
Parte I

Introduzione

Docente

Prof. Silvio Salza

Dipartimento di Informatica e Sistemistica
Via Ariosto 35, Il piano stanza B211
salza@dis.uniroma1.it (usare con criterio)
<http://www.dis.uniroma1.it/~salza/>

Ricevimento

verificare su web

Orario

Lunedì	14.00 -15.30	aula	21
Martedì	14.00 -15.30	aula	11

Materiale didattico

- Testo di riferimento
Atzeni, Ceri, Paraboschi, Torlone,
“Basi di dati: linguaggi, sistemi e architetture”,
Seconda edizione, McGraw-Hill, 1999.
oppure la versione abbreviata
Atzeni, Ceri, Paraboschi, Torlone,
“Basi di dati: modelli e linguaggi di interrogazione”,
McGraw-Hill, 2002.
- Copia delle trasparenze del docente, e altro materiale: sul sito web
<http://www.dis.uniroma1.it/~salza/>

Contenuti del corso

- *Modelli per la rappresentazione e la gestione efficiente di insiemi complessi ed estesi di dati*
- *Linguaggi e formalismi per la definizione, la manipolazione e l'interrogazione delle basi di dati*
- *Architettura dei Sistemi di Gestione di Basi di Dati (DBMS)*
- *Metodologie sistematiche per il progetto di basi di dati, sia dal punto di vista della completezza e correttezza della rappresentazione, che da quello dell'efficienza*
- *Amministrazione delle basi di dati*
- *Sviluppo di applicazioni su basi dati, basi di dati e Web*

Organizzazione del corso

1. Introduzione
 - Concetti fondamentali
 - Basi di dati e DBMS
 - Schemi ed istanze
 - Linguaggi di accesso
2. Il Modello relazionale
 - Relazioni, tabelle, attributi, domini
 - Vincoli di integrità
 - Chiavi e valori nulli
3. L'algebra relazionale
 - Operatori dell'algebra
 - L'algebra come l'ingaggio di interrogazione
 - Equivalenza delle espressioni

Organizzazione del corso

4. Il linguaggio SQL
 - Definizione dei dati
 - Interrogazioni e manipolazioni dei dati
 - Embedding nei linguaggi applicativi
5. Progettazione concettuale
 - Il modello Entità-Relazione
 - Costrutti del modello
 - Documentazione di schemi ER
 - La progettazione concettuale
6. Progettazione logica
 - Ristrutturazione degli schemi ER
 - Traduzione di schemi ER
 - Normalizzazione

Organizzazione del corso

7. Architettura del DBMS
 - Transazioni e gestione della concorrenza
 - Gestione del buffer
 - Controllo di affidabilità
8. Progettazione fisica
 - Strutture di accesso
 - Ottimizzazione delle interrogazioni
 - Tuning delle applicazioni
9. Basi di dati distribuite
 - Strutture di accesso
 - Ottimizzazione delle interrogazioni
 - Tuning delle applicazioni

Esercitazioni

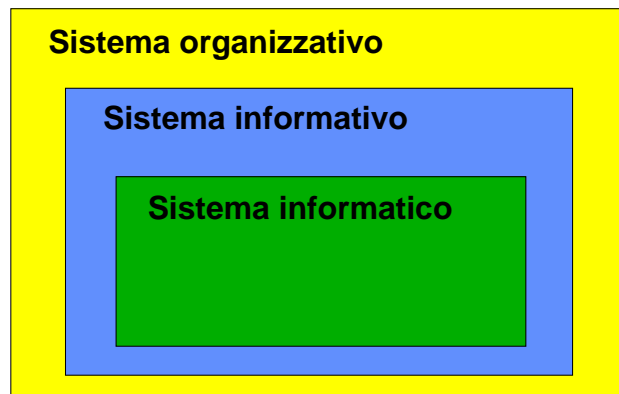
- Si terranno nell'orario di lezione, una volta alla settimana
- Faranno riferimento a DBMS relazionali: *SQL Server* e *Access* (possibile usare anche qualsiasi altro RDBMS)
- Progettazione di basi di dati tramite il modello ER
- Definizione e gestione di basi di dati relazionali
- Il linguaggio SQL
- Realizzazione di interfacce grafiche
- Accesso a basi dati tramite Web

