Call Level Interface

- Vantaggi rispetto a SQL immerso
  - ⇨indipendenza dell'applicazione rispetto al DBMS (API standard)
  - ⇨possibilità di accedere più di una sorgente di dati nella stessa applicazione
- Svantaggi rispetto ad SQL immerso
  - ⇨può essere meno efficiente se il driver non è ottimizzato
  - ⇨architettura più complessa (driver manager)

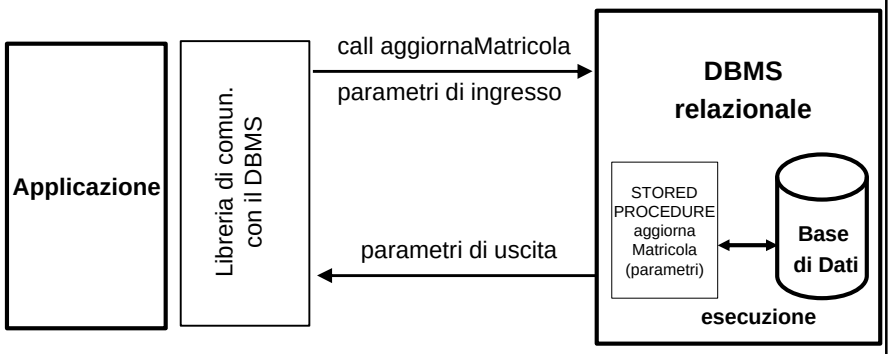
Stored Procedures

- Negli approcci visti finora
  - ⇨la logica applicativa (procedure di calcolo) viene eseguita sul client
  - ⇨invia richieste SQL al server
  - ⇨riceve ed elabora i risultati
  - ⇨esiste però un approccio alternativo
- "Stored Procedures"
  - ⇨procedure di calcolo che vengono eseguite direttamente sul DBMS

Stored Procedures

- Stored Procedure
  - ⇨modulo del DBMS scritto in un linguaggio ibrido, parte procedurale, parte SQL, per eseguire operazioni su una base di dati
  - ⇨comunica con l'esterno attraverso parametri
- Parametri
  - ⇨IN: parametri esclusivamente d'ingresso
  - ⇨OUT: parametri esclusivamente d'uscita
  - ⇨INOUT: parametri di ingresso e uscita

Stored Procedures



Stored Procedures

- Linguaggio di programmazione
  - ogni DBMS ha il suo linguaggio per scrivere Stored Procedures
  - es: Oracle: Pl/SQL
  - es: PostgreSQL: Pl/PgSQL
- Tipicamente
  - linguaggio procedurale (variabili, tipi, strutture di controllo ecc.)
  - con SQL immerso

Stored Procedures

- Creazione della Stored Procedure
  - in Pl/SQL: istruzione CREATE PROCEDURE
- Si specifica
  - il nome della procedura
  - i parametri (nome e tipo)
  - il codice della procedura
- Dal quel momento
  - la procedura fa parte del catalogo della bd

Stored Procedures: Esempio in PI/SQL

- Esempio
  - ⇨una stored procedure che data una città calcola e restituisce il numero di studenti residenti nella città
- Parametri
  - ⇨città: parametro IN
  - ⇨numero studenti: parametro OUT

Stored Procedures: Esempio in PI/SQL

```
CREATE PROCEDURE contaCitta (
    p_citta IN CHAR,
    p_numStudenti OUT NUMBER) AS
BEGIN
  select count(*)
  from studenti
  where cittadiresidenza = p_citta
      INTO p_numStud;
END
```

Stored Procedures

- Approcci applicativi alla prog. sul DBMS
  - ⇨due approcci completamente diversi
- Approccio con architettura “aperta”
  - ⇨logica applicativa separata dai dati
  - ⇨modello, vista, controllo
- Approccio con architettura “legacy”
  - ⇨logica applicativa contenuta nel DBMS
  - ⇨il client contiene solo vista e controllo

Stored Procedures

- Architettura legacy
  - ⇨architettura applicativa centrata sul DBMS
  - ⇨approccio tradizionale allo sviluppo di applicazioni in cui la maggior parte del codice risiede nel DBMS
  - ⇨fortemente basato su stored procedures, viste e “trigger”

Stored Procedures

- Vantaggi
  - ⇨il DBMS ha il tempo di effettuare tutti i controlli e le ottimizzazioni sulle istruzioni
  - ⇨analisi sintattica
  - ⇨ottimizzazione delle interrogazioni
  - ⇨compilazione del codice
  - ⇨inoltre si riduce la comunicazione di client e server attraverso la rete
  - ⇨tipicamente ottime prestazioni

Stored Procedures

- Svantaggi
  - ⇨problemi di portabilità: i linguaggi sono normalmente incompatibili
  - ⇨è difficile riscrivere le stored procedures passando da un DBMS ad un altro
  - ⇨l'applicazione diventa dipendente dal DBMS
  - ⇨il DBMS scelto diventa nel tempo un "legame" tecnologico

In Questo Corso

- Le scelte tecnologiche
  - ⇨architettura applicativa di tipo “aperto” (logica applicativa separata dal DBMS)
  - ⇨tecnologia di tipo CLI
- Tecnologie di riferimento
  - ⇨JDBC
  - ⇨ADO.NET

Riassumendo

- Panoramica
- SQL Immerso (“Embedded SQL”)
  - ⇨Cursori
- Call Level Interface (CLI)
- Stored Procedures
- In Questo Corso

Termini della Licenza