Basi di Dati
Introduzione alle basi di dati e DBMS

Slide adattate da Atzeni, Ceri, Fraternali, Paraboschi, Torlone, Basi di dati, 5/ed, McGraw-Hill Education, 2018

Basi di dati

- Insieme organizzato di dati utilizzati per il supporto allo svolgimento di attività (di enti, aziende, uffici, persone, …)
- Di solito un DB modella un'organizzazione reale (impresa, università, …)
- Cambiamenti nell'organizzazione → cambiamenti nel DB
- Esempi: dati del personale, dati bancari, prenotazioni voli, …

Motivazione per i Database

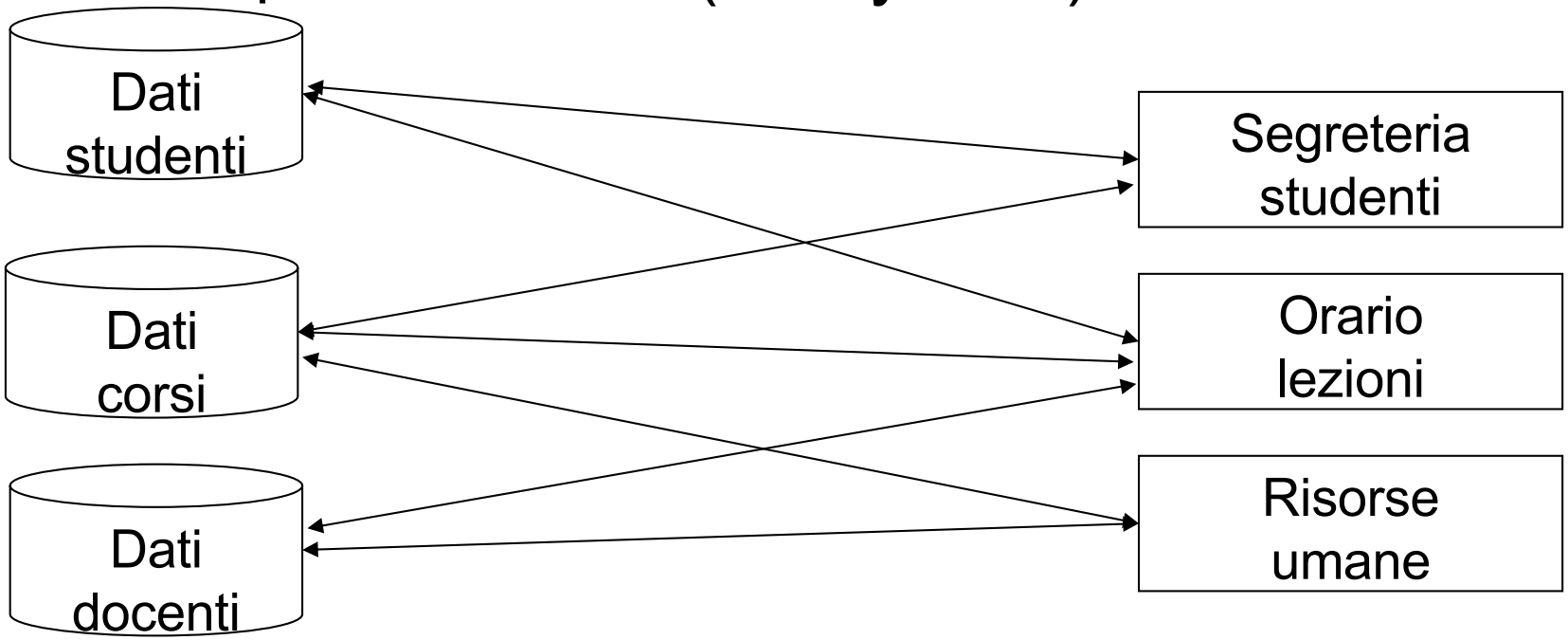
- Un programma tipicamente è composto da codice+dati
- Per es. ordinare 1000 numeri
  - 2, 101, 42, 63, 99, 1, …
  - Memorizzare questi numeri in un array
  - Scrivere codice per ordinarli
- Codice e dati sono immagazzinati in memoria principale
- Ma...

Motivazione per i Database

- Ma...
- I dati possono essere enormi (gestire non 1000 numeri, ma miliardi di numeri)
- Cosa succede se i dati non stanno in memoria principale?
- Inoltre altre applicazioni potrebbero volere accedere agli stessi dati
- Duplicare codice di gestione dati per ogni applicazione?

Database vs File system

- Esempio università (file system)



Si vuole che queste applicazioni :

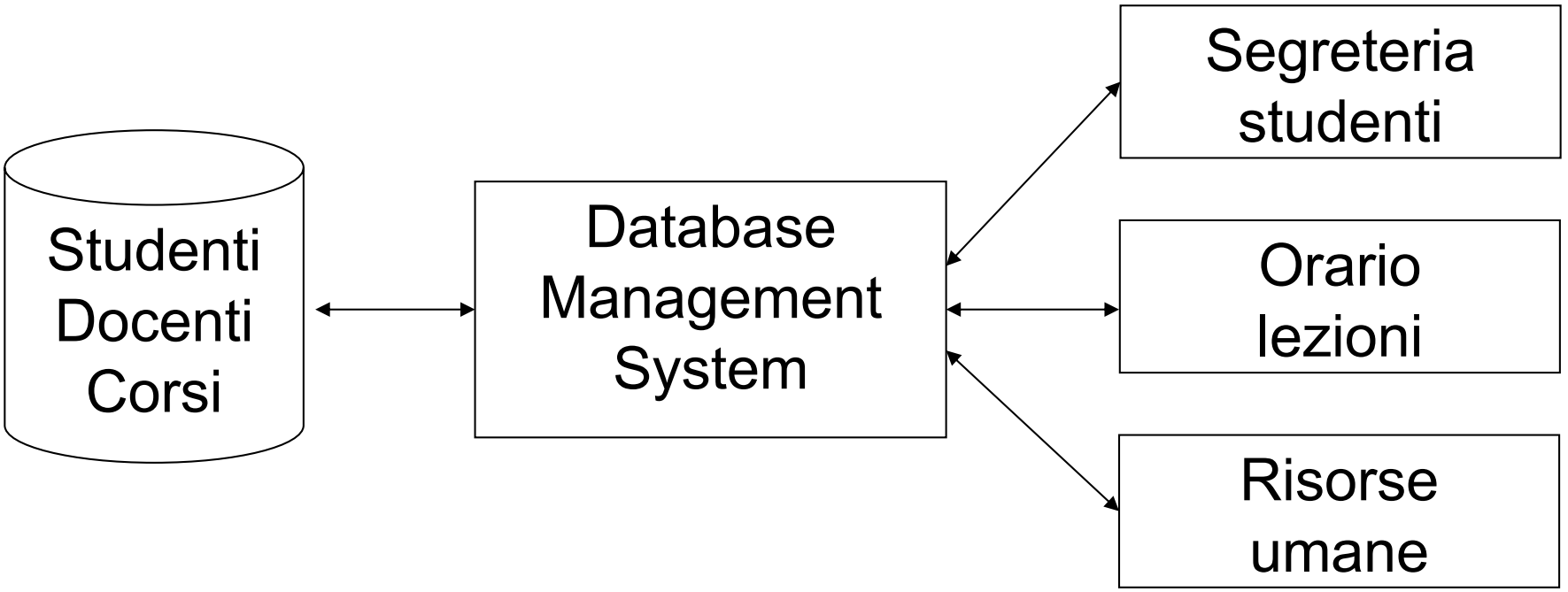
- gestiscano grandi quantità di dati (centinaia di gigabyte)
- offrano protezione da malfunzionamenti
- impediscano accessi non autorizzati
- permettano di interrogare, modificare i dati
- permettano a centinaia/migliaia di utenti di accedere simultaneamente ai dati