Sistema informativo

- Dati ed informazioni sono una risorsa fondamentale di qualsiasi organizzazione
- Il sistema informativo è la parte di un'organizzazione relativa alla gestione dei dati e delle informazioni
- Funzioni del sistema informativo
 - Raccolta e acquisizione delle informazioni
 - Archiviazione e conservazione delle informazioni
 - Elaborazione delle informazioni
 - Distribuzione e scambio delle informazioni

Sistema informativo

- È la porzione automatizzata del sistema informativo
- Gestisce le informazioni per mezzo della tecnologia informatica



Dati e informazioni

- Nei sistemi informatici (e non solo), le **informazioni** vengono rappresentate in modo essenziale, spartano: attraverso i **dati**
 - **informazione**: notizia, dato o elemento che consente di avere conoscenza più o meno esatta di fatti, situazioni, modi di essere.
 - **dato**: ciò che è immediatamente presente alla conoscenza, prima di ogni elaborazione
- In informatica: i dati sono elementi di informazione costituiti da simboli che debbono essere elaborati

Informazioni e dati

- I caratteri:

Mario 275

scritti su un foglio di carta sono due *dati* e non significano molto

- Se il foglio di carta viene fornito in risposta alla domanda

“A chi mi devo rivolgere per il problema X; qual è il suo numero di telefono?”

allora i dati possono essere interpretati per fornire *informazione* e arricchire la conoscenza

Base di dati

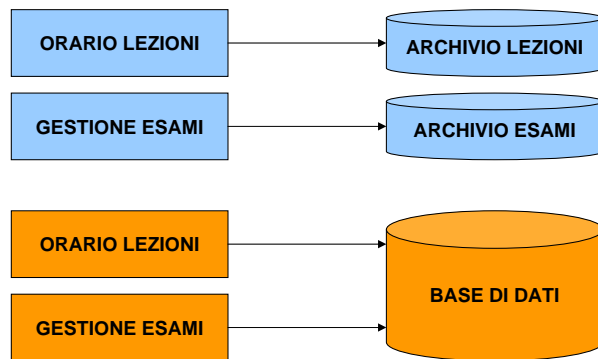
(accezione generica)

Collezione di dati, utilizzati per rappresentare le informazioni di interesse per una o più applicazioni di una organizzazione.

(accezione specifica)

Collezione di dati in memoria secondaria gestita da un apposito sistema software, chiamato DBMS (Sistema di Gestione di Basi di Dati).

Archivi e basi di dati

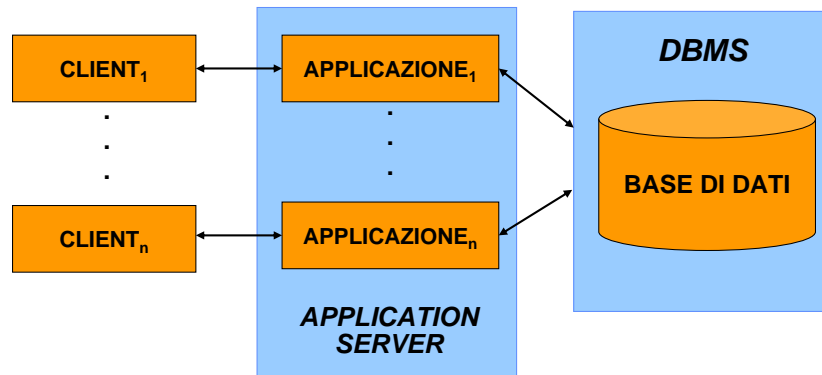


In una base di dati sono *integrati*, e *gestiti unitariamente* archivi diversi, condivisi da più applicazioni

DataBase Management System (DBMS)

- Sistema software in grado di gestire in modo efficiente, affidabile e sicuro *collezioni di dati* (basi di dati)
- Le basi di dati gestite da un DBMS devono poter essere:
 - *Grandi* : cioè di dimensioni molto maggiori della memoria centrale dei sistemi di calcolo utilizzati
 - *Persistenti* : con un periodo di vita indipendente dalle singole esecuzioni dei programmi che le utilizzano
 - *Condivise* : utilizzate da applicazioni diverse

Architettura delle applicazioni



Il DBMS è l'unico responsabile della gestione della base di dati: i dati sono accessibili all'esterno *solo* tramite il DBMS

Condivisione

- Gestione centralizzata della base di dati
- Condivisione dei dati da parte di applicazioni e utenti
L'integrazione e la condivisione permettono di ridurre la *ridondanza* (evitando ripetizioni)
- Evitato il verificarsi di fenomeni di inconsistenza
- La condivisione richiede un opportuno coordinamento degli accessi (*controllo della concorrenza*)
- Ripartiti su più applicazioni i costi di *progettazione* e di *mantenimento* della base di dati

Persistenza

- L'organizzazione dei dati ed il loro ciclo di vita sono indipendenti da quelli delle applicazioni
- I valori assunti dei dati sono preservati anche oltre il termine dell'esecuzione delle applicazioni
- Memorizzazione su dispositivi di memoria permanente (memoria secondaria), o di memoria stabile (RAID)
- La memorizzazione dei dati è garantita anche in presenza di guasti di dispositivo e di sistema

Caratteristiche del DBMS

Il DBMS è progettato in modo da garantire, al di là di quanto viene gestito a livello delle applicazioni:

- *affidabilità* : resistenza a guasti e malfunzionamenti hardware e software
- *sicurezza* : controllo degli accessi, e garanzia della riservatezza delle informazioni
- *efficienza* : capacità di utilizzare al meglio le risorse del sistema, e di ottimizzare i tempi di risposta
- *efficacia* : rendere produttive le attività dei suoi utilizzatori

Transazioni

- I DBMS supportano l'esecuzione transazioni
 - Insiemi di operazioni sulla base di dati effettuate da un'applicazione
 - Ciascuna transazione deve essere considerata come indivisibile (*atomica*)
 - La correttezza dell'esecuzione deve essere garantita anche in presenza di *concorrenza*
 - Gli effetti della transazione devono essere persistenti
- Il supporto delle transazioni da parte del DBMS semplifica ed alleggerisce lo sviluppo delle applicazioni

Transazioni: esempio

- Bonifico bancario dal conto A al conto B
 1. Prelievo dal conto A
 2. Aggiornamento della lista di operazioni di A
 3. Accredito sul conto B
 4. Aggiornamento della lista di operazioni di B
- Va evitata in ogni caso una esecuzione parziale delle operazioni che costituiscono la transazione
- Una volta che la transazione è stata confermata, la permanenza dei suoi effetti sulla base di dati deve essere garantita
- L'accesso concorrente ai conti A e B da parte di altre transazioni non deve pregiudicare la correttezza

DBMS e file system

- I file system offrono alcune funzioni proprie dei DBMS
 - Memorizzazione permanente
 - Controllo degli accessi
 - Strutture di accesso (indici)
- Particolari ambienti applicativi (*TP-monitor*) supportano l'esecuzione di transazioni
- Prima dell'avvento dei DBMS, le applicazioni sono state sviluppate direttamente sul file system
- I DBMS offrono enormi vantaggi in termini di costo di sviluppo e di gestione delle applicazioni

Perché i DBMS: coerenza dei dati

- I file system non prevedono verifiche sulla coerenza dei dati:
 - diversi file possono essere creati, cancellati e modificati indipendentemente l'uno dall'altro
 - questo può creare incoerenze rispetto alle informazioni che il contenuto dei file rappresentano
- Occorre sviluppare i relativi controlli a livello delle applicazioni: aumento dei costi di sviluppo
- Nei DBMS, sono presenti meccanismi che permettono di garantire che venga mantenuta la coerenza reciproca dei dati