Approccio File System

- Il programmatore ha un alto carico di lavoro (definire strutture dati efficienti, prevedere modalità di accesso, …). Es.:
  - Come si memorizzano centinaia di gigabyte di dati in modo efficiente?
  - Deve scrivere nuovo codice per ogni interrogazione garantendo performance?
  - Come ci si protegge da crash di sistema durante la modifica dei dati?
  - Come si evita interferenza tra utenti diversi?
- Ridondanza non controllata
- Possibile incoerenza dei dati
- Mancanza flessibilità
- Dati non condivisibili tra applicazioni diverse
- Grande carico di lavoro per la manutenzione
- Poca robustezza
- Meglio demandare a un DBMS la gestione di questi problemi…

Database vs File system

- Esempio università (database)



Approccio Database

- Gestione dati tramite database
  - Il programmatore ha un carico di lavoro più contenuto
  - Ridondanza controllata (coerenza dei dati e vincoli di integrità)
  - Integrazione dei dati (autocontenuti, rappresentano la semantica dell'applicazione)
  - Flessibilità (dati indipendenti e accessibili, le modalità di accesso e interrogazione non devono essere definite a priori)
  - Grande carico di lavoro per la manutenzione
  - Servizi di sicurezza, backup, gestione condivisione forniti automaticamente

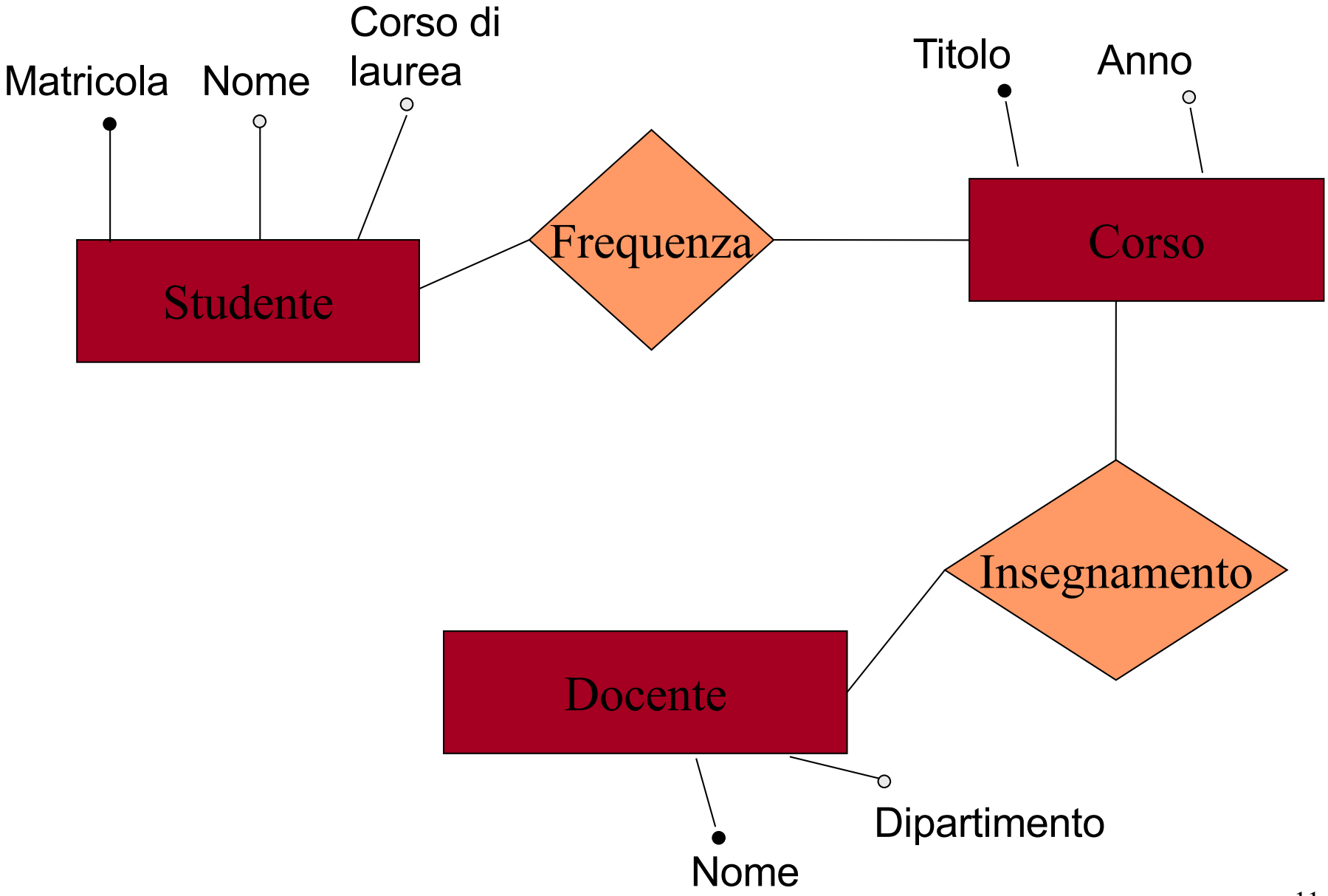
Svantaggi database

- Se un'applicazione è semplice, è usata da un solo utente, è statica e gestisce pochi dati, i file possono essere vantaggiosi
- I DBMS sono «costosi», complessi, potenti

Sviluppare un'applicazione che usa un DBMS

1. **Progettazione**
   - Partendo dai requisiti decidere quali entità includere e come sono collegate
   - Definire la struttura del DB e i tipi di dati
2. **Implementazione**
   - Creare le strutture sul DBMS
   - Popolare il DB
3. **Scrivere applicazioni che usano il DBMS**
   - Manipolazione dei dati: inserimento, interrogazione, aggiornamento, cancellazione dei dati (CRUD: Create, Read, Update, Delete)
   - Molto più facile ora perché il DBMS si occupa della gestione dei dati

Schema concettuale



Schema Logico

- Tabelle:

Studenti:

Matricola	Nome	CorsoDiLaurea
456789	Chiara	Informatica
567890	Daniele	Biologia
	…	…

Frequenza:

Matricola	Corso
456789	MFN444
456789	MFN541
567890	MFN444
	…

Corsi:

Codice	Titolo	Anno
MFN444	Database	2
MFN541	Algoritmi	2

- Separa la vista logica dei dati dalla rappresentazione fisica

Interrogare un Database

- “Trova tutti i corsi frequentati da Chiara”
- S(tructured) Q(uery) L(anguage)

```
select  C.nome
from    Studenti S, Frequenza F, Corsi C
where   S.nome = 'Chiara' and
        S.matricola = F.matricola and
        F.corso = C.codice
```