Perché i DBMS: condivisione dei dati

- I file system prevedono forme di condivisione:
 - accesso condiviso in lettura
 - accesso esclusivo in scrittura
- Occorrono controlli ad hoc quando l'accesso coinvolge più file: aumento dei costi di sviluppo
- Occorre coordinare lo sviluppo e la manutenzione di applicazioni diverse che condividono gli stessi dati
- I DBMS integrano e centralizzano la gestione dei dati, assicurando i controlli in maniera trasparente per le applicazioni

Perché i DBMS: catalogo dei dati

- Nei programmi tradizionali che accedono a file:
 - ogni programma contiene una descrizione della struttura del file stesso
 - nascono rischi di incoerenza fra le descrizioni (ripetute in ciascun programma) e i file stessi
- I DBMS gestiscono un catalogo, cioè una descrizione centralizzata dei dati
 - il catalogo è utilizzato da tutti i programmi che condividono i dati
 - La gestione del catalogo è centralizzata ed integrata

Perché i DBMS: indipendenza dei dati

- I DBMS consentono di utilizzare diversi livelli di descrizione e rappresentazione dei dati
- Organizzazione logica della base di dati (livello alto) : è quella cui i programmi fanno riferimento
- Organizzazione fisica dei dati (livello basso) : può essere modificata senza necessità di modifica dei programmi
- In questo modo il DBMS garantisce l'indipendenza dei dati dalla rappresentazione fisica
- Gli sviluppatori delle applicazioni vedono solo l'organizzazione logica

Perché i DBMS: interrogazione dei dati

- Nei programmi tradizionali che accedono a file ogni programma deve realizzare:
 - i meccanismi di accesso ai dati nei file
 - i metodi di composizione di tali dati per ottenere i risultati richiesti
- I DBMS mettono a disposizione potenti linguaggi di interrogazione, che:
 - permettono di accedere in modo semplice e flessibile ai dati memorizzati
 - permettono di comporre dati elementari per effettuare la valutazione di interrogazioni complesse

Modello dei dati

Insieme di costrutti utilizzati per organizzare i dati di interesse e descriverne la dinamica

- Componente fondamentale: meccanismi di strutturazione (o costruttori di tipo)
- Come nei linguaggi di programmazione esistono meccanismi che permettono di definire nuovi tipi, così ogni modello dei dati prevede alcuni costruttori
- Esempio: il modello relazionale prevede il costruttore relazione, che permette di definire insiemi di record omogenei

Tabelle

Orario

Insegnamento	Docente	Aula	Ora
Analisi I	Luigi Neri	N1	8:00
Basi di dati	Piero Rossi	N2	9:45
Chimica	Nicola Mori	N1	9:45
Fisica I	Mario Bruni	N1	11:45
Fisica II	Mario Bruni	N3	9:45
Sistemi inform.	Piero Rossi	N3	8:00

Aule

Nome	Edificio	Piano
N1	A	1
N2	B	1
N3	A	2

- Nel modello relazionale l'organizzazione dei dati è di tipo tabellare
- Una tabella rappresenta *un insieme omogeneo di record*

Schemi ed istanze

In ogni base di dati distinguiamo due aspetti

- Lo schema (*aspetto intensionale*)
 - descrive la struttura della base di dati, cioè l'organizzazione dei dati
 - è sostanzialmente invariante nel tempo
- L'istanza (*aspetto estensionale*)
 - è costituita dai valori attuali, contenuti nella base di dati
 - possono cambiare molto e rapidamente
- Allo stesso schema corrispondono più istanze

Schemi ed istanze: esempio

- In una base di dati relazionale lo schema è costituito dalle intestazioni delle tabelle (più altro):