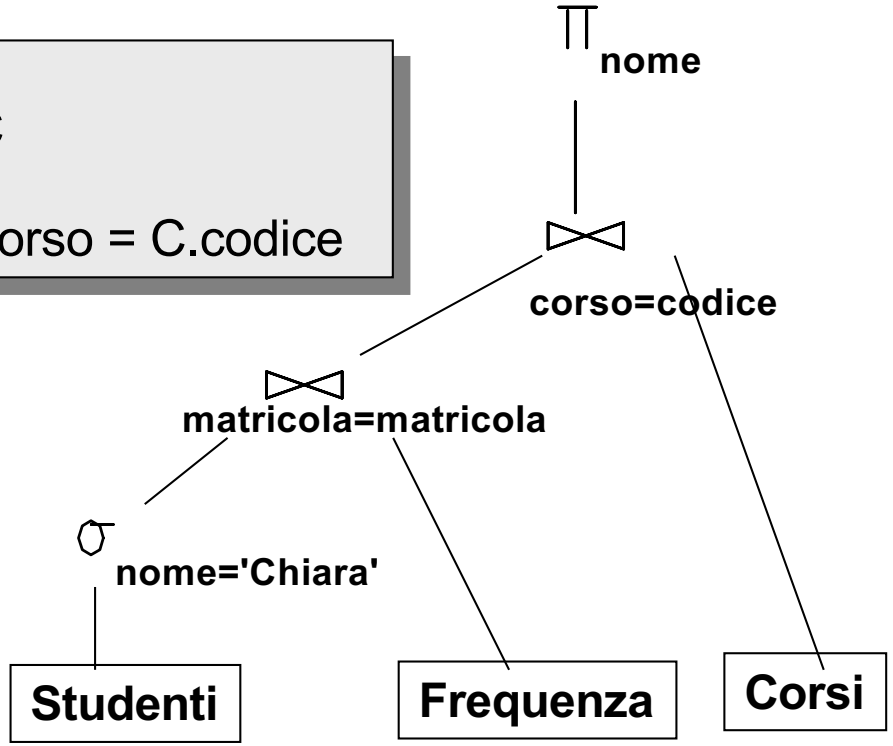
- Il DBMS elabora un piano per rispondere alla query in modo efficiente.

Ottimizzazione Query

Obiettivo:

Query SQL dichiarativa ⟶ ***Piano di esecuzione imperativo:***

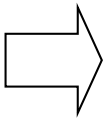
```
select  C.nome
from    Studenti S, Frequenza F, Corsi C
where   S.nome = 'Chiara' and
        S.matricola = F.matricola and F.corso = C.codice
```



Piano: albero di operatori dell'algebra relazionale e scelta dell'algoritmo per eseguire ogni operatore

Sistema informativo

- Componente di una organizzazione che gestisce le informazioni di interesse (cioè utilizzate per il perseguimento degli scopi dell'organizzazione)
- Ogni organizzazione ha un sistema informativo, eventualmente non esplicitato nella struttura
- Il sistema informativo è di supporto ad altri sottosistemi, e va quindi studiato nel contesto in cui è inserito



Sistemi informativi e automazione

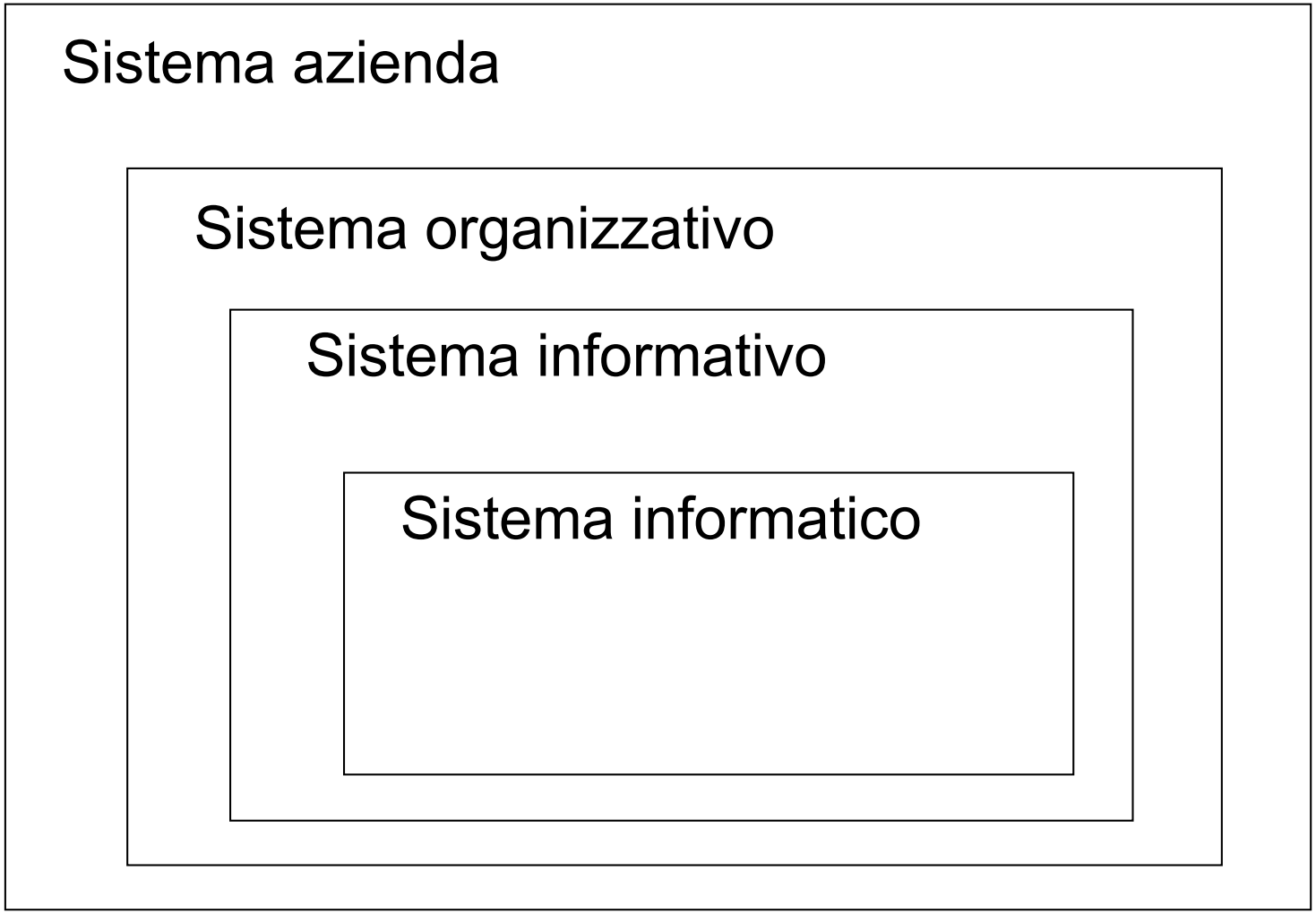
- Il concetto di “sistema informativo” è indipendente da qualsiasi automatizzazione:
  - esistono organizzazioni la cui ragion d’essere è la gestione di informazioni (per es. servizi anagrafici e banche) e che operano da secoli

Sistema Informatico

- Porzione automatizzata del sistema informativo:

  la parte del sistema informativo che gestisce informazioni con tecnologia informatica

Sistema Informatico



Database relazionali

- I database relazionali, fin dagli anni ‘70, hanno avuto un enorme successo
- Nuovi tipi sono emersi in seguito (ad es. NoSQL)
- Ma i database relazionali rimangono fondazionali e pervasivi e probabilmente rimarranno tali in futuro
- In questo corso ci focalizziamo sui database relazionali
- I concetti che introdurremo sono comunque generali e validi nell’ambito della gestione dei dati, che sta diventando sempre più importante
- Nela maggior parte degli impieghi per un informatico è richiesta conoscenza di database relazionali

DBMS

- Principali Data Base Management Systems relazionali:
  - *Oracle DB* (1979) proprietario, molto diffuso commercialmente, potente,
  - *PostgreSQL* (1989) open source, potente, grande aderenza agli standard,
  - *MySQL* (e *MariaDB*) (1995) inizialmente libero poi acquisito da Oracle, molto diffuso nelle applicazioni web, ha alcune limitazioni e non è particolarmente aderente agli standard; MariaDB è una versione open source,
  - *Microsoft SQL Server* (1989) proprietario, limitato supporto a SO diversi da Windows
  - *Microsoft Access* (1992) proprietario, limitato, utile per uso personale, integra ambiente di sviluppo grafico, solo su Windows
  - *SQLite* (2000), open source, contenuto in una libreria C, non è client/server, molto diffuso come DBMS embedded nelle applicazioni

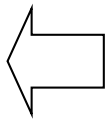
Sistema di gestione di basi di dati DataBase Management System (DBMS)

- Sistema che gestisce collezioni di dati:
  - grandi
  - persistenti
  - condivise

  garantendo
  - privatezza
  - affidabilità
  - efficienza
  - efficacia

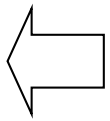
Le basi di dati sono... grandi

- Dimensioni (molto) maggiori della memoria centrale dei sistemi di calcolo utilizzati
- Il limite deve essere solo quello fisico dei dispositivi di memoria secondaria
- Esempi di dimensioni molto grandi
  - 500 Gigabyte (dati transazionali)
  - 10 Terabyte (dati decisionali)
  - 500 Terabyte (dati scientifici)
  - 100 miliardi di record



Le basi di dati sono... persistenti

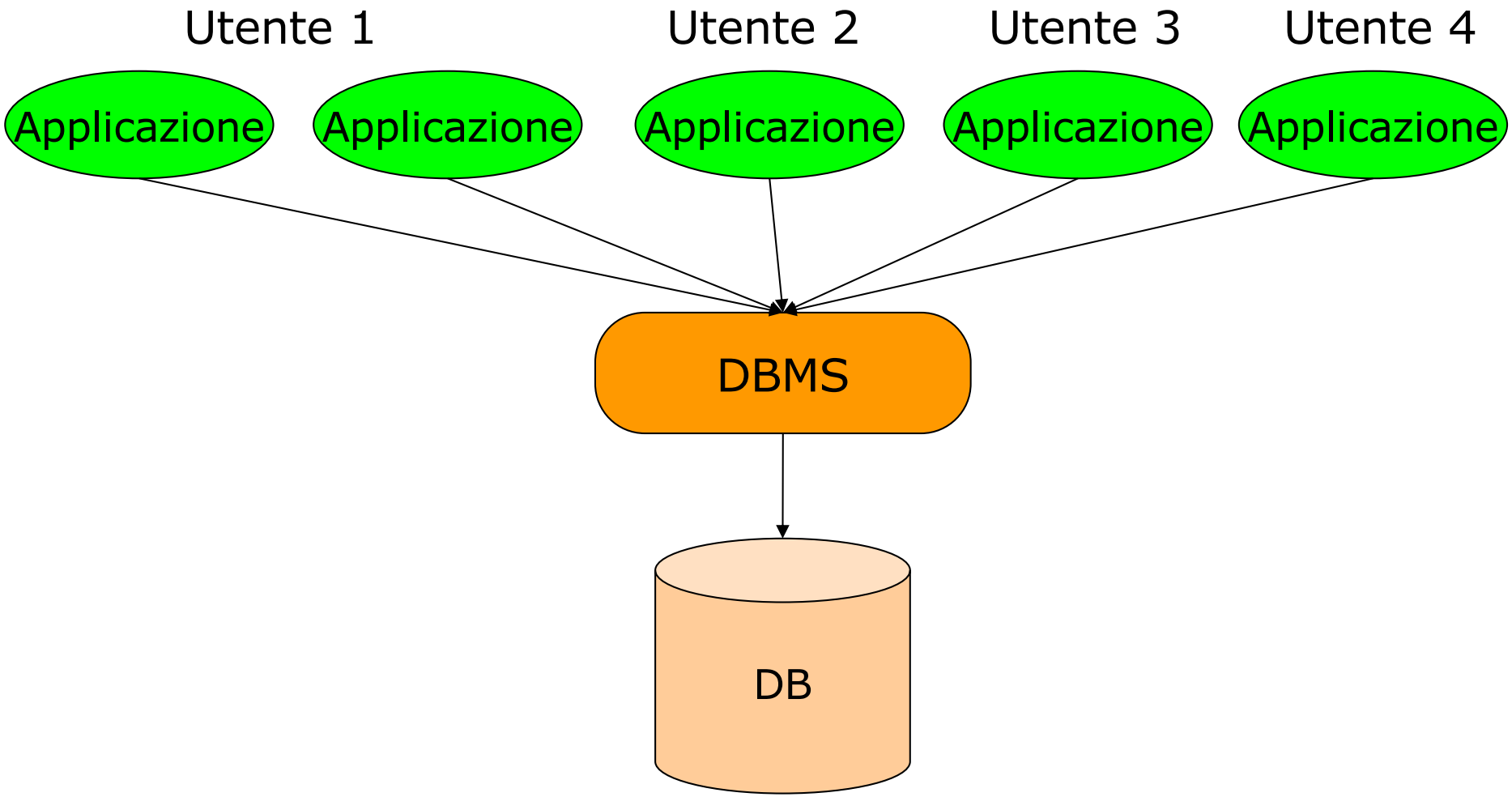
- I dati hanno un tempo di vita che non è limitato a quello delle singole esecuzioni delle applicazioni



Le basi di dati sono... condivise

- Ogni organizzazione (specie se grande) è divisa in settori o comunque svolge diverse attività
- Ciascun settore/attività ha un (sotto)sistema informativo (non necessariamente disgiunto)

Le basi di dati sono... condivise



Le basi di dati sono... condivise

La condivisione permette di evitare *ridondanza* e *incoerenza*

- *Ridondanza*: informazioni ripetute
- *Incoerenza*: errori di “allineamento” dei dati
  se i dati fossero ripetuti, sarebbe necessario mantenere “allineate” le varie copie

UNIVERSITA' DEGLI STUDI DI CHISSADOVE

Corso di Studi in Ingegneria Informatica

ORARIO DELLE LEZIONI PER L'ANNO ACCADEMICO 1999-2000

INSEGNAMENTO	Docente	Aula	Orario
Analisi matematica I	Luigi Neri	N1	8:00-9:30
Basi di dati	Piero Rossi	N2	9:45-11:15
Chimica	Nicola Mori	N1	9:45-11:30
Fisica I	Mario Bruni	N1	11:45-13:00
Fisica II	Mario Bruni	N3	9:45-11:15
Sistemi informativi	Piero Rossi	N3	8:00-9:30

Esempio applicativo di gestione dell'orario delle lezioni

UNIVERSITA' DEGLI STUDI CHISSADOVE

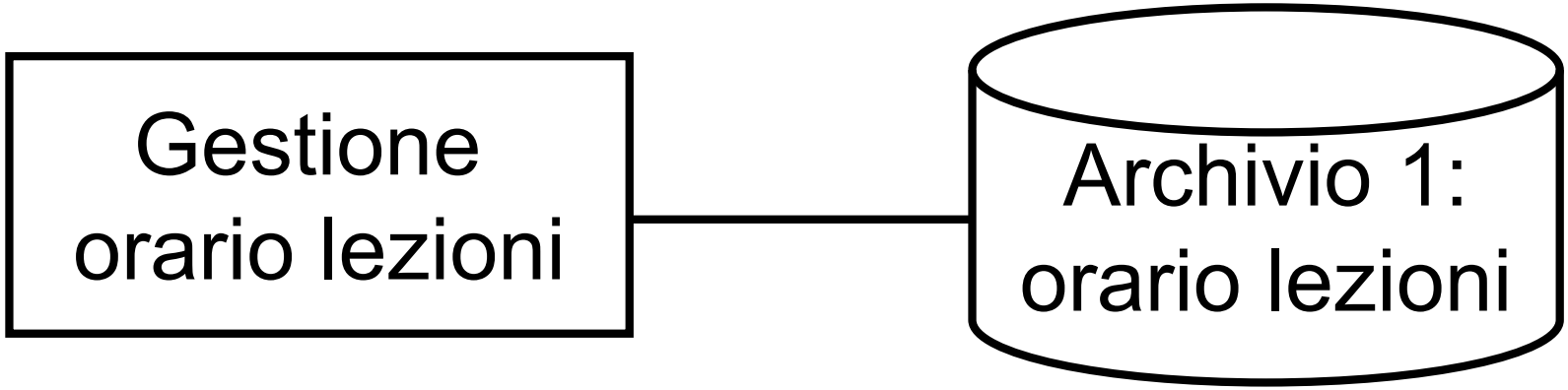
Corso di Studi in Ingegneria Informatica