Orario	Insegnamento	Docente	Aula	Ora
Aule	Nome	Edificio	Piano	

- L'istanza è costituita dal corpo di ciascuna tabella, ad un certo istante; ad esempio per la tabella **Aula**

N1	A	1
N2	B	1
N3	A	2

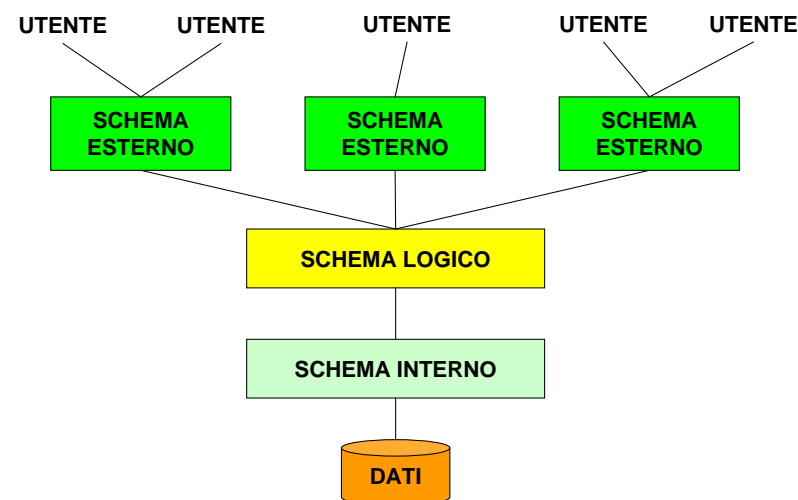
Modelli logici e concettuali

- **Modelli logici:** utilizzati nei DBMS esistenti per l'organizzazione dei dati; ad essi fanno riferimento i programmi; sono indipendenti dalle strutture fisiche
- **Modelli concettuali:** permettono di rappresentare i dati in modo indipendente da ogni sistema, cercando di descrivere i concetti del mondo reale; sono utilizzati nelle fasi preliminari di progettazione

Esempi: relazionale, reticolare, gerarchico, a oggetti

Esempi : il più noto è il modello Entità-Relazione

Architettura a tre livelli (ANSI/SPARCS)



Architettura ANSI/SPARCS: schemi

- **Schema logico:** descrizione dell'intera base di dati nel modello logico adottato dal DBMS
- **Schema esterno:** descrizione di una porzione della base di dati di interesse in un modello logico ("viste" parziali, derivate, anche in modelli diversi)
- **Schema interno (o fisico):** rappresentazione dello schema logico per mezzo di strutture fisiche di memorizzazione

Esempio: schema logico

Corsi			Aule		
Corso	Docente	Aula	Nome	Edificio	Piano
Analisi	Rossi	DS3	DS1	OMI	Terra
Sistemi	Neri	N3	N3	OMI	Terra
Reti	Bruni	N3	G	Pincherle	Primo
Controlli	Bruni	G			

CorsiSedi	Corso	Aula	Edificio	Piano
	Sistemi	N3	OMI	Terra
	Reti	N3	OMI	Terra
	Controlli	G	Pincherle	Primo

Rappresenta l'organizzazione dell'intera base di dati, in questo caso nel modello relazionale

Esempio: schema esterno

Corsi-a

Corso	Aula
Analisi	DS3
Sistemi	N3
Reti	N3
Controlli	G

Aule

Nome	Edificio
DS1	OMI
N3	OMI
G	Pincherle

- Rappresenta la parte della base di dati visibile ad una certa applicazione, od insieme di utenti
- La visibilità è limitata ad un sottoinsieme di tabelle, ed a parte dello schema di alcune tabelle
- Diverse applicazioni hanno diversi livelli di visibilità