Orario di ricevimento dei docenti

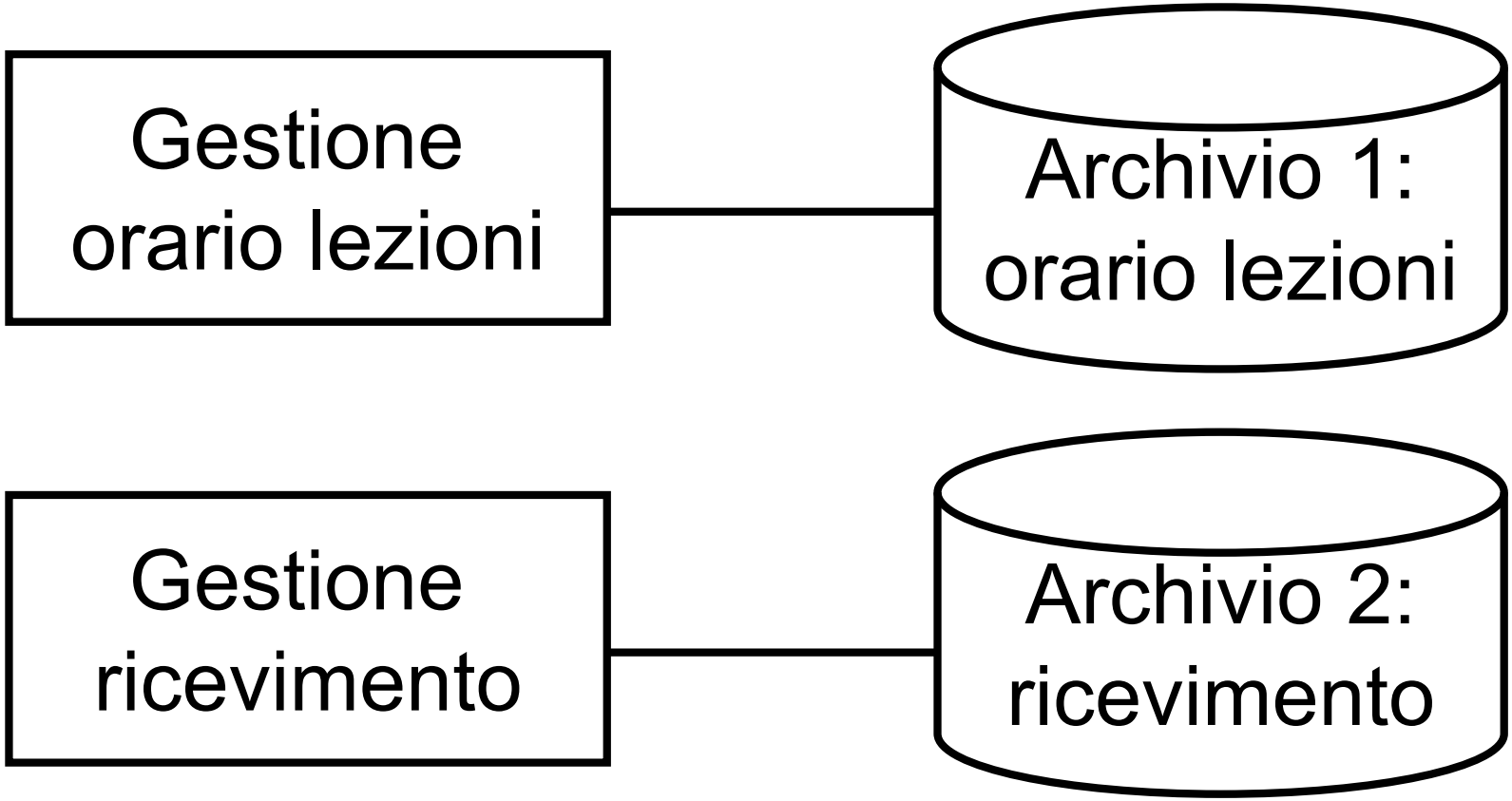
DOCENTE	INSEGNAMENTI	ORARIO
Mario BRUNI	Fisica I Fisica II	Martedi' 10-12
Luigi NERI	Analisi matematica I	Lunedi' 12-13
Piero ROSSI	Basi di dati Sistemi informativi	Giovedi' 11-13
Nicola MORI	Chimica	Martedi' 16-18