Indipendenza dei dati

- Conseguenza diretta della articolazione in livelli
- L'accesso avviene solo tramite il livello esterno (che può coincidere con il livello logico)
- Si estrinseca a due livelli:
 - *Indipendenza fisica*: il livello esterno e quello logico sono disaccoppiati da quello fisico
 - *Indipendenza logica*: il livello esterno è disaccoppiato da quello logico
- *Trasparenza*: modifiche apportate ad un livello non necessariamente interessano gli altri

Indipendenza fisica

- Il livello logico e quello esterno sono indipendenti da quello fisico
- Una vista (schema esterno) e una relazione (schema logico) non sono interessate da modifiche all'organizzazione fisica della base di dati
- Modifiche all'organizzazione fisica della base di dati possono essere necessarie per motivi di efficienza
- L'organizzazione fisica della base di dati può cambiare senza che debbano essere modificati i programmi
- Questo semplifica enormemente la manutenzione delle applicazioni

Indipendenza logica

- Il livello esterno è indipendente da quello logico
- Aggiunte o modifiche alle viste non richiedono modifiche al livello logico
- Modifiche allo schema logico che lascino inalterato lo schema esterno sono trasparenti
- Questo semplifica lo sviluppo di nuove applicazioni, e rende largamente indipendente la manutenzione delle diverse applicazioni
- Le modifiche al livello logico e fisico possono essere apportate dagli amministratori della base di dati, e non dai responsabili delle applicazioni

Accesso alle basi di dati

- Le applicazioni, e gli amministratori, con diversi privilegi, accedono alla base di dati, in tre diverse modalità
- Definizione: per modificare lo schema, esterno, logico, o fisico della base di dati
- Interrogazione: per effettuare ricerche il cui esito dipende dai valori contenuti nell'istanza corrente
- Manipolazione: per modificare il contenuto della base di dati, e cioè per modificare l'istanza corrente

Data Definition e Data Manipulation

- Tutti i linguaggi per accesso a basi di dati fanno una chiara distinzione tra il livello intensionale ed il livello estensionale
- Data Definition Language (DDL)
per la definizione di schemi (logici, esterni, fisici) e altre operazioni generali
- Data Manipulation Language (DML)
per l'interrogazione e l'aggiornamento di (istanze di) basi di dati

Linguaggi per basi di dati

L'accesso ai dati può avvenire con diverse modalità:

- Tramite linguaggi testuali interattivi (ad es. SQL)
- Tramite comandi (come quelli del linguaggio interattivo) immersi in un linguaggio ospite (Java, C, Cobol, etc.)
- Tramite comandi (come quelli del linguaggio interattivo) immersi in un linguaggio ad hoc, con anche altre funzionalità (ad es. per grafici o stampe strutturate), anche con l'ausilio di strumenti di sviluppo (ad es. per la gestione di maschere)
- Tramite interfacce amichevoli, tipicamente grafiche (senza ricorso a linguaggi testuali)

Esempio: creazione di una tabella

ESPRESSIONE SQL

```
CREATE TABLE Orario (  
    Insegnamento CHAR(20) ,  
    Docente CHAR(20) ,  
    Aula CHAR(4) ,  
    Ora CHAR(5) );
```

SCHEMA DELLA TABELLA

Orario	Insegnamento	Docente	Aula	Ora
--------	--------------	---------	------	-----

Esempio: interrogazione

BASE DI DATI

Corsi

Corso	Docente	Aula
Analisi	Rossi	DS3
Sistemi	Neri	N3
Reti	Bruni	N3
Controlli	Bruni	G

Aule

Nome	Edificio	Piano
DS1	OMI	Terra
N3	OMI	Terra
G	Pincherle	Primo

INTERROGAZIONE (in linguaggio naturale)

Trovare i corsi tenuti in aule al piano terra

Esempio: interrogazione in SQL

INTERROGAZIONE IN SQL

```
SELECT Corso, Aula, Piano
FROM Aule, Corsi
WHERE Nome = Aula
AND Piano = "Terra"
```