Esempio applicativo di gestione del ricevimento studenti

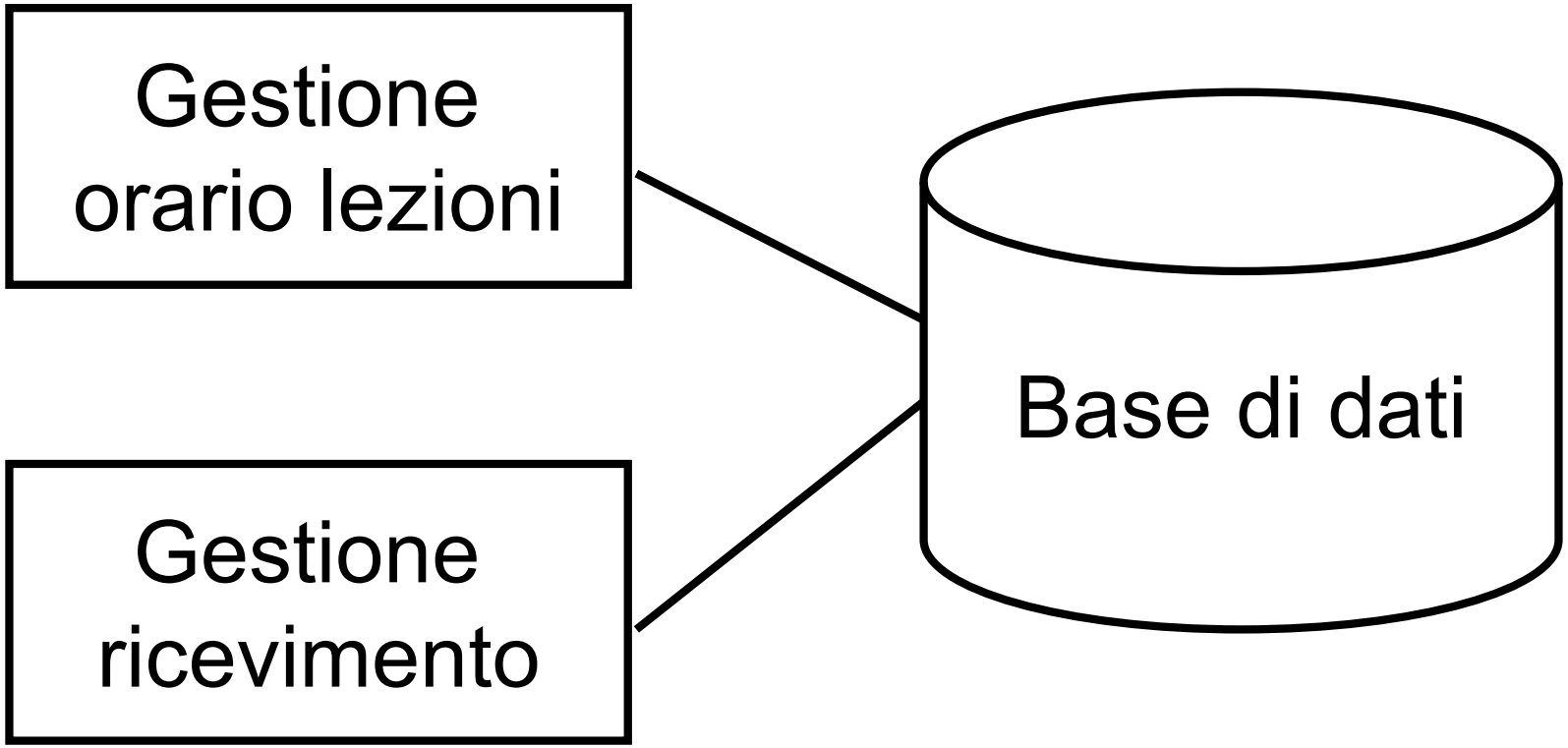
Archivi e basi di dati



Archivi e basi di dati



Archivi e basi di dati



Le basi di dati sono… condivise

- Una base di dati è una risorsa integrata, condivisa fra applicazioni
- Questo permette di evitare ridondanze e incoerenze

Le basi di dati sono… condivise

Condivisione → Concorrenza

Concorrenza: nello stesso momento più applicazioni possono accedere al medesimo dato; tali accessi non devono interferire tra loro per garantire l'integrità dei dati

I DBMS forniscono meccanismi per gestire la concorrenza e regolamentare gli accessi

I DBMS garantiscono... privatezza

- Si possono definire meccanismi di **autorizzazione**
  - l'utente A è autorizzato a leggere tutti i dati e a modificare X
  - l'utente B è autorizzato a leggere dati X e a modificare Y
- Es. **biblioteca**:
  - il *lettore* ha diritto di lettura e ricerca dei dati, ma non di modifica/inserimento
  - il *bibliotecario* ha diritto di modificare i dati: aggiunge/dismette libri e segna i prestiti

I DBMS garantiscono... affidabilità

- Affidabilità (per le basi di dati):
  - resistenza a malfunzionamenti hardware e software
- Una base di dati è una risorsa pregiata e quindi deve essere conservata a lungo termine
- Tecnica fondamentale:
  - gestione delle transazioni

Transazione

- Sequenza di operazioni da considerare indivisibile ("atomico"), corretta anche in presenza di concorrenza e con effetti definitivi

Le transazioni sono… atomiche

- Una sequenza di operazioni correlate…
  - trasferimento di fondi da un conto A a un conto B: sono due operazioni, il prelevamento da A e il versamento su B
- … deve essere eseguita per intero o per niente:
  - o si esegue sia il prelevamento da A che il versamento su B oppure nessuno dei due

Le transazioni sono… concorrenti

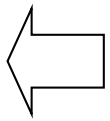
- L'effetto di transazioni concorrenti deve essere coerente
  - se due assegni emessi sullo stesso conto corrente vengono incassati contemporaneamente

    … si deve evitare di trascurarne uno
  - se due persone richiedono lo stesso posto (libero) su un treno o un aereo

    … si deve evitare di assegnarlo a entrambe

I risultati delle transazioni sono permanenti

- La conclusione positiva di una transazione corrisponde a un impegno (in inglese commit) a mantenere traccia del risultato in modo definitivo, anche in presenza di guasti e di esecuzione concorrente



I DBMS devono essere... efficienti

- Cercano di utilizzare al meglio le risorse di spazio di memoria (principale e secondaria) e tempo (di esecuzione e di risposta)
- I DBMS, nonostante offrano molte funzioni su grandi quantità di dati, devono svolgere le operazioni in tempi accettabili.
- Grandi investimenti e competizione
- L'efficienza è anche il risultato della qualità delle applicazioni che si interfacciano con il DBMS

I DBMS debbono essere... efficaci

- Cercano di rendere produttive le attività dei loro utilizzatori, offrendo funzionalità articolate, potenti e flessibili

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

UNIVERSITA' DEGLI STUDI DI CHISSADOVE

Corso di Studi in Ingegneria Informatica

ORARIO DELLE LEZIONI PER L'ANNO ACCADEMICO 1999-2000

INSEGNAMENTO	Docente	Aula	Orario
Analisi matematica I	Luigi Neri	N1	8:00-9:30
Basi di dati	Piero Rossi	N2	9:45-11:15
Chimica	Nicola Mori	N1	9:45-11:30
Fisica I	Mario Bruni	N1	11:45-13:00
Fisica II	Mario Bruni	N3	9:45-11:15
Sistemi informativi	Piero Rossi	N3	8:00-9:30

Esempio dell’orario delle lezioni

Organizzazione dei dati in una base di dati

Orario

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

Basi di dati: schema e istanza

Lo schema della base di dati

Orario

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

L'istanza della base di dati

Schema e istanza

- In ogni base di dati esistono:
  - lo schema, sostanzialmente invariante nel tempo, che ne descrive la struttura (aspetto intensionale)
    - es.: le intestazioni delle tabelle
  - l'istanza, i valori attuali, che possono cambiare anche molto rapidamente (aspetto estensionale)
    - es.: il "corpo" di ciascuna tabella

Due tipi (principali) di modelli

- modelli concettuali
- modelli logici

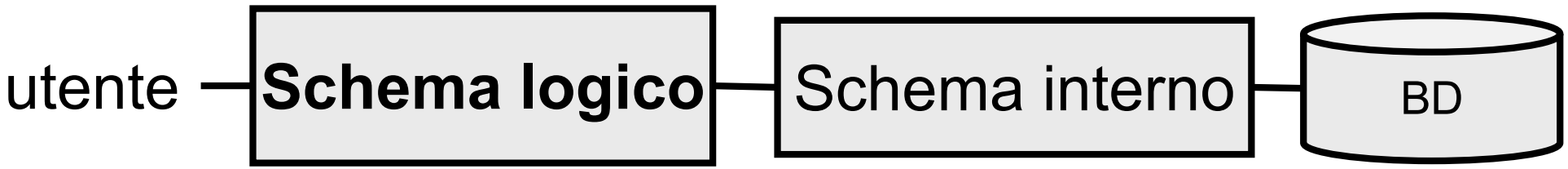
Modelli concettuali

- Non sono disponibili nei DBMS commerciali
- Permettono di rappresentare i dati in modo indipendente da ogni sistema
  - cercano di descrivere i concetti del mondo reale
  - sono utilizzati nelle fasi preliminari di progettazione
- Il più diffuso è il modello Entity-Relationship (Entità-Associazione)

Modelli logici

- Adottati nei DBMS esistenti per l'organizzazione dei dati
  - utilizzati dai programmi
  - indipendenti dalle strutture fisiche
- Esempi: relazionale, reticolare, gerarchico, a oggetti, basato su XML, NoSQL (document-based, colonnari, graph-based, RDF…)

Architettura (semplificata) di un DBMS



Schema di una specifica base di dati rappresentato secondo un **modello logico** (ad es. relazionale)