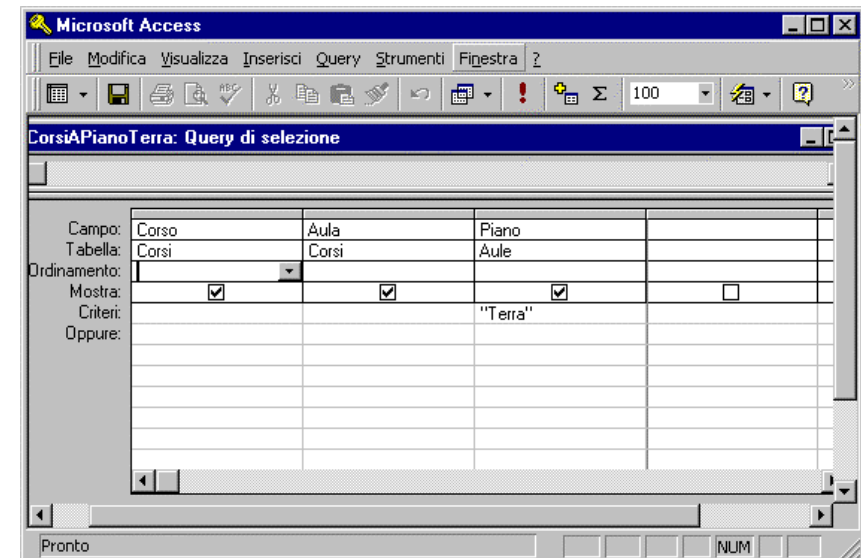
RISULTATO

Corso	Aula	Piano
Sistemi	N3	Terra
Reti	N3	Terra

Esempio: linguaggio ospite

```
write('nome della città?'); readln(città);
EXEC SQL DECLARE P CURSOR FOR
  SELECT NOME, REDDITO
  FROM PERSONE
  WHERE CITTA = :città ;
EXEC SQL OPEN P ;
EXEC SQL FETCH P INTO :nome, :reddito ;
while SQLCODE = 0 do begin
  write('nome della persona:', nome, 'aumento?');
  readln(aumento);
  EXEC SQL UPDATE PERSONE
    SET REDDITO = REDDITO + :aumento
    WHERE CURRENT OF P
  EXEC SQL FETCH P INTO :nome, :reddito
end;
EXEC SQL CLOSE CURSOR P
```

Esempio: interazione non testuale



Personaggi e interpreti

- Progettisti e realizzatori di DBMS
- Progettisti della base di dati
- Amministratori della base di dati (DBA, Data Base Administrator)
- Progettisti di applicazioni
- Utenti
 - Utenti finali (terminalisti): eseguono applicazioni predefinite (transazioni)
 - Utenti casuali: eseguono operazioni non previste a priori, usando linguaggi interattivi

Il DataBase Administrator (DBA)

- Persona o gruppo di persone responsabile del controllo centralizzato e della gestione del sistema:
 - Prestazioni
 - Affidabilità
 - Sicurezza
- Le funzioni del DBA includono quelle di progettazione, anche se in progetti complessi ci possono essere distinzioni
- I progettisti delle applicazioni, avendo privilegi limitati devono interagire con il DBA

Vantaggi dei DBMS

- Dati come risorsa comune, schema dei dati come modello della realtà
- Gestione centralizzata con possibilità di standardizzazione ed “economia di scala”
- Disponibilità di servizi integrati
- Riduzione di ridondanze e incoerenze
- Indipendenza dei dati (favorisce lo sviluppo e la manutenzione delle applicazioni)
- Portabilità delle applicazioni

Svantaggi dei DBMS

- Costo dei prodotti e della transizione verso di essi
- Non scorporabilità delle funzionalità (con riduzione di efficienza)
- Minore controllo sull'efficienza delle applicazioni:
 - Il DBMS ‘nasconde’ il livello fisico
 - L'analisi delle prestazioni è più complessa
 - La capacità di intervento è più limitata
- La trasparenza, oltre che un vantaggio può essere, in certe situazioni, uno svantaggio