Rappresentazione interna della base di dati dipendente dallo **specifico DBMS** usato

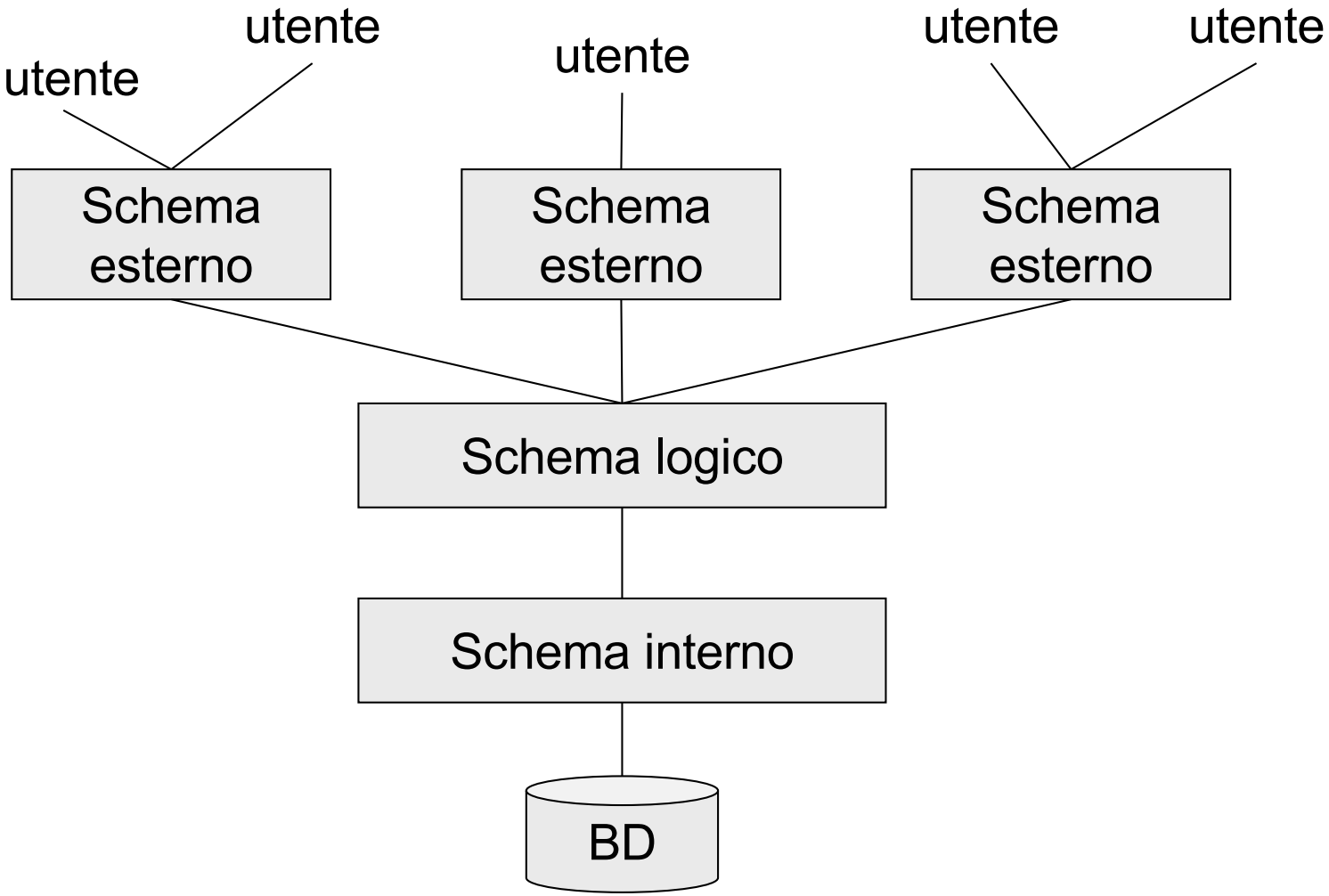
Architettura semplificata di un DBMS: schemi

- schema logico: descrizione della base di dati nel modello logico (ad es., la struttura della tabella)
- schema interno (o fisico): rappresentazione dello schema logico per mezzo di strutture di memorizzazione (file; ad es., record con puntatori, ordinati in un certo modo)

Indipendenza dei dati

- Il livello **logico** è **indipendente** da quello **fisico**:
  - una tabella è utilizzata nello stesso modo qualunque sia la sua realizzazione fisica (che può anche cambiare nel tempo)
- Perciò vedremo solo il livello logico e non quello fisico

Architettura standard (ANSI/SPARC) a tre livelli per DBMS



Architettura ANSI/SPARC: schemi

Schema **interno** (o **fisico**): rappresentazione dello schema logico per mezzo di strutture fisiche di memorizzazione

Schema **logico**: descrizione dell'intera base di dati nel modello logico "principale" del DBMS

Schema **esterno**: descrizione di parte della base di dati in un modello logico ("viste" parziali, derivate, anche in modelli diversi)

Una vista

Schema logico

Corsi

Corso	Docente	Aula
Basi di dati	Rossi	DS1
Sistemi	Neri	N3
Reti	Bruni	N3
Controlli	Bruni	G

Aule

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

Schema esterno

Corsi Sedi

Corso	Aula	Edificio	Piano
Basi di dati	DS1	OMI	Terra
Sistemi	N3	OMI	Terra
Reti	N3	OMI	Terra
Controlli	G	Pincherle	Primo

Indipendenza dei dati

- Conseguenza dell'articolazione in livelli
- L'accesso avviene solo tramite il livello **esterno** (che può coincidere con il livello logico)
- Due forme:
  - indipendenza fisica
  - indipendenza logica

Indipendenza fisica

- Il **livello logico e quello esterno** sono **indipendenti** da quello **fisico**
  - una relazione è utilizzata nello stesso modo qualunque sia la sua realizzazione fisica
  - la realizzazione fisica può cambiare senza che debbano essere modificati i programmi

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

Linguaggi per basi di dati

- Un altro contributo all'efficacia: disponibilità di vari linguaggi e interfacce
  - ⇨ linguaggi testuali interattivi (**SQL**)
  - ⇨ comandi (SQL) immersi in un linguaggio di programmazione ospite (Java, Python, C...)
  - ⇨ comandi (SQL) immersi in un linguaggio di programmazione ad hoc del DBMS
  - ⇨ con interfacce grafiche (senza linguaggio testuale)

SQL come linguaggio interattivo

Corsi

Corso	Docente	Aula
Basi di dati	Rossi	DS1
Sistemi	Neri	N3
Reti	Bruni	N3
Controlli	Bruni	G

Aule

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

- "Trovare i corsi tenuti in aule al piano terra"

SQL come linguaggio interattivo

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

Corso	Aula	Piano
Basi di dati	DS1	Terra
Sistemi	N3	Terra
Reti	N3	Terra

SQL immerso in linguaggio ospite (Java)

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.ResultSetMetaData;

public class esempioDB {
    private static String dbURL = "jdbc:postgresql://localhost:5432/postgres?user=postgres&password=prova";
   private static Connection conn = null;
   private static Statement stmt = null;

   public static void main(String args[]) {
        try {        Class.forName("org.postgresql.Driver").newInstance();
             conn = DriverManager.getConnection(dbURL);
             conn.setAutoCommit(false);
             System.out.println("Opened database successfully");

             stmt = conn.createStatement();
             ResultSet results = stmt.executeQuery(" SELECT Corso, Aula, Piano FROM Aule, Corsi WHERE Nome = Aula AND Piano =
        'Terra';");
             int numberCols = results.getMetaData().getColumnCount();
             while (results.next()) {
                for (int i=1; i<=numberCols; i++)
                {
                      String field = results.getString(i);
                      System.out.print(field + "\t");
                 }
                 System.out.println("");
             }
             results.close();
             stmt.close();

             if (stmt != null)
                stmt.close();
             if (conn != null) {
                DriverManager.getConnection(dbURL + ";shutdown=true");
                conn.close();
             }
         }
         catch (Exception except) {     except.printStackTrace(); } } }
```

SQL immerso in linguaggio ospite (C)

```
#include <libpq-fe.h>
int main(void) {
  PGconn *conn = PQconnectdb("postgresql://localhost");
  if (PQstatus(conn) == CONNECTION_OK) {
     PGresult *result = PQexec(conn, "SELECT Corso, Aula, Piano FROM Aule, Corsi
WHERE Nome = Aula AND Piano = 'Terra'");
     for (int i = 0; i < PQntuples(result); i++) {
       char *value = PQgetvalue(result, i, 0);
       if (value)
          printf("%s\n", value);
     }
     PQclear(result);
  }
  PQfinish(conn);
}
```

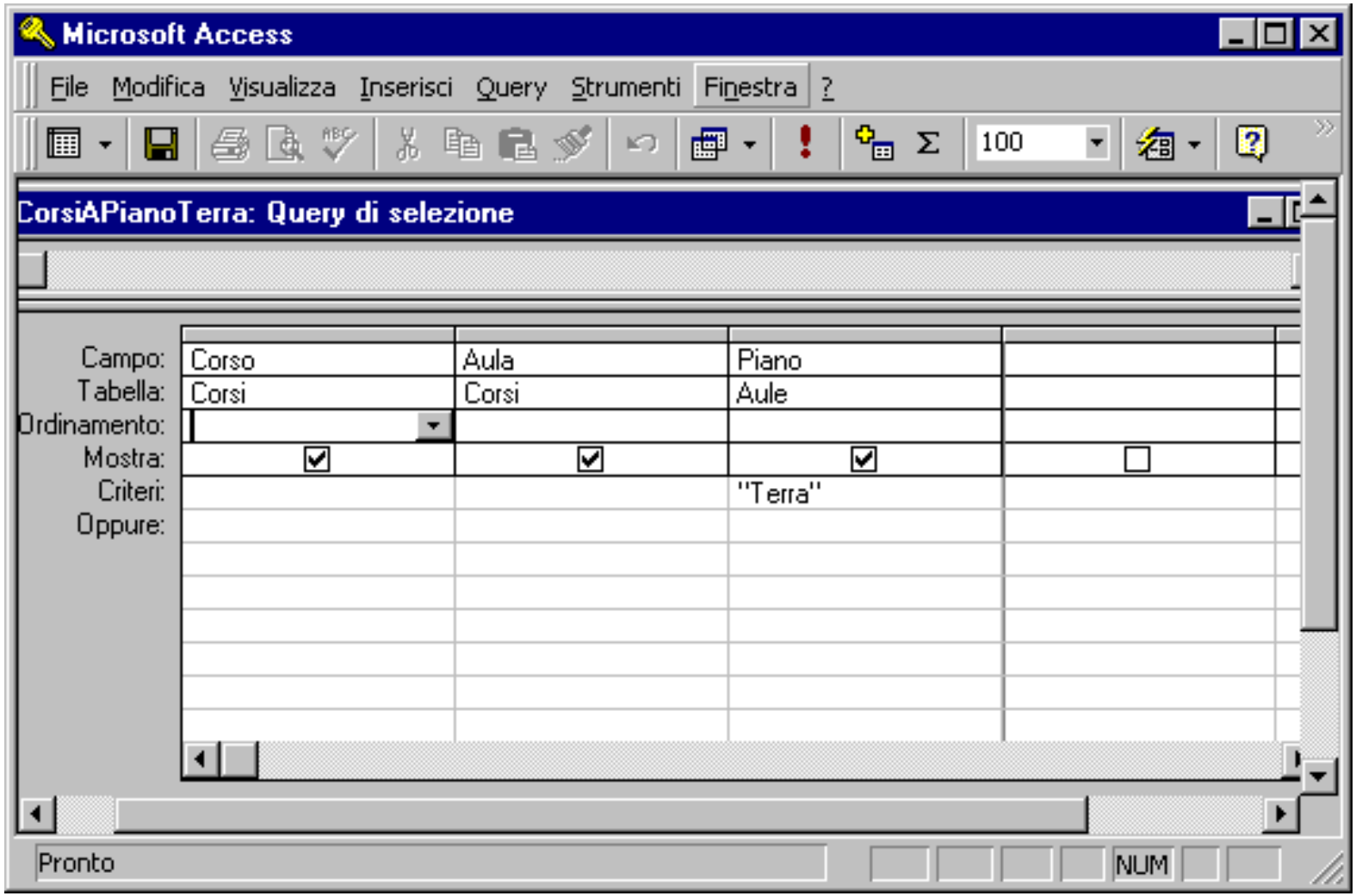
Compilare con:

```
clang main.c -I$(pg_config --includedir) -L$(pg_config --libdir) -lpq ./a.out
```

SQL in linguaggio ad hoc (Oracle PL/SQL)

```
declare  Stip number;
begin
     SELECT STIPENDIO INTO STIP FROM IMPIEGATO
     WHERE MATRICOLA = '575488' FOR UPDATE OF STIPENDIO;
     if Stip > 30 then
        UPDATE IMPIEGATO SET STIPENDIO = STIPENDIO * 1.1
             WHERE MATRICOLA = '575488';
     else
        UPDATE IMPIEGATO SET STIPENDIO = STIPENDIO * 1.15
             WHERE MATRICOLA = '575488';
     end if;
     commit;
  exception
     when no_data_found then
       INSERT INTO ERRORI
             VALUES('MATRICOLA INESISTENTE',SYSDATE);
  end;
```

Interazione non testuale (Access)



Una distinzione

data manipulation language (DML)

per l'interrogazione e l'aggiornamento di (istanze di) basi di dati

data definition language (DDL)

per la definizione di schemi (logici, esterni, fisici) e altre operazioni generali

Un'operazione DDL (sullo schema)

```
CREATE TABLE orario (
  insegnamento      CHAR(20),
  docente           CHAR(20),
  aula              CHAR(4),
  ora               CHAR(5)  );
```

Persone che interagiscono con un DBMS

- (progettisti e sviluppatori dei DBMS stessi)
- progettisti della base di dati e amministratori della base di dati (DBA)
- progettisti e programmatori di applicazioni che interagiscono con i DBMS
- utenti
  - utenti finali (terminalisti): eseguono procedure definite a priori (transazioni)
  - utenti "casuali": eseguono operazioni non previste a priori usando linguaggi interattivi

Database administrator (DBA)

- Persona o gruppo di persone responsabile del controllo centralizzato e della gestione del sistema, delle prestazioni, dell'affidabilità, delle autorizzazioni
- Le funzioni del DBA possono includere la progettazione, anche se in progetti complessi ci possono essere distinzioni

Vantaggi e svantaggi dei DBMS, 1

Pro:

- dati come risorsa comune, base di dati come modello della realtà
- gestione centralizzata con possibilità di standardizzazione ed “economia di scala”
- disponibilità di servizi integrati
- riduzione di ridondanze e inconsistenze
- indipendenza dei dati (favorisce lo sviluppo e la manutenzione delle applicazioni)

Vantaggi e svantaggi dei DBMS, 2

Contro:

- costo dei prodotti e della transizione verso di essi
- non scorporabilità delle funzionalità (con riduzione di efficienza